



MSML610: Advanced Machine Learning

9.1: Reasoning Over Time

Instructor: Dr. GP Saggese - gsaggese@umd.edu

References: - AIMA 14: Probabilistic reasoning over time -
<https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python>

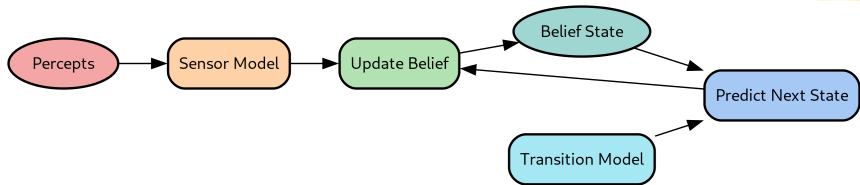
- 
- ***Reasoning Over Time***
 - Definitions
 - Defining Temporal Inference Tasks
 - Solving Temporal Inference Tasks

- Reasoning Over Time
 - *Definitions*
 - Defining Temporal Inference Tasks
 - Solving Temporal Inference Tasks

Static vs Dynamic Probabilistic Reasoning

- **Static probabilistic reasoning**
 - Random variables have *fixed* values
 - E.g., repairing a car:
 - Broken parts remain broken during diagnosis
 - Observed evidence stays fixed
- **Dynamic probabilistic reasoning**
 - Random variables *change over time*
 - E.g., tracking a plane's location, economic activity
 - E.g., treating a diabetic patient
 - Assess patient state, decide insulin dose
 - Evidence: insulin doses, food intake, blood sugar (change over time)
 - Time dependency (e.g., metabolic activity, time of day)

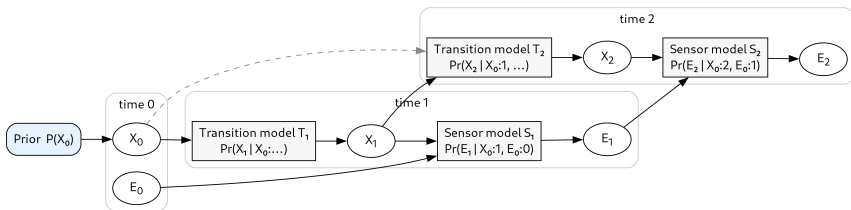
Agents in Partially Observable Environments



- Agents in *partially observable environments* track current state using:
 1. **Belief state**
 - Store possible world states
 - Use probability theory to quantify belief
 - Posterior distribution of current state given all evidence
 2. **Belief state + Transition model**
 - Predict world evolution in next step
 3. **Sensor model + Percepts**
 - Update belief state
- *Idea:* handle time by making each quantity a function of time
 - $\underline{X}_{a:b}$ represents variables in $[a, b]$

Agent: Model Components

1. **State of the world:** \underline{X}_t
 - Not directly observable
2. **Prior probability of the state** at time 0: \underline{X}_0
3. **Evidence variables:** \underline{E}_t
 - Observable
4. **Transition model:** $\Pr(\underline{X}_t | \underline{X}_{0:t-1})$
 - Models world evolution
 - Specifies probability distribution of state \underline{X}_t given previous values
5. **Sensor model:** $\Pr(\underline{E}_t | \underline{X}_{0:t}, \underline{E}_{0:t-1})$
 - Models generation of evidence variables \underline{E}_t



Discrete vs Continuous Time Models

- **Discrete time models**

- View world as time slices (“snapshots”)
 - Equal time intervals, equispaced samples
 - Label times $t = 0, 1, 2, \dots$
- Each slice contains random variables:
 - Hidden RVs (e.g., $\underline{\mathbf{X}}_t$)
 - Observable RVs (e.g., $\underline{\mathbf{E}}_t$)

- **Continuous time models**

- Model uncertainty over continuous time with stochastic differential equations (SDEs)
- Discrete time models approximate SDEs

Markov Property

- **In general**, current state $\underline{\mathbf{X}}_t$ depends on a growing number of past states:

$$\Pr(\underline{\mathbf{X}}_t | \text{history}) \triangleq \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{0:t-1}) = \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_0, \underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_{t-1})$$

- Of course, there can't be dependency from the future $\underline{\mathbf{X}}_{t+k}$ with $k > 0$
- **Markov property**: current state depends (conditionally) only on a finite fixed number of k previous states:

$$\begin{aligned}\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{0:t-1}) &= \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_0, \underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_{t-k-1}, \underline{\mathbf{X}}_{t-k}, \dots, \underline{\mathbf{X}}_{t-1}) \\ &= \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-k:t-1})\end{aligned}$$

Markov Process

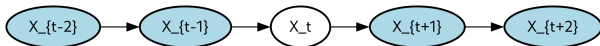
- **Markov processes** (aka Markov chains) have the Markov property

$$\Pr(\underline{\mathbf{X}}_t | \text{history}) = \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-k:t-1}) \quad \forall k, t$$

- **First-order Markov process**: current state $\underline{\mathbf{X}}_t$ depends only on previous state $\underline{\mathbf{X}}_{t-1}$:

$$\Pr(\underline{\mathbf{X}}_t | \text{history}) = \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-1}) \quad \forall k, t$$

- Next state depends only on previous state, not full history
- *Intuition*: system “forgets” everything except immediate last state
- Bayesian network for a first-order Markov process:



- E.g., probability of rain today depends only on yesterday, $\Pr(R_t | R_{t-1}) \quad \forall t$
- **Second-order Markov process**: current state $\underline{\mathbf{X}}_t$ depends only on $\underline{\mathbf{X}}_{t-1}$ and $\underline{\mathbf{X}}_{t-2}$

Time-Homogeneous Process

- Even with the Markov assumption, there is an infinite number of probability distributions, one for each t

$$\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-1:t-k})$$

- **Time-homogeneous** (aka stationarity): probability remains constant by translation over t

$$\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{0:t-1}) = \Pr(\underline{\mathbf{X}}_{t-k} | \underline{\mathbf{X}}_{0:t-k-1}) \forall k, t$$

- *Intuition*: even if process evolves, governing laws remain unchanged
- E.g., in the real-world, most physical laws are constant

First-Order Time-Homogeneous Process

- **First-order Markov property**

$$\Pr(\underline{\mathbf{X}}_t | \text{history}) = \Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-1})$$

- **Time-homogeneous**

$$\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{0:t-1}) = \Pr(\underline{\mathbf{X}}_{t-k} | \underline{\mathbf{X}}_{0:t-k-1}) \forall k, t$$

- **First-order time-homogeneous**: putting both properties together, one conditional probability table suffices:

$$\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{t-1}) = \Pr(\underline{\mathbf{X}}_{t-k} | \underline{\mathbf{X}}_{t-k-1}) \forall k, t$$

- E.g., rain probability for today depends only on yesterday and is constant:
 $\Pr(R_t | R_{t-1}) = f(R_{t-1}) \forall t$

Sensor Model

- **In general**, evidence variables \underline{E}_t depend on:
 - Previous state of the world $\underline{X}_{0:t}$
 - Previous sensor values $\underline{E}_{0:t-1}$

$$\Pr(\underline{E}_t | \underline{X}_{0:t}, \underline{E}_{0:t-1})$$

- **Sensor Markov property**

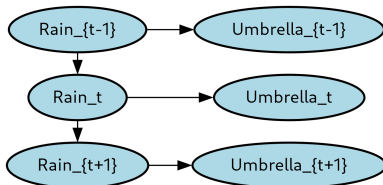
- Assume sensor value \underline{E}_t depends only on current state \underline{X}_t , not on previous sensor values

$$\Pr(\underline{E}_t | \underline{X}_{0:t}, \underline{E}_{0:t-1}) = \Pr(\underline{E}_t | \underline{X}_t)$$

- In a Bayesian network, even if \underline{X}_t and \underline{E}_t are contemporaneous, the arrow goes from $\underline{X}_t \rightarrow \underline{E}_t$ since the world causes the sensor to take on particular values

Sensor Model: Rain Example

- Consider the Bayesian network for the **rain model**
- The world causes the sensor to take specific values $\underline{X}_t \rightarrow \underline{E}_t$
 - E.g., $Rain_t \rightarrow Umbrella_t$, since rain “causes” the umbrella to appear
 - Inference goes the other direction: “see the umbrella, guess if it’s raining”
- The **transition model** is $\Pr(Rain_t | Rain_{t-1})$
 - $\Pr(R_t | R_{t-1} = T) = 0.7$
 - $\Pr(R_t | R_{t-1} = F) = 0.4$
 - The sum doesn’t have to be 1 since it’s a conditional probability
- The **sensor model** is $\Pr(Umbrella_t | Rain_t)$
 - $\Pr(U_t | R_t = T) = 0.9$ (people forget the umbrella)
 - $\Pr(U_t | R_t = F) = 0.2$ (people are paranoid)



Prior Probability

- Complete system specification needs **prior probability of state variables** at initial time, $\Pr(\underline{\mathbf{X}}_0)$
 - Represents initial belief about system state before observations
 - Crucial for initializing state estimation process
- E.g.,
 - $\underline{\mathbf{X}}_0$ represents position and velocity of a moving object
 - $\Pr(\underline{\mathbf{X}}_0)$ could be a Gaussian distribution centered around an initial guess of object's position and velocity with uncertainty

First-Order Markov Process: Joint Distribution

- **Goal:** model a sequence of states $\underline{\mathbf{X}}_0, \underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_t$ and observations $\underline{\mathbf{E}}_1, \dots, \underline{\mathbf{E}}_t$ over time, i.e., $\Pr(\underline{\mathbf{X}}_{0:t}, \underline{\mathbf{E}}_{1:t})$
- **Express the joint distribution** of n random variables using the chain rule:

$$\Pr(\underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_n) = \prod_{i=1}^n \Pr(\underline{\mathbf{X}}_i | \underline{\mathbf{X}}_{0:i-1})$$

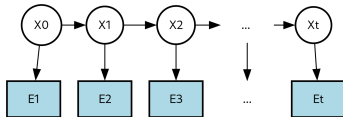
- Like for Bayesian networks **factorize joint distribution** according to graph dependencies:

$$\Pr(\underline{\mathbf{X}}_1, \dots, \underline{\mathbf{X}}_n) = \prod_{i=1}^n \Pr(\underline{\mathbf{X}}_i | \text{parents}(\underline{\mathbf{X}}_i))$$

- **First-order Markov assumption:**

$$\Pr(\underline{\mathbf{X}}_i | \underline{\mathbf{X}}_{0:i-1}) = \Pr(\underline{\mathbf{X}}_i | \underline{\mathbf{X}}_{i-1})$$

- **First-order Markov sensor model:**



$$\Pr(\underline{\mathbf{E}}_i | \underline{\mathbf{X}}_{0:i}, \underline{\mathbf{E}}_{1:i-1}) = \Pr(\underline{\mathbf{E}}_i | \underline{\mathbf{X}}_i)$$

First-Order Markov Process: Joint Distribution

- Putting everything together, the joint distribution probability for a time-homogeneous first-order Markov process:

$$\begin{aligned}\Pr(\underline{\mathbf{X}}_{0:t}, \underline{\mathbf{E}}_{1:t}) &= \Pr(\underline{\mathbf{X}}_0) \prod_{i=1}^t \Pr(\underline{\mathbf{X}}_i | \underline{\mathbf{X}}_{i-1}) \Pr(\underline{\mathbf{E}}_i | \underline{\mathbf{X}}_i) \\ &= \text{prior} \times \prod_i \text{transition model} \times \text{sensor model}\end{aligned}$$

- Remarks:**
 - The state evolves probabilistically from the previous state (transition model)
 - This structure reduces complexity and enables tractable inference
 - A Bayesian network can represent a temporal model by modeling time with indices t , i.e., “unrolling the model”
- Problem:** infinite t , even assuming the Markov property

Improving Approximation of Real-World Systems

- Is first-order Markov process a **reasonable approximation of reality**?
 - Particle following random walk is well represented by Markov process
 - In umbrella example, rain depends only on previous day
- **How to improve the approximation**
 1. *Increase order of Markov process model*
 - Model “rarely rains more than two days in a row” with second-order Markov model $\Pr(Rain_t | Rain_{t-1}, Rain_{t-2})$
 2. *Increase number of state variables*
 - Add $Season_t$ to incorporate historical records
 - Transition model becomes more complicated
 3. *Increase number of sensor variables*
 - Add $Location_t, Temperature_t, Humidity_t, Pressure_t$
 - Simplifies modeling of state

- Reasoning Over Time
 - Definitions
 - *Defining Temporal Inference Tasks*
 - Solving Temporal Inference Tasks

Inference Tasks in Temporal Models

- There are **several tasks** in temporal inference

Task	Description	Estimate
Filtering	Estimate <i>current</i> state given past / current obs	$\Pr(\underline{X}_t \underline{E}_{1:t})$
Prediction	Estimate <i>future</i> state given past / current obs	$\Pr(\underline{X}_{t+k} \underline{E}_{1:t})$ for $k > 0$
Smoothing	Estimate <i>past</i> state given past, current, and <i>future</i> obs	$\Pr(\underline{X}_k \underline{E}_{1:T})$ for $T < k$
Most likely explanation	Find most probable sequence of states given the evidence	$\operatorname{argmax}_{\underline{x}_{1:T}} \Pr(\underline{X}_{1:T} \underline{E}_{1:t})$
Learning	Learn model parameters or structure from data	θ of a model

Task 1: Filtering

- **Filtering** (aka “state estimation”) computes the posterior distribution of the *current state* given *all evidence to date*:

$$\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{E}}_{1:t} = \underline{\mathbf{e}}_{1:t})$$

- Estimate probability of rain today, given all umbrella observations so far $\Pr(\text{Rain}_t | \text{Umbrella}_{1:t})$
- **Application**
 - Filtering needed by a rational agent to *track current state*
 - Agent believes current state $\Pr(\underline{\mathbf{X}}_{t-1})$ at time $t - 1$
 - New evidence $\underline{\mathbf{e}}_t$ arrives for time t
 - Agent updates belief about current state $\Pr(\underline{\mathbf{X}}_t)$ at time t
- “*Filtering*” refers to filtering out noise in a signal by estimating system parameters

Task 2: Prediction

- **Prediction** involves predicting the posterior distribution over a *future state*, given *all evidence to date*:

$$\Pr(\underline{X}_{t+k} | \underline{E}_{1:t} = \underline{e}_{1:t}) \text{ with } k > 0$$

- E.g., compute the probability of rain three days from now:

$$\Pr(Rain_{t+3} | Umbrella_{1:t})$$

- **Application**
 - Prediction helps rational agents evaluate actions based on expected outcomes

Task 3: Smoothing

- **Smoothing** compute posterior distribution over a *past state* given *all past, present, and future evidence*:

$$\Pr(\underline{X}_k | \underline{E}_{1:t} = \underline{e}_{1:t}) \text{ with } 0 \leq k < t$$

- **Important**: you have information about the “future” of the evidence, but not the state
- **Application**
 - Smoothing provides a better state estimate by incorporating more future evidence
 - E.g., compute the probability it rained last Wednesday, given all observations up to today
- The term “*smoothing*” refers to the state estimate being smoother than filtering

Task 4: Most-Likely Explanation

- **Most-likely explanation** finds the sequence of states $\underline{X}_{1:t}$ most likely to have generated observations $\underline{E}_{1:t}$:

$$\operatorname{argmax}_{\underline{X}_{1:t}} \Pr(\underline{X}_{1:t} | \underline{E}_{1:t})$$

- E.g.,
 - Umbrella appeared on 3 days, not on the fourth
 - Most likely explanation: rained for 3 days, then stopped
- **Applications**
 - Speech recognition: most likely sequence of words given sounds
 - Digital processing: reconstruct bit strings over a noisy channel

Task 5: Learning

- **Learning** involves estimating the transition model $\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{X}}_{0:t-1})$ and the sensor model $\Pr(\underline{\mathbf{E}}_i | \underline{\mathbf{X}}_i)$ from observations
- Learning benefits from smoothing rather than filtering for better state estimates
 - Smoothing uses all data to estimate states, leading to more accurate models
 - E.g., in weather prediction, smoothing uses past, present, and future data to better estimate current weather state

- Reasoning Over Time
 - Definitions
 - Defining Temporal Inference Tasks
 - *Solving Temporal Inference Tasks*

Solving Task 1: Filtering

- **Filtering** computes the posterior distribution of the *current state* given *all evidence to date*, i.e., $\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{E}}_{1:t} = \underline{\mathbf{e}}_{1:t})$
- A practical filtering algorithm updates the current state estimate $\underline{\mathbf{X}}_{t+1}$ using the previous state $\underline{\mathbf{X}}_t$ and the new evidence $\underline{\mathbf{e}}_{t+1}$
 - Instead of recomputing each state by going over the entire history of the percepts
 - Aka “recursive state estimation”

$$\Pr(\underline{\mathbf{X}}_{t+1} | \underline{\mathbf{e}}_{1:t+1}) = f(\Pr(\underline{\mathbf{X}}_t | \underline{\mathbf{e}}_{1:t}), \underline{\mathbf{e}}_{t+1})$$
$$\text{NextState} = f(\text{PreviousState}, \underline{\mathbf{e}}_{t+1})$$

- **Why?**
 - Time and space requirements for updating must be constant for a (finite) agent to keep track of current state indefinitely
- **Is it possible?**
 - What is the formula $f(\dots)$?

Recursive Filtering: Update Formula

- Compute the state at time $t + 1$ with all the evidence up to that time
- Assume that state and evidence are scalar and not vector

$$\begin{aligned} & \Pr(X_{t+1} | e_{1:t+1}) \\ &= \Pr(X_{t+1} | \mathbf{e}_{1:t}, \mathbf{e}_{t+1}) && \text{Divide up the evidence} \\ &= \alpha \Pr(e_{t+1} | X_{t+1}, \mathbf{e}_{1:t}) \Pr(X_{t+1} | \mathbf{e}_{1:t}) && \text{Bayes rule given} \\ &= \alpha \Pr(e_{t+1} | X_{t+1}) \Pr(X_{t+1} | \mathbf{e}_{1:t}) && \text{Markov sensor assumption} \\ &= \alpha \Pr(e_{t+1} | X_{t+1}) \sum_{x_t} \Pr(X_{t+1} | x_t, \mathbf{e}_{1:t}) \Pr(x_t | \mathbf{e}_{1:t}) && \text{Condition on current state} \\ &= \alpha \Pr(e_{t+1} | X_{t+1}) \sum_{x_t} \Pr(X_{t+1} | x_t) \Pr(x_t | \mathbf{e}_{1:t}) && \text{Markov assumption} \end{aligned}$$

- It has the expected form:

$$\Pr(X_{t+1} | e_{1:t+1}) = f(\Pr(X_t | e_{1:t}), e_{t+1})$$

Recursive Filtering: Update Formula

- The update formula for the state is:

$$\Pr(X_{t+1}|e_{1:t+1}) = \alpha \Pr(e_{t+1}|X_{t+1}) \sum_{x_t} \Pr(X_{t+1}|x_t) \Pr(x_t|e_{1:t})$$

- The next state is “Sensor model x Transition model x Recursive state”
 - Sensor model: $\Pr(e_{t+1}|X_{t+1})$
 - Transition model: $\Pr(X_{t+1}|x_t)$
 - Recursive term: $\Pr(x_t|e_{1:t})$

Recursive Filtering: Intuition

- Recursive state estimation updates the state belief as new evidence arrives

$$\Pr(X_{t+1}|e_{1:t+1}) = \alpha \Pr(e_{t+1}|X_{t+1}) \sum_{x_t} \Pr(X_{t+1}|x_t) \Pr(x_t|e_{1:t})$$

in **two steps**

- Prediction step:** Use the transition model to predict the next state based on the current belief

$$\Pr(X_{t+1}|e_{1:t}) = \sum_{x_t} \Pr(X_{t+1}|x_t) \Pr(x_t|e_{1:t})$$

- Intuition: Project the current belief forward using the model of system evolution

- Update step:** Incorporate the new observation to refine the prediction

$$\Pr(X_{t+1}|e_{1:t+1}) = \alpha \Pr(e_{t+1}|X_{t+1}) \Pr(X_{t+1}|e_{1:t})$$

- Intuition: Correct the prediction using the likelihood of the new evidence
- Maintain $\Pr(X_t|e_{1:t})$, the probability of the current state given all past evidence

- E.g., in a weather model, if it was likely to rain today and rain usually continues, the prediction leans toward rain tomorrow

- Seeing an umbrella supports this and updates the belief accordingly

Forward update

- We achieved:

$$\begin{aligned}\Pr(\underline{\mathbf{X}}_{t+1}|\underline{\mathbf{e}}_{1:t+1}) &= \alpha \Pr(\underline{\mathbf{e}}_{t+1}|\underline{\mathbf{X}}_{t+1}) \sum_{\mathbf{x}_t} \Pr(\underline{\mathbf{X}}_{t+1}|\mathbf{x}_t) \Pr(\mathbf{x}_t|\underline{\mathbf{e}}_{1:t}) \\ &= f(\Pr(\underline{\mathbf{X}}_t|\underline{\mathbf{e}}_{1:t}), \underline{\mathbf{e}}_{t+1})\end{aligned}$$

- The filtered estimate $\underline{\mathbf{f}}_{1:t} = \Pr(\underline{\mathbf{X}}_t|\underline{\mathbf{e}}_{1:t})$ is propagated forward and updated by each transition and new observation

$$\underline{\mathbf{f}}_{1:t+1} = \text{Forward}(\underline{\mathbf{f}}_{1:t}, \underline{\mathbf{e}}_{t+1})$$

starting with the initial condition $\underline{\mathbf{f}}_{1:0} = \Pr(\underline{\mathbf{X}}_0)$

- This is called “forward update”
- This process allows efficient online inference without storing the full history
 - Time and space requirements for updating is constant
- A (finite) agent can keep track of current state indefinitely

Solving Task 2: Prediction

- Prediction is equivalent to filtering without updating the state with new evidence, since there is no evidence
 - Only the transition model is needed, not the sensor model
- The rule predicting state $\underline{\mathbf{X}}_{t+k+1}$ given state $\underline{\mathbf{X}}_{t+k}$ and evidence $\underline{\mathbf{E}}_{1:t}$ is:

$$\Pr(\underline{\mathbf{X}}_{t+k+1} | \underline{\mathbf{e}}_{1:t}) = \sum_{\underline{\mathbf{x}}_{t+k}} \Pr(\underline{\mathbf{X}}_{t+k+1} | \underline{\mathbf{x}}_{t+k}) \Pr(\underline{\mathbf{x}}_{t+k} | \underline{\mathbf{e}}_{1:t})$$

- This equation can be used recursively to advance over time
 - Predicting even a few steps ahead generally incurs large uncertainty

Solving Task 3: Smoothing

- You want to calculate the probability distribution over the hidden state at time k , given all evidence up to time t (in the future!)

$$\Pr(X_k | e_{1:t}) \text{ where } 0 \leq k < t$$

- Filtering gives $\Pr(X_k | e_{1:k})$ using past and present evidence
- Smoothing refines the estimate of past states using later evidence
- **Example**
 - You're tracking whether it was raining yesterday
 - You had some evidence up to yesterday (e.g., a cloudy sky)
 - Today you see puddles on the ground
 - That new observation supports the idea that yesterday was raining

Task 3: Smoothing: Update Formula

- Using the same math as for filtering and the two key assumptions of Markov process and Markov sensor

- **Forward Pass (aka filtering):**

- Move forward through time, using the filtering algorithm to compute:

$$f_{1:k} = \Pr(X_k | e_{1:k})$$

- This gives you a “best guess” of the state at time k , based only on evidence up to k

- **Backward Pass (aka smoothing):**

- Move backward through time from time t , computing:

$$b_{k+1:t} = \Pr(e_{k+1:t} | X_k)$$

- This captures how likely the future evidence is, given a particular value of X_k

- **Combine them:**

- Multiply forward and backward messages to get:

$$\Pr(X_k | e_{1:t}) \propto f_{1:k} \times b_{k+1:t}$$

Task 4: Most Likely Explanation: Intuition 1/2

- You are tracking the weather (sunny or rainy) based on whether someone carries an umbrella
 - You can't see *Weather* directly (hidden state), but you observe umbrellas (which is a noisy observation)
 - You have 5 observations $Umbrella = [T, T, F, T, T]$
- **Question:** what is the most likely sequence of *Weather* states that explains the *Umbrella* observations?
 - You know something about:
 - the transition model (i.e., "it tends to rain several days in a row")
 - the sensor model (i.e., "people often forget the umbrella")
- Mathematically:

$$\operatorname{argmax}_{x_{1:t}} \Pr(x_{1:t} | e_{1:t}) = \operatorname{argmax}_{Weather_{1:t}} \Pr(Weather_{1:t} | Umbrella_{1:t})$$

Task 4: Most Likely Explanation: Intuition 2/2

- **Naive approach:** Use smoothing to choose the most likely state at each time step
 - Cons
 - Might lead to an implausible overall path
 - Suboptimal since the question addresses joint probability and we are not using all the information (only one step at the time!)
- **Viterbi algorithm:**
 - Constructs a path through a state-time graph with states as nodes and transitions as edges
 - Finds the most likely entire path through the hidden states
- **Key difference:**
 - In speech recognition, find the most likely word sequence behind a noisy audio signal
 - Smoothing: Best guess per time step (may miss non-English words or suboptimal sequence)
 - Viterbi: Best overall path (maximizes joint probability of the entire sequence)

Viterbi Algorithm: Intuition

- **Goal:** Find the most likely sequence of hidden states given observations

1. Initialization

- At $t = 1$, estimate probability of starting in each state using initial state distribution and observation likelihood

2. Recursion via dynamic programming

- For each $t > 1$, for each state x_t :
- Compute maximum probability path to x_t from any previous state
- Use:
 - $\Pr(x_t|x_{t-1})$: transition model
 - $\Pr(e_t|x_t)$: sensor model
 - Best path probability to x_{t-1} from prior step
- Store probability and corresponding back-pointer to x_{t-1}

3. Termination and backtrace

- At final time $t = T$, identify state with highest final probability
- Trace back through stored pointers to reconstruct optimal path

Viterbi Algorithm: Example 1/2

- You observe a friend carrying an umbrella over 3 days
 - *Umbrella* = [Yes, Yes, No]
- You want to infer the most likely sequence of hidden *Weather* states
 - States: $S = \{\text{Sunny, Rainy}\}$ (weather)
 - Observations: $O = \{\text{Yes, No}\}$ (umbrella)
 - Initial Probabilities:

$$\Pr(\text{Sunny}) = 0.6, \quad \Pr(\text{Rainy}) = 0.4$$

- Transition Probabilities:

$$\Pr(\text{Sunny} \rightarrow \text{Sunny}) = 0.7, \quad \Pr(\text{Sunny} \rightarrow \text{Rainy}) = 0.3$$

$$\Pr(\text{Rainy} \rightarrow \text{Sunny}) = 0.4, \quad \Pr(\text{Rainy} \rightarrow \text{Rainy}) = 0.6$$

- Observation (Emission) Probabilities:

$$\Pr(\text{Yes}|\text{Sunny}) = 0.1, \quad \Pr(\text{No}|\text{Sunny}) = 0.9$$

$$\Pr(\text{Yes}|\text{Rainy}) = 0.8, \quad \Pr(\text{No}|\text{Rainy}) = 0.2$$

Viterbi Algorithm: Example 2/2

- Viterbi table

Day	State	Probability	Backpointer
1	Sunny	$0.6 \times 0.1 = \mathbf{0.06}$	—
	Rainy	$0.4 \times 0.8 = \mathbf{0.32}$	—
2	Sunny	$\max(0.06 \times 0.7, 0.32 \times 0.4) \times 0.1 = \mathbf{0.0128}$	Rainy
	Rainy	$\max(0.06 \times 0.3, 0.32 \times 0.6) \times 0.8 = \mathbf{0.1536}$	Rainy
3	Sunny	$\max(0.0128 \times 0.7, 0.1536 \times 0.4) \times 0.9 = \mathbf{0.0553}$	Rainy
	Rainy	$\max(0.0128 \times 0.3, 0.1536 \times 0.6) \times 0.2 = \mathbf{0.0184}$	Rainy

- Final most probable state
 - Sunny (Day 3)
- Find the most likely sequence
 - Rainy \rightarrow Rainy \rightarrow Sunny

