

07-380 Notes

Pre-reading: Markov Chain Inference

Carnegie Mellon University
School of Computer Science

Contents

1	Background from the Bayes Nets Pre-reading (Assumed)	2
2	Reasoning over Time	2
2.1	Sequences of Observations	2
2.2	One Random Variable per Time Step	3
3	Markov Chains	3
3.1	The Markov Property	3
3.2	Transition Model and Stationarity	4
3.3	Example: Markov Chain Weather	5
4	Predicting Tomorrow's Weather	6
4.1	From $P(X_1)$ to $P(X_2)$	6
4.2	The Mini Joint	6
4.3	From $P(X_2)$ to $P(X_3)$	7
5	The Forward Algorithm (Simple Form)	7
5.1	Why One Step Is Enough	7
5.2	The Update	8
5.3	A Table on One Line	9
5.4	Example: Three Days of Weather	9
5.5	Many Days Later	10
6	Where This Is Going: Hidden Markov Models	11
7	Summary	11

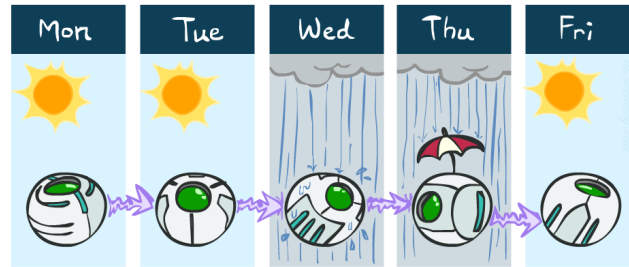
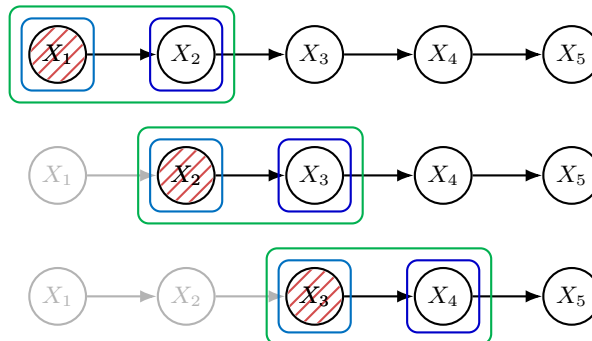


Illustration: Ketrina Yim / UC Berkeley AI



1 Background from the Bayes Nets Pre-reading (Assumed)

These notes pick up where the Bayes Nets pre-reading left off, and they use the following tools from it without re-deriving them.

Tool	Rule
Capital letters mean tables	$P(X_2)$ is a table with one entry per outcome of X_2 ; $P(X_2 = \text{sun})$ is one number.
Product rule	$P(A, B) = P(B A) P(A)$
Marginalization	$P(B) = \sum_a P(a, B)$
Bayes net joint	$P(X_1, \dots, X_N) = \prod_{i=1}^N P(X_i \text{Parents}(X_i))$, “nodes given parents”
Missing edges	The edges left out of a Bayes net, compared with the full chain rule, are its independence assumptions.

If any of these look unfamiliar, review the [Bayes Nets pre-reading](#) before continuing. In this pre-reading we apply these tools to one particular Bayes net, a chain of random variables indexed by time, and we work out the one calculation that the hidden Markov model lecture builds on: predicting the distribution of the next time step from the distribution of the current one.

2 Reasoning over Time

2.1 Sequences of Observations

Often, we want to reason about a sequence of observations rather than a single snapshot of the world:

- **Speech recognition:** the words change from moment to moment, and so does the sound.
- **Robot localization:** the robot moves, and every step brings a new range reading.
- **User attention:** what a user is looking at changes, and we want to predict what they do next.
- **Medical monitoring:** a patient’s health changes, and new readings keep arriving.

In each case something changes over time, and we want to keep track of it. The Bayes nets we have seen so far have no notion of time: every variable is a single snapshot. We need to introduce time (or space, for a sequence laid out along a line) into our models.

Our running example is the weather. Each day is either sunny or rainy, and the weather tomorrow depends on the weather today. We will build a model of a week like the one on the right, and then ask a question of the form “if we know how likely it is to be sunny today, how likely is it to be sunny tomorrow? The day after?”

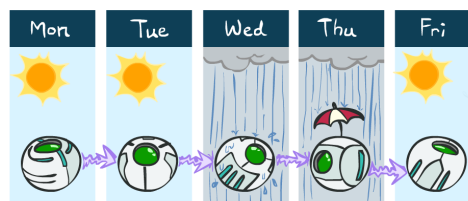


Illustration: Ketrina Yim / UC Berkeley AI

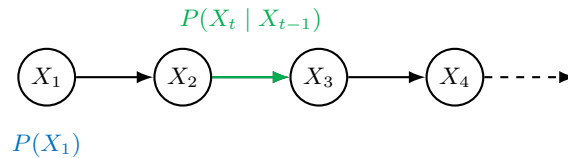
2.2 One Random Variable per Time Step

To put time into a Bayes net, we make a separate random variable for the quantity at each time step:

$$X_1, X_2, X_3, X_4, \dots$$

where X_t is the weather on day t . The subscript is the **time step**, and the value of X_t at a given time, sun or rain, is called the **state**. Each X_t has the same set of outcomes, here {sun, rain}, but a different day can be in a different state.

Now we connect the variables. Today's weather affects tomorrow's, so we draw an edge from X_1 to X_2 , from X_2 to X_3 , and so on. The result is a Bayes net shaped like a chain, with one node per time step. The dashed arrow says that the chain can keep going for as many days as we like.



Nothing about this picture is new. It is a Bayes net, so everything from the Bayes Nets pre-reading applies: one conditional probability table per node, nodes given parents, and the joint distribution is the product of the tables. What is new is that this particular net has a very regular shape, and that regularity is what lets us do inference on it one time step at a time.

3 Markov Chains

3.1 The Markov Property

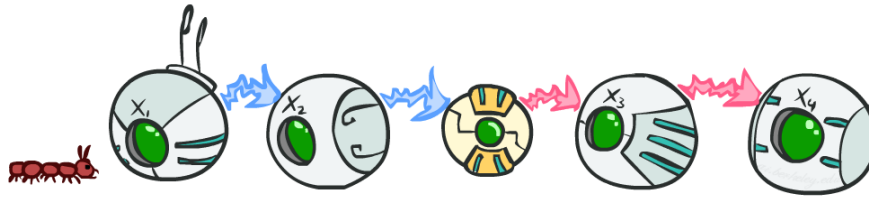
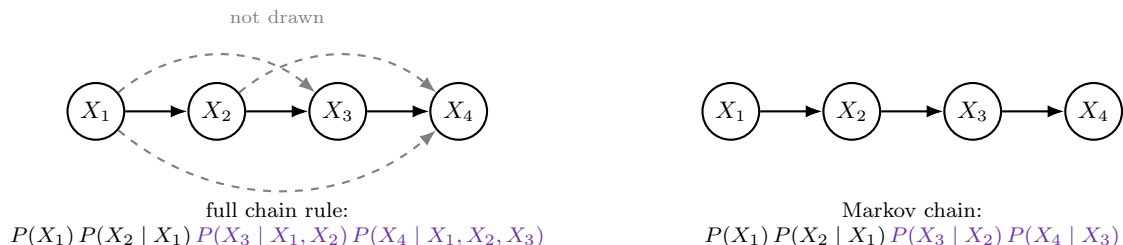


Illustration: Ketrina Yim / UC Berkeley AI

Look at which edges the chain leaves out. The full chain rule for four variables would have X_3 conditioned on both X_1 and X_2 , and X_4 conditioned on X_1 , X_2 , and X_3 . The chain draws only the edge from each day to the next:



As always, the missing edges are the independence assumptions. Leaving out $X_1 \rightarrow X_3$, $X_1 \rightarrow X_4$, and $X_2 \rightarrow X_4$ assumes that

- each time step depends only on the previous one: once we know today’s weather, knowing yesterday’s weather tells us nothing more about tomorrow’s, and
- the past and the future are independent given the present, for example $X_2 \perp\!\!\!\perp X_4 \mid X_3$.

This assumption is called the **Markov property**, and a chain of random variables that satisfies it is a **Markov chain**. It is the causal chain triple from the Bayes Nets pre-reading, $A \rightarrow B \rightarrow C$ with $C \perp\!\!\!\perp A \mid B$, repeated along the whole sequence. Strictly, this is the **first-order** Markov property, since each state looks back only one step. A second-order chain would draw edges from each X_t to both X_{t+1} and X_{t+2} , like a model of language in which the next word depends on the previous two. In this course a Markov chain is always first order.

Reading the joint off the chain, nodes given parents, gives

$$P(X_1, X_2, \dots, X_T) = P(X_1) \prod_{t=2}^T P(X_t \mid X_{t-1}).$$

A Markov chain is just a Bayes net that can be grown to any length T . If we truncate it at a fixed length, we can run any Bayes net reasoning we like on it.

3.2 Transition Model and Stationarity

A Markov chain has two kinds of parameters:

- The **initial distribution** $P(X_1)$, a table with one entry per state, which gets the chain started.
- The **transition probabilities** $P(X_t \mid X_{t-1})$, also called the **transition model** or the **dynamics**. This conditional probability table says how the state evolves from one time step to the next.

We make one more assumption, the **stationarity assumption**: the transition probabilities are the same at every time step. The table for $P(X_{10} \mid X_9)$ is the same table as $P(X_{20} \mid X_{19})$, so we only write it once as $P(X_t \mid X_{t-1})$. The weather on November 20th given the weather on November 19th follows the same table as November 30th given the 29th.

Stationarity does not say that the weather itself stays the same. It says that the *rule* for getting from one day to the next stays the same. If that seems too strong, since July is surely different from November, remember that the table can be as large as we like. If the state included the temperature to the nearest ten degrees, then “80s today given 90s yesterday” and “20s today given 30s yesterday” would both be entries in the one table, and both would apply on every day of the year.

3.3 Example: Markov Chain Weather

Here is the weather chain we will use for the rest of these notes. The state is $X_t \in \{\text{sun}, \text{rain}\}$, and

- the initial distribution says that day 1 is sunny for certain: $P(X_1 = \text{sun}) = 1.0$ and $P(X_1 = \text{rain}) = 0.0$, and
- the transition model says that a sunny day is usually followed by another sunny day, and a rainy day is usually followed by another rainy day.

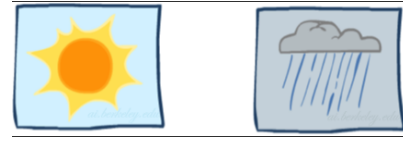


Illustration: Ketrina Yim / UC Berkeley AI

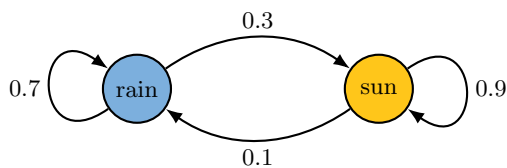
Three pictures of the same table: The transition model is a conditional probability table, exactly like the ones in the Bayes Nets pre-reading, and we can write it the way we wrote those, one row per combination of values. Because it has one variable on each side of the bar, it is also convenient to write it as a grid, with one row per given value of X_{t-1} and one column per value of X_t . The two tables below hold the same four numbers.

X_{t-1}	X_t	$P(X_t X_{t-1})$
sun	sun	0.9
sun	rain	0.1
rain	sun	0.3
rain	rain	0.7

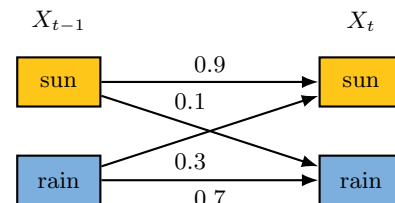
	X_t	
X_{t-1}	sun	rain
sun	0.9	0.1
rain	0.3	0.7

In the grid, each row is a distribution inside one given world (“given that yesterday was sunny”), so each row sums to one, while the columns do not have to. This is the rule from the Bayes Nets pre-reading about when a table sums to one: one specific value on the right of the bar, all values on the left.

Markov chains are often drawn in two other ways, and both are the *same table again*, not new information.



State diagram:
one node per *state*



Lattice:
one column per *time step*

In the **state diagram** there is one node per state and one arrow per table entry, including the self-loops for staying in the same state. The numbers on the arrows leaving a node sum to one. The **lattice** (or trellis) instead has one column per time step and one node per state in each column, so the arrows show the same four numbers as transitions from day $t - 1$ to day t . Whenever you see either picture, translate it back to the table $P(X_t | X_{t-1})$.

4 Predicting Tomorrow's Weather

4.1 From $P(X_1)$ to $P(X_2)$

We know the distribution of the weather on day 1, and we know the transition model. What is the probability that day 2 is sunny? We want $P(X_2 = \text{sun})$, which is not given anywhere. It is not conditioned on anything, and the only table that mentions X_2 is $P(X_2 | X_1)$, which is conditioned on X_1 . We get rid of the mismatch with two of the rules from Section 1: un-marginalize to bring X_1 into the picture, then use the product rule to split the joint into tables we have.

$$\begin{aligned}
 P(X_2 = \text{sun}) &= \sum_{x_1} P(X_1 = x_1, X_2 = \text{sun}) && \text{marginalization} \\
 &= \sum_{x_1} P(X_2 = \text{sun} | X_1 = x_1) P(X_1 = x_1) && \text{product rule} \\
 &= P(X_2 = \text{sun} | X_1 = \text{sun}) P(X_1 = \text{sun}) && \text{expand the sum} \\
 &\quad + P(X_2 = \text{sun} | X_1 = \text{rain}) P(X_1 = \text{rain}) \\
 &= 0.9 \cdot 1.0 + 0.3 \cdot 0.0 = 0.9 && \text{look up the numbers}
 \end{aligned}$$

The colors will stay fixed for the rest of these notes, in the equations and in the pictures: **red** for a summation, **purple** for a joint distribution, **green** for the transition model, **navy** for the table we already have from the previous time step, and **dark blue** for the table we are computing.

The second entry of the table $P(X_2)$ follows the same steps with rain in place of sun on the left of the bar:

$$P(X_2 = \text{rain}) = 0.1 \cdot 1.0 + 0.7 \cdot 0.0 = 0.1,$$

and the two entries sum to one, as the table $P(X_2)$ must. We have turned $P(X_1)$ into $P(X_2)$ using nothing but the **transition model**.

You may be able to get 0.9 by intuition: day 1 is sunny for sure, so day 2 is sunny with probability 0.9. That is fine, but make sure you can also get it by following the rules step by step, because the next example will not be a special case and because we are going to ask a computer to do these steps for us.

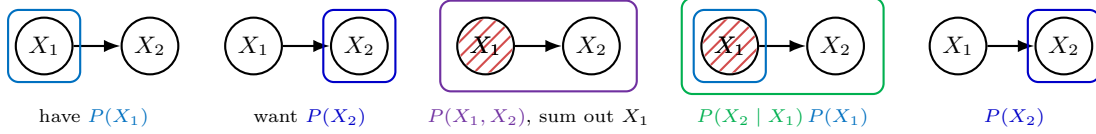
4.2 The Mini Joint

It is worth pausing on the purple term in the derivation, $P(X_1, X_2)$. It is a joint distribution, but only over two of the variables in the chain, so let's call it a **mini joint**. It is built from the two tables we have by the product rule, $P(X_1, X_2) = P(X_2 | X_1) P(X_1)$, one entry at a time:

X_1	X_2	$P(X_1, X_2) = P(X_2 X_1) P(X_1)$
sun	sun	$(0.9)(1.0) = 0.9$
sun	rain	$(0.1)(1.0) = 0.1$
rain	sun	$(0.3)(0.0) = 0.0$
rain	rain	$(0.7)(0.0) = 0.0$

Marginalizing X_1 out of the mini joint means adding up the rows that share a value of X_2 : $P(X_2 = \text{sun}) = 0.9 + 0.0$ and $P(X_2 = \text{rain}) = 0.1 + 0.0$.

The picture below is how we will draw this calculation on the chain, following the derivation in Section 4.1 frame by frame. We start with the table we have, $P(X_1)$, and the table we want, $P(X_2)$. Marginalization forms the mini joint over both days (purple box) and sums out X_1 (red stripes). The product rule breaks the mini joint into the transition model, which spans both days, times the table we already have for day 1. Looking up the numbers leaves us with $P(X_2)$ alone, and X_1 is gone for good.



4.3 From $P(X_2)$ to $P(X_3)$

Now we know $P(X_2)$, with $P(X_2 = \text{sun}) = 0.9$ and $P(X_2 = \text{rain}) = 0.1$. What is $P(X_3 = \text{sun})$? Take a minute to work it out yourself before reading on. The steps are the same as in Section 4.1, with every subscript moved up by one:

$$\begin{aligned}
 P(X_3 = \text{sun}) &= \sum_{x_2} P(X_2 = x_2, X_3 = \text{sun}) \\
 &= \sum_{x_2} P(X_3 = \text{sun} \mid X_2 = x_2) P(X_2 = x_2) \\
 &= 0.9 \cdot 0.9 + 0.3 \cdot 0.1 = 0.81 + 0.03 = 0.84.
 \end{aligned}$$

Likewise $P(X_3 = \text{rain}) = 0.1 \cdot 0.9 + 0.7 \cdot 0.1 = 0.09 + 0.07 = 0.16$, and again the two entries sum to one. Here intuition is less help. A tempting answer is 0.81, but that is the probability that day 2 *and* day 3 are both sunny, which is only one of the two ways day 3 can be sunny. The other way, rain on day 2 followed by sun on day 3, contributes the 0.03.

The key point: To compute $P(X_3)$ we used $P(X_2)$ and the transition model, and nothing else. We did not need $P(X_1)$, because X_1 was summed out when we computed $P(X_2)$, and it never comes back. Having computed $P(X_3)$, we can use it in exactly the same way to get $P(X_4)$, then $P(X_5)$, and so on down the chain, one step at a time.

5 The Forward Algorithm (Simple Form)

5.1 Why One Step Is Enough

Section 4.3 claimed that $P(X_3)$ depends on the past only through $P(X_2)$. Let's check that claim against the Bayes Nets pre-reading, where the recipe for a query is to build the whole joint and sum out everything we do not want. Suppose we know $P(X_4)$ and want $P(X_5)$. The two routes are

$$\begin{aligned}
 \text{one step:} \quad P(X_5) &= \sum_{x_4} P(x_4, X_5) = \sum_{x_4} P(X_5 \mid x_4) P(x_4), \\
 \text{whole joint:} \quad P(X_5) &= \sum_{x_1, x_2, x_3, x_4} P(x_1, x_2, x_3, x_4, X_5).
 \end{aligned}$$

Here we are using the shorthand from the Bayes Nets pre-reading: a lowercase x_4 inside $P(\cdot)$ means " $X_4 = x_4$ ", and $\sum_{x_4}$ loops over sun and rain. The whole-joint route looks very different, but a few

steps turn it into the one-step route:

$$\begin{aligned}
 P(X_5) &= \sum_{x_1, x_2, x_3, x_4} P(x_1, x_2, x_3, x_4, X_5) && \text{marginalization} \\
 &= \sum_{x_1, x_2, x_3, x_4} P(X_5 | x_4) P(x_4 | x_3) P(x_3 | x_2) P(x_2 | x_1) P(x_1) && \text{nodes given parents} \\
 &= \sum_{x_4} P(X_5 | x_4) \sum_{x_1, x_2, x_3} P(x_4 | x_3) P(x_3 | x_2) P(x_2 | x_1) P(x_1) && \text{pull the factor out} \\
 &= \sum_{x_4} P(X_5 | x_4) \sum_{x_1, x_2, x_3} P(x_1, x_2, x_3, x_4) && \text{nodes given parents, backwards} \\
 &= \sum_{x_4} P(X_5 | x_4) P(x_4) && \text{marginalization}
 \end{aligned}$$

The second line is the joint of the chain, nodes given parents. In the third line, the factor $P(X_5 | x_4)$ does not involve x_1 , x_2 , or x_3 , so it can be pulled outside the sums over those variables. This is just the distributive law, $ab + ac = a(b + c)$, applied to a lot of terms at once. The remaining product in the fourth line is itself a Bayes net joint, the joint of the first four days, and summing x_1 , x_2 , and x_3 out of it leaves $P(x_4)$.

So the two routes agree, and the one-step route is far cheaper. The whole-joint route adds up $2^4 = 16$ terms, each a product of five numbers, and the count doubles with every day we add. The one-step route adds up two terms, provided we already have $P(X_4)$ from the step before, so the work grows by a fixed amount per day instead of doubling. Pushing a summation inward past the factors that do not involve it is the same move as in variable elimination, and the chain is the easiest case of it.

5.2 The Update

Putting Sections 4.1 through 5.1 together: if we know the distribution of the state at time $t - 1$, the distribution at time t is

$$P(X_t) = \sum_{x_{t-1}} P(x_{t-1}, X_t) = \sum_{x_{t-1}} P(X_t | x_{t-1}) P(x_{t-1})$$


This update, iterated from $t = 2$ onward starting at the initial distribution $P(X_1)$, is the **forward algorithm** in its simple form. Each iteration forms the mini joint over two consecutive time steps, sums out the earlier one, and hands the result to the next iteration, where the **dark blue** table we just computed becomes the **navy** table we have. The name comes from marching forward through the chain. In lecture we will meet the full forward algorithm, which does the same march on a hidden Markov model.

Two things to keep straight when using the update:

- **X_t is capital.** Each iteration must compute the *whole* table $P(X_t)$, every entry, not just the one we happen to be curious about, because the next iteration sums over all of them. With K states, each iteration computes K entries, each a sum of K products, for K^2 multiplications per time step, independent of t .
- **Which table is which.** The green factor is the same transition table at every iteration (stationarity). The navy factor is the output of the previous iteration, so it changes every time.

A note on indexing: some examples, and the lecture slides, start the clock at $t = 0$ rather than $t = 1$, so that the initial distribution is $P(X_0)$. Nothing changes except the subscripts.

5.3 A Table on One Line

Writing out one entry at a time gets tedious, so we will often write a probability table on one line, in angle brackets, with the entries in a fixed order. For the weather, the order is $\langle \text{sun}, \text{rain} \rangle$, so

$$P(X_1) = \langle 1.0, 0.0 \rangle \quad \text{means} \quad P(X_1 = \text{sun}) = 1.0 \quad \text{and} \quad P(X_1 = \text{rain}) = 0.0.$$

Each row of the transition grid is also a table, the distribution of today's weather given one value of yesterday's, so

$$P(X_t \mid X_{t-1} = \text{sun}) = \langle 0.9, 0.1 \rangle, \quad P(X_t \mid X_{t-1} = \text{rain}) = \langle 0.3, 0.7 \rangle.$$

With this notation the two entries of the update can be computed at once. Multiplying a table by a number scales every entry, and adding two tables adds them entry by entry:

$$P(X_2) = \sum_{x_1} P(X_2 \mid x_1) P(x_1) = 1.0 \langle 0.9, 0.1 \rangle + 0.0 \langle 0.3, 0.7 \rangle = \langle 0.9, 0.1 \rangle.$$

Compare this with Section 4.1: the first entries, $1.0 \cdot 0.9 + 0.0 \cdot 0.3 = 0.9$, are the sun calculation, and the second entries are the rain calculation. In words, the new table is a weighted sum of the rows of the transition grid, where the weights are the previous table.

The angle brackets are just a table written sideways. They are not a vector in the linear algebra sense, and nothing here is a dot product or a matrix product. The connection to matrices is real, but we will not use it in this course.

5.4 Example: Three Days of Weather

Let's run the forward algorithm on the same transition model from a less certain start. Suppose we have no idea what the weather is on day 1, so $P(X_1) = \langle 0.5, 0.5 \rangle$, and we want the distribution of the weather on each of the next three days.

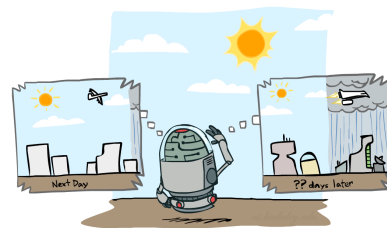


Illustration: Ketrina Yim / UC Berkeley AI

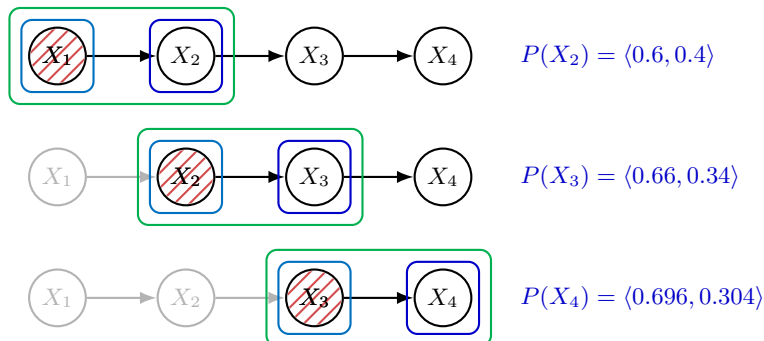
Day 2 uses $P(X_1)$, day 3 uses the day 2 answer, and day 4 uses the day 3 answer:

$$\begin{aligned} \text{Day 2: } P(X_2) &= \sum_{x_1} P(X_2 \mid x_1) P(x_1) = 0.5 \langle 0.9, 0.1 \rangle + 0.5 \langle 0.3, 0.7 \rangle \\ &= \langle 0.45 + 0.15, 0.05 + 0.35 \rangle = \langle 0.6, 0.4 \rangle \end{aligned}$$

$$\begin{aligned} \text{Day 3: } P(X_3) &= \sum_{x_2} P(X_3 \mid x_2) P(x_2) = 0.6 \langle 0.9, 0.1 \rangle + 0.4 \langle 0.3, 0.7 \rangle \\ &= \langle 0.54 + 0.12, 0.06 + 0.28 \rangle = \langle 0.66, 0.34 \rangle \end{aligned}$$

$$\begin{aligned} \text{Day 4: } P(X_4) &= \sum_{x_3} P(X_4 \mid x_3) P(x_3) = 0.66 \langle 0.9, 0.1 \rangle + 0.34 \langle 0.3, 0.7 \rangle \\ &= \langle 0.594 + 0.102, 0.066 + 0.238 \rangle = \langle 0.696, 0.304 \rangle \end{aligned}$$

Each line uses the same green numbers, and the dark blue answer on each line becomes the navy input on the next. Every result sums to one, which is a quick check worth doing at every step. The picture below shows the three iterations on the chain, drawn as in Section 4.2: the table we have (navy) around the earlier day, the transition model spanning both days, the earlier day summed out (red stripes), and the table we are computing (dark blue) around the later day. Days that have already been summed out are grayed, because the algorithm never looks at them again.



5.5 Many Days Later

The robot in the cartoon is also wondering about the weather many days from now. We can keep iterating. The table below shows $P(X_t = \text{sun})$ for the first several days of our two examples, the one that started with a certainly sunny day and the one that started with no idea.

t	1	2	3	4	5	6	...
$P(X_t = \text{sun})$ from $P(X_1) = \langle 1.0, 0.0 \rangle$	1.0	0.9	0.84	0.804	0.7824	0.76944	$\rightarrow 0.75$
$P(X_t = \text{sun})$ from $P(X_1) = \langle 0.5, 0.5 \rangle$	0.5	0.6	0.66	0.696	0.7176	0.73056	$\rightarrow 0.75$

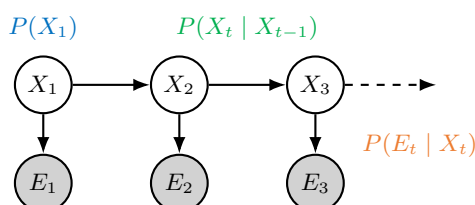
Both runs are heading for the same place, $\langle 0.75, 0.25 \rangle$, from opposite directions. That table is special: plugging it into the update gives it right back, $0.75 \cdot 0.9 + 0.25 \cdot 0.3 = 0.75$, so once the chain reaches it, it stays there. It is called the **stationary distribution** of the chain. The further into the future we predict, the less the starting distribution matters and the closer the prediction gets to the stationary distribution, which is why the robot's picture of "?? days later" is so vague. We will not need stationary distributions in lecture, but the pattern in the table is worth noticing: with only the transition model to go on, every prediction drifts toward the same table, and whatever we knew about day 1 fades away.

6 Where This Is Going: Hidden Markov Models

In the weather chain, we never actually observed the weather; we only predicted it from the day before. In most real problems the state is *hidden* and we get a noisy observation of it at each time step: a noisy distance reading rather than the ghost's location, the microphone signal rather than the word, the umbrella rather than the rain. A **hidden Markov model** (HMM) is a Markov chain over states X_t with an **evidence** variable E_t hanging off each state, and one more table, the **sensor model** $P(E_t | X_t)$.



Illustration: Ketrina Yim / UC Berkeley AI



It is still a Bayes net, so its joint is still nodes given parents, now with an orange factor for each time step as well as a green one. The question we will ask of it is the **filtering** query: what is the current state, given all of the evidence so far, $P(X_t | e_1, \dots, e_t)$? The algorithm that answers it marches forward through the chain exactly as in Section 5.2, with one extra half-step per iteration to fold in the new evidence using the sensor model. The Markov chain update in these notes is the part of that algorithm that moves time forward, and it is the part to be comfortable with before lecture.

7 Summary

Term	What it is
State X_t	One random variable per time step, all with the same set of outcomes. The value at time t is the state.
Markov property	Each time step depends only on the previous one: past and future are independent given the present. The chain Bayes net draws only the edges $X_{t-1} \rightarrow X_t$, and the missing edges are this assumption.
Markov chain	A Bayes net shaped like a chain, with two kinds of parameters: the initial distribution $P(X_1)$ and the transition model $P(X_t X_{t-1})$. Its joint is $P(X_1) \prod_t P(X_t X_{t-1})$.
Stationarity	The transition table is the same at every time step.
Transition model pictures	The grid (rows sum to one), the state diagram (one node per state), and the lattice (one column per time step) all show the same table.
Forward algorithm (simple form)	$P(X_t) = \sum_{x_{t-1}} P(X_t x_{t-1}) P(x_{t-1})$: form the mini joint over X_{t-1} and X_t , sum out X_{t-1} , pass $P(X_t)$ forward. Compute the whole table at every step.
One-line tables	$\langle 0.6, 0.4 \rangle$ is the table with $P(\text{sun}) = 0.6$ and $P(\text{rain}) = 0.4$; the update is a weighted sum of the rows of the transition grid, with the previous table as the weights.