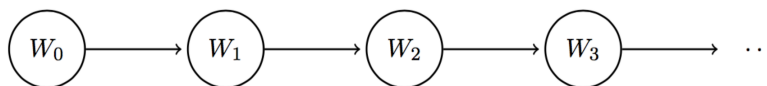


Markov Models

In previous notes, we talked about Bayes' nets and how they are a wonderful structure used for compactly representing relationships between random variables. We'll now cover a very intrinsically related structure called a **Markov model**, which for the purposes of this course can be thought of as analogous to a chain-like, infinite-length Bayes' net. The running example we'll be working with in this section is the day-to-day fluctuations in weather patterns. Our weather model will be **time-dependent** (as are Markov models in general), meaning we'll have a separate random variable for the weather on each day. If we define W_i as the random variable representing the weather on day i , the Markov model for our weather example would look like the following Bayes Net:



What information should we store about the random variables involved in our Markov model? To track how our quantity under consideration (in this case, the weather) changes over time, we need to know both its **initial distribution** at time $t = 0$ and some sort of **transition model** that characterizes the probability of moving from one state to another between timesteps. The initial distribution of a Markov model is enumerated by the probability table given by $Pr(W_0)$ and the transition model of transitioning from state i to $i + 1$ is given by $Pr(W_{i+1}|W_i)$. Note that this transition model implies that the value of W_{i+1} is conditionally dependent only on the value of W_i . In other words, the weather at time $t = i + 1$ satisfies the **Markov property** or **memoryless property**, and is independent of the weather at all other timesteps besides $t = i$.

Using our Markov model for weather, if we wanted to reconstruct the joint between W_0 , W_1 , and W_2 using the chain rule, we would want:

$$Pr(W_0, W_1, W_2) = Pr(W_0)Pr(W_1|W_0)Pr(W_2|W_1, W_0)$$

However, with our assumption that the Markov property holds true and $W_0 \perp\!\!\!\perp W_2 | W_1$, the joint simplifies to:

$$Pr(W_0, W_1, W_2) = Pr(W_0)Pr(W_1|W_0)Pr(W_2|W_1)$$

And we have everything we need to calculate this from the Markov model. More generally, Markov models make the following independence assumption at each timestep: $W_{i+1} \perp\!\!\!\perp \{W_0, \dots, W_{i-1}\} | W_i$. This allows us to reconstruct the joint distribution for the first $n + 1$ variables via the chain rule as follows:

$$Pr(W_0, W_1, \dots, W_n) = Pr(W_0)Pr(W_1|W_0)Pr(W_2|W_1) \dots Pr(W_n|W_{n-1}) = Pr(W_0) \prod_{i=0}^{n-1} Pr(W_{i+1}|W_i)$$

A final assumption that's typically made in Markov models is that the transition model is **time-homogeneous**. In other words, for all values of i , $Pr(W_{i+1}|W_i)$ is identical. This allows us to represent a Markov model with only two tables: one for $Pr(W_0)$ and one for $Pr(W_{i+1}|W_i)$.

The Mini-Forward Algorithm

We now know how to compute the joint distribution across timesteps of a Markov model. However, this doesn't explicitly help us answer the question of the distribution of the weather on some given day t . As with any Bayes Net, we can use **inference by elimination** to compute the joint and then **marginalize** (sum out) over all other variables. But this is extremely inefficient, since with j variables with domains of d possible values, the size of the joint distribution is $O(d^j)$. Instead, we'll present a more efficient technique analogous to **inference by elimination**, called the **mini-forward algorithm**.

Here's how it works. By properties of marginalization, we know that

$$Pr(W_{i+1}) = \sum_{w_i} Pr(w_i, W_{i+1})$$

By the chain rule we can reexpress this as follows:

$$Pr(W_{i+1}) = \sum_{w_i} Pr(W_{i+1}|w_i)Pr(w_i)$$

This equation should make some intuitive sense - to compute the distribution of the weather at timestep $i+1$, we look at the probability distribution at timestep i given by $Pr(W_i)$ and "advance" this model a timestep with our transition model $Pr(W_{i+1}|W_i)$. With this equation, we can iteratively compute the distribution of the weather at any timestep of our choice by starting with our initial distribution $Pr(W_0)$ and using it to compute $Pr(W_1)$, then in turn using $Pr(W_1)$ to compute $Pr(W_2)$, and so on. Let's walk through an example, using the following initial distribution and transition model:

W_0	$Pr(W_0)$	W_{i+1}	W_i	$Pr(W_{i+1} W_i)$
<i>sun</i>	0.8	<i>sun</i>	<i>sun</i>	0.6
<i>rain</i>	0.2	<i>rain</i>	<i>sun</i>	0.4
		<i>sun</i>	<i>rain</i>	0.1
		<i>rain</i>	<i>rain</i>	0.9

Using the mini-forward algorithm we can compute $Pr(W_1)$ as follows:

$$\begin{aligned} Pr(W_1 = \textit{sun}) &= \sum_{w_0} Pr(W_1 = \textit{sun}|w_0)Pr(w_0) \\ &= Pr(W_1 = \textit{sun}|W_0 = \textit{sun})Pr(W_0 = \textit{sun}) + Pr(W_1 = \textit{sun}|W_0 = \textit{rain})Pr(W_0 = \textit{rain}) \\ &= 0.6 \cdot 0.8 + 0.1 \cdot 0.2 = \boxed{0.5} \\ Pr(W_1 = \textit{rain}) &= Pr(W_1 = \textit{rain}|w_0)Pr(w_0) \\ &= Pr(W_1 = \textit{rain}|W_0 = \textit{sun})Pr(W_0 = \textit{sun}) + Pr(W_1 = \textit{rain}|W_0 = \textit{rain})Pr(W_0 = \textit{rain}) \\ &= 0.4 \cdot 0.8 + 0.9 \cdot 0.2 = \boxed{0.5} \end{aligned}$$

Hence our distribution for $Pr(W_1)$ is

W_1	$Pr(W_1)$
<i>sun</i>	0.5
<i>rain</i>	0.5

Notably, the probability that it will be sunny has decreased from 80% at time $t=0$ to only 50% at time $t=1$. This is a direct result of our transition model, which favors transitioning to rainy days over sunny days. This gives rise to a natural follow-up question: could the probabilities of being in any state *not* change? We'll address this in the following section.

Stationary Distribution

To solve the problem stated above, we must compute the **stationary distribution** of the weather. As the name suggests, the stationary distribution is one that remains the same after the passage of time, i.e.

$$Pr(W_{t+1}) = Pr(W_t)$$

We can compute these converged probabilities of being in a given state by combining the above equivalence with the same equation used by the mini-forward algorithm:

$$Pr(W_{t+1}) = Pr(W_t) = \sum_{w_t} Pr(W_{t+1}|w_t)Pr(w_t)$$

For our weather example, this gives us the following two equations:

$$\begin{aligned} Pr(W_t = sun) &= Pr(W_{t+1} = sun|W_t = sun)Pr(W_t = sun) + Pr(W_{t+1} = sun|W_t = rain)Pr(W_t = rain) \\ &= 0.6 \cdot Pr(W_t = sun) + 0.1 \cdot Pr(W_t = rain) \\ Pr(W_t = rain) &= Pr(W_{t+1} = rain|W_t = sun)Pr(W_t = sun) + Pr(W_{t+1} = rain|W_t = rain)Pr(W_t = rain) \\ &= 0.4 \cdot Pr(W_t = sun) + 0.9 \cdot Pr(W_t = rain) \end{aligned}$$

We now have exactly what we need to solve for the stationary distribution, a system of 2 equations in 2 unknowns! We can get a third equation by using the fact that $Pr(W_t)$ is a probability distribution and so must sum to 1:

$$\begin{aligned} Pr(W_t = sun) &= 0.6 \cdot Pr(W_t = sun) + 0.1 \cdot Pr(W_t = rain) \\ Pr(W_t = rain) &= 0.4 \cdot Pr(W_t = sun) + 0.9 \cdot Pr(W_t = rain) \\ 1 &= Pr(W_t = sun) + Pr(W_t = rain) \end{aligned}$$

Solving this system of equations yields $Pr(W_t = sun) = 0.2$ and $Pr(W_t = rain) = 0.8$. Hence the table for our stationary distribution, which we'll henceforth denote as $Pr(W_\infty)$, is the following:

W_∞	$Pr(W_\infty)$
<i>sun</i>	0.2
<i>rain</i>	0.8

To verify this result, let's apply the transition model to the stationary distribution:

$$\begin{aligned} Pr(W_{\infty+1} = sun) &= Pr(W_{\infty+1} = sun|W_\infty = sun)Pr(W_\infty = sun) + Pr(W_{\infty+1} = sun|W_\infty = rain)Pr(W_\infty = rain) \\ &= 0.6 \cdot 0.2 + 0.1 \cdot 0.8 = \boxed{0.2} \\ Pr(W_{\infty+1} = rain) &= Pr(W_{\infty+1} = rain|W_\infty = sun)Pr(W_\infty = sun) + Pr(W_{\infty+1} = rain|W_\infty = rain)Pr(W_\infty = rain) \\ &= 0.4 \cdot 0.2 + 0.9 \cdot 0.8 = \boxed{0.8} \end{aligned}$$

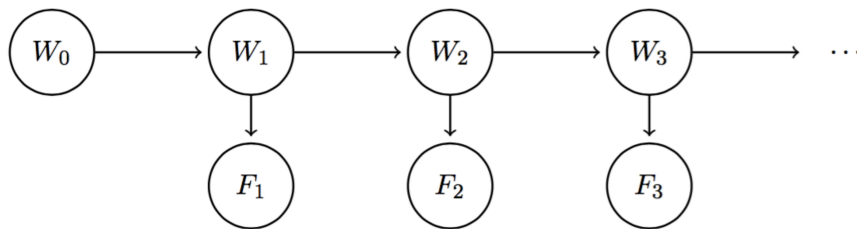
As expected, $Pr(W_{\infty+1}) = Pr(W_\infty)$. In general, if W_t had a domain of size k , the equivalence

$$Pr(W_t) = \sum_{w_t} Pr(W_{t+1}|w_t)Pr(w_t)$$

yields a system of k equations, which we can use to solve for the stationary distribution.

Hidden Markov Models

With Markov models, we saw how we could model change over time through a chain of random variables. For example, if we want to know the weather on day 10 from the Markov model above, we begin with the initial distribution $Pr(W_0)$ and use the mini-forward algorithm to compute $Pr(W_{10})$. However, between time $t = 0$ and time $t = 10$, we may collect new meteorological evidence that might affect our belief of the weather probability distribution at a given timestep. For instance, if we use mini-forward to predict an 80% chance of rain on day 10, but there are clear skies on the night of day 9, that 80% probability might drop drastically. This is exactly what the **Hidden Markov Model** helps us with - it allows us to observe some evidence at each timestep, which can potentially affect the belief distribution at each of the states. The Hidden Markov Model for our weather model can be described using the following Bayes' net structure:



Unlike vanilla Markov models, we now have two different types of nodes. To make this distinction, we'll call each W_i a **state variable** and each weather forecast F_i an **evidence variable**. Since W_i encodes our belief of the probability distribution for the weather on day i , it should be a natural result that the weather forecast for day i is conditionally dependent upon this belief. The model implies similar conditional independence relationships as standard Markov models, with an additional set of relationships for the evidence variables:

$$\begin{aligned}
 F_1 &\perp\!\!\!\perp W_0 | W_1 \\
 \forall i &= 2, \dots, n; \quad W_i \perp\!\!\!\perp \{W_0, \dots, W_{i-2}, F_1, \dots, F_{i-1}\} | W_{i-1} \\
 \forall i &= 2, \dots, n; \quad F_i \perp\!\!\!\perp \{W_0, \dots, W_{i-1}, F_1, \dots, F_{i-1}\} | W_i
 \end{aligned}$$

Just like Markov models, Hidden Markov Models make the assumption that the transition model $Pr(W_{i+1}|W_i)$ is time-homogeneous. Hidden Markov Models make the additional simplifying assumption that the **sensor model** $Pr(F_i|W_i)$ is time-homogeneous as well. Hence any Hidden Markov Model can be represented compactly with just three probability tables: the initial distribution, the transition model, and the sensor model.

As a final point on notation, we'll define the **belief distribution** at time i with all evidence F_1, \dots, F_i observed up to date:

$$B(W_i) = Pr(W_i | f_1, \dots, f_i)$$

Similarly, we'll define $B'(W_i)$ as the belief distribution at time i with evidence f_1, \dots, f_{i-1} observed:

$$B'(W_i) = Pr(W_i | f_1, \dots, f_{i-1})$$

Defining e_i as evidence observed at timestep i , you might sometimes see the aggregated evidence from timesteps $1 \leq i \leq t$ reexpressed in the following form:

$$e_{1:t} = e_1, \dots, e_t$$

Under this notation, $Pr(W_i | f_1, \dots, f_{i-1})$ can be written as $Pr(W_i | f_{1:(i-1)})$. This notation will become relevant in the upcoming sections, where we'll discuss time elapse updates that iteratively incorporate new evidence into our weather model.

The Forward Algorithm

Using the conditional probability assumptions stated above and marginalization properties of conditional probability tables, we can derive a relationship between $B(W_i)$ and $B'(W_{i+1})$ that's of the same form as the update rule for the mini-forward algorithm. We begin by using marginalization:

$$B'(W_{i+1}) = Pr(W_{i+1}|f_1, \dots, f_i) = \sum_{w_i} Pr(W_{i+1}, w_i|f_1, \dots, f_i)$$

This can be reexpressed then with the chain rule as follows:

$$B'(W_{i+1}) = Pr(W_{i+1}|f_1, \dots, f_i) = \sum_{w_i} Pr(W_{i+1}|w_i, f_1, \dots, f_i) Pr(w_i|f_1, \dots, f_i)$$

Noting that $Pr(w_i|f_1, \dots, f_i)$ is simply $B(w_i)$ and that $W_{i+1} \perp\!\!\!\perp \{f_1, \dots, f_i\} | W_i$, this simplifies to our final relationship between $B(W_i)$ to $B'(W_{i+1})$:

$$B'(W_{i+1}) = \sum_{w_i} Pr(W_{i+1}|w_i) B(w_i)$$

Now let's consider how we can derive a relationship between $B'(W_{i+1})$ and $B(W_{i+1})$. From Bayes' rule, we can see that

$$B(W_{i+1}) = Pr(W_{i+1}|f_1, \dots, f_{i+1}) = \frac{Pr(W_{i+1}, f_{i+1}|f_1, \dots, f_i)}{Pr(f_{i+1}|f_1, \dots, f_i)}$$

When dealing with conditional probabilities, we can delay normalization until necessary. That is, since the denominator in the above expansion of $B(W_{i+1})$ is the same for all the entries in the CPT for $B(W_{i+1})$, we skip actually dividing by $Pr(f_{i+1}|f_1, \dots, f_i)$. Instead, we note that $B(W_{i+1})$ is proportional to the numerator:

$$B(W_{i+1}) \propto Pr(W_{i+1}, f_{i+1}|f_1, \dots, f_i)$$

with a constant of proportionality equal to $Pr(f_{i+1}|f_1, \dots, f_i)$. Whenever we decide we want to recover the belief distribution $B(W_{i+1})$, we can **normalize** by dividing each computed value by this constant of proportionality. Now, using the chain rule we can observe the following:

$$B(W_{i+1}) \propto Pr(W_{i+1}, f_{i+1}|f_1, \dots, f_i) = Pr(f_{i+1}|W_{i+1}, f_1, \dots, f_i) Pr(W_{i+1}|f_1, \dots, f_i)$$

By the conditional independence assumptions associated with Hidden Markov Models stated previously, $Pr(f_{i+1}|W_{i+1}, f_1, \dots, f_i) = Pr(f_{i+1}|W_{i+1})$ and by definition $Pr(W_{i+1}|f_1, \dots, f_i) = B'(W_{i+1})$. This allows us to express the relationship between $B'(W_{i+1})$ and $B(W_{i+1})$ in it's final form:

$$B(W_{i+1}) \propto Pr(f_{i+1}|W_{i+1}) B'(W_{i+1})$$

Combining the two relationships we've just derived yields an iterative algorithm known as the **forward algorithm**, the Hidden Markov Model analog of the mini-forward algorithm from earlier:

$$B(W_{i+1}) \propto Pr(f_{i+1}|W_{i+1}) \sum_{w_i} Pr(W_{i+1}|w_i) B(w_i)$$

The forward algorithm can be thought of as consisting of two distinctive steps: the **time elapse update** which corresponds to determining $B'(W_{i+1})$ from $B(W_i)$ and the **observation update** which corresponds to determining $B(W_{i+1})$ from $B'(W_{i+1})$. Hence, in order to advance our belief distribution by one timestep (i.e. compute $B(W_{i+1})$ from $B(W_i)$), we must first advance our model's state by one timestep with the time elapse update, then incorporate new evidence from that timestep with the observation update. Consider the following initial distribution, transition model, and sensor model:

W_0	$B(W_0)$	W_{i+1}	W_i	$Pr(W_{i+1} W_i)$	F_i	W_i	$Pr(F_i W_i)$
<i>sun</i>	0.8	<i>sun</i>	<i>sun</i>	0.6	<i>good</i>	<i>sun</i>	0.8
<i>rain</i>	0.2	<i>rain</i>	<i>sun</i>	0.4	<i>bad</i>	<i>sun</i>	0.2
		<i>sun</i>	<i>rain</i>	0.1	<i>good</i>	<i>rain</i>	0.3
		<i>rain</i>	<i>rain</i>	0.9	<i>bad</i>	<i>rain</i>	0.7

To compute $B(W_1)$, we begin by performing a time update to get $B'(W_1)$:

$$\begin{aligned}
B'(W_1 = \text{sun}) &= \sum_{w_0} Pr(W_1 = \text{sun}|w_0)B(w_0) \\
&= Pr(W_1 = \text{sun}|W_0 = \text{sun})B(W_0 = \text{sun}) + Pr(W_1 = \text{sun}|W_0 = \text{rain})B(W_0 = \text{rain}) \\
&= 0.6 \cdot 0.8 + 0.1 \cdot 0.2 = \boxed{0.5} \\
B'(W_1 = \text{rain}) &= \sum_{w_0} Pr(W_1 = \text{rain}|w_0)B(w_0) \\
&= Pr(W_1 = \text{rain}|W_0 = \text{sun})B(W_0 = \text{sun}) + Pr(W_1 = \text{rain}|W_0 = \text{rain})B(W_0 = \text{rain}) \\
&= 0.4 \cdot 0.8 + 0.9 \cdot 0.2 = \boxed{0.5}
\end{aligned}$$

Hence:

W_1	$B'(W_1)$
<i>sun</i>	0.5
<i>rain</i>	0.5

Next, we'll assume that the weather forecast for day 1 was good (i.e. $F_1 = \text{good}$), and perform an observation update to get $B(W_1)$:

$$\begin{aligned}
B(W_1 = \text{sun}) &\propto Pr(F_1 = \text{good}|W_1 = \text{sun})B'(W_1 = \text{sun}) = 0.8 \cdot 0.5 = \boxed{0.4} \\
B(W_1 = \text{rain}) &\propto Pr(F_1 = \text{good}|W_1 = \text{rain})B'(W_1 = \text{rain}) = 0.3 \cdot 0.5 = \boxed{0.15}
\end{aligned}$$

The last step is to normalize $B(W_1)$, noting that the entries in table for $B(W_1)$ sum to $0.4 + 0.15 = 0.55$:

$$\begin{aligned}
B(W_1 = \text{sun}) &= 0.4/0.55 = \frac{8}{11} \\
B(W_1 = \text{rain}) &= 0.15/0.55 = \frac{3}{11}
\end{aligned}$$

Our final table for $B(W_1)$ is thus the following:

W_1	$B(W_1)$
<i>sun</i>	8/11
<i>rain</i>	3/11

Note the result of observing the weather forecast. Because the weatherman predicted good weather, our belief that it would be sunny increased from $\frac{1}{2}$ after the time update to $\frac{8}{11}$ after the observation update.

As a parting note, the normalization trick discussed above can actually simplify computation significantly when working with Hidden Markov Models. If we began with some initial distribution and were interested in computing the belief distribution at time t , we could use the forward algorithm to iteratively compute $B(W_1), \dots, B(W_t)$ and normalize only once at the end by dividing each entry in the table for $B(W_t)$ by the sum of its entries.

Particle Filtering

Recall that with Bayes' nets, when running exact inference was too computationally expensive, using one of the sampling techniques we discussed was a viable alternative to efficiently approximate the desired probability distribution(s) we wanted. Hidden Markov Models have the same drawback - the time it takes to run exact inference with the forward algorithm scales with the number of values in the domains of the random variables. This was acceptable in our current weather problem formulation where the weather can only take on 2 values, $W_i \in \{sun, rain\}$, but say instead we wanted to run inference to compute the distribution of the actual temperature on a given day to the nearest tenth of a degree. The Hidden Markov Model analog to Bayes' net sampling is called **particle filtering**, and involves simulating the motion of a set of particles through a state graph to approximate the probability (belief) distribution of the random variable in question.

Instead of storing a full probability table mapping each state to its belief probability, we'll instead store a list of n particles, where each particle is in one of the d possible states in the domain of our time-dependent random variable. Typically, n is significantly smaller than d (denoted symbolically as $n \ll d$) but still large enough to yield meaningful approximations; otherwise the performance advantage of particle filtering becomes negligible. Our belief that a particle is in any given state at any given timestep is dependent entirely on the number of particles in that state at that timestep in our simulation. For example, say we indeed wanted to simulate the belief distribution of the temperature T on some day i and assume for simplicity that this temperature can only take on integer values in the range $[10, 20]$ ($d = 11$ possible states). Assume further that we have $n = 10$ particles, which take on the following values at timestep i of our simulation:

[15, 12, 12, 10, 18, 14, 12, 11, 11, 10]

By taking counts of each temperature that appears in our particle list and dividing by the total number of particles, we can generate our desired empirical distribution for the temperature at time i :

T_i	10	11	12	13	14	15	16	17	18	19	20
$B(T_i)$	0.2	0.2	0.3	0	0.1	0.1	0	0	0.1	0	0

Now that we've seen how to recover a belief distribution from a particle list, all that remains to be discussed is how to generate such a list for a timestep of our choosing.

Particle Filtering Simulation

Particle filtering simulation begins with particle initialization, which can be done quite flexibly - we can sample particles randomly, uniformly, or from some initial distribution. Once we've sampled an initial list of particles, the simulation takes on a similar form to the forward algorithm, with a time elapse update followed by an observation update at each timestep:

- *Time Elapse Update* - Update the value of each particle according to the transition model. For a particle in state t_i , sample the updated value from the probability distribution given by $Pr(T_{i+1}|t_i)$. Note the similarity of the time elapse update to prior sampling with Bayes' nets, since the frequency of particles in any given state reflects the transition probabilities.
- *Observation Update* - During the observation update for particle filtering, we use the sensor model $Pr(F_i|T_i)$ to weight each particle according to the probability dictated by the observed evidence and the particle's state. Specifically, for a particle in state t_i with sensor reading f_i , assign a weight of $Pr(f_i|t_i)$. The algorithm for the observation update is as follows:

1. Calculate the weights of all particles as described above.
2. Calculate the total weight for each state.
3. If the sum of all weights across all states is 0, reinitialize all particles.
4. Else, normalize the distribution of total weights over states and resample your list of particles from this distribution.

Note the similarity of the observation update to likelihood weighting, where we again downweight samples based on our evidence.

Let's see if we can understand this process slightly better by example. Define a transition model for our weather scenario using temperature as the time-dependent random variable as follows: for a particular temperature state, you can either stay in the same state or transition to a state one degree away, within the range $[10, 20]$. Out of the possible resultant states, the probability of transitioning to the one closest to 15 is 80% and the remaining resultant states uniformly split the remaining 20% probability amongst themselves.

Our temperature particle list was as follows:

[15, 12, 12, 10, 18, 14, 12, 11, 11, 10]

To perform a time elapse update for the first particle in this particle list, which is in state $T_i = 15$, we need the corresponding transition model:

T_{i+1}	14	15	16
$Pr(T_{i+1} T_i = 15)$	0.1	0.8	0.1

In practice, we allocate a different range of values for each value in the domain of T_{i+1} such that together the ranges entirely span the interval $[0, 1)$ without overlap. For the above transition model, the ranges are as follows:

1. The range for $T_{i+1} = 14$ is $0 \leq r < 0.1$.
2. The range for $T_{i+1} = 15$ is $0.1 \leq r < 0.9$.
3. The range for $T_{i+1} = 16$ is $0.9 \leq r < 1$.

In order to resample our particle in state $T_i = 15$, we generate a random number in the range $[0, 1)$ and see which range it falls in. Hence if our random number is $r = 0.467$, then the particle at $T_i = 15$ remains in $T_{i+1} = 15$ since $0.1 \leq r < 0.9$. Now consider the following list of 10 random numbers in the interval $[0, 1)$:

[0.467, 0.452, 0.583, 0.604, 0.748, 0.932, 0.609, 0.372, 0.402, 0.026]

If we use these 10 values as the random value for resampling our 10 particles, our new particle list after the full time elapse update should look like this:

[15, 13, 13, 11, 17, 15, 13, 12, 12, 10]

Verify this for yourself! The updated particle list gives rise to the corresponding updated belief distribution $B(T_{i+1})$:

T_i	10	11	12	13	14	15	16	17	18	19	20
$B(T_{i+1})$	0.1	0.1	0.2	0.3	0	0.2	0	0.1	0	0	0

Comparing our updated belief distribution $B(T_{i+1})$ to our initial belief distribution $B(T_i)$, we can see that as a general trend the particles tend to converge towards a temperature of $T = 15$.

Next, let's perform the observation update, assuming that our sensor model $Pr(F_i|T_i)$ states that the probability of a correct forecast $f_i = t_i$ is 80%, with a uniform 2% chance of the forecast predicting any of the other 10 states. Assuming a forecast of $F_{i+1} = 13$, the weights of our 10 particles are as follows:

Particle	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
State	15	13	13	11	17	15	13	12	12	10
Weight	0.02	0.8	0.8	0.02	0.02	0.02	0.8	0.02	0.02	0.02

Then we aggregate weights by state:

State	10	11	12	13	15	17
Weight	0.02	0.02	0.04	2.4	0.04	0.02

Summing the values of all weights yields a sum of 2.54, and we can normalize our table of weights to generate a probability distribution by dividing each entry by this sum:

State	10	11	12	13	15	17
Weight	0.02	0.02	0.04	2.4	0.04	0.02
Normalized Weight	0.0079	0.0079	0.0157	0.9449	0.0157	0.0079

The final step is to resample from this probability distribution, using the same technique we used to resample during the time elapse update. Let's say we generate 10 random numbers in the range $[0, 1)$ with the following values:

[0.315, 0.829, 0.304, 0.368, 0.459, 0.891, 0.282, 0.980, 0.898, 0.341]

This yields a resampled particle list as follows:

[13, 13, 13, 13, 13, 13, 13, 13, 15, 13, 13]

With the corresponding final new belief distribution:

T_i	10	11	12	13	14	15	16	17	18	19	20
$B(T_{i+1})$	0	0	0	0.9	0	0.1	0	0	0	0	0

Observe that our sensor model encodes that our weather prediction is very accurate with probability 80%, and that our new particles list is consistent with this since most particles are resampled to be $T_{i+1} = 13$.

Summary

In this note, we covered two new types of models:

- *Markov models*, which encode time-dependent random variables that possess the Markov property. We can compute a belief distribution at any timestep of our choice for a Markov model using probabilistic inference with the mini-forward algorithm.

- *Hidden Markov Models*, which are Markov models with the additional property that new evidence which can affect our belief distribution can be observed at each timestep. To compute the belief distribution at any given timestep with Hidden Markov Models, we use the forward algorithm.

Sometimes, running exact inference on these models can be too computationally expensive, in which case we can use particle filtering as a method of approximate inference.