PROBABILISTIC MODELS – PART 6: TEMPORAL MODELS AND DYNAMIC BAYESIAN NETWORKS

Gerhard Widmer

Institute of Computational Perception
Johannes Kepler University
Linz, Austria

gerhard.widmer@jku.at
www.cp.jku.at/people/widmer

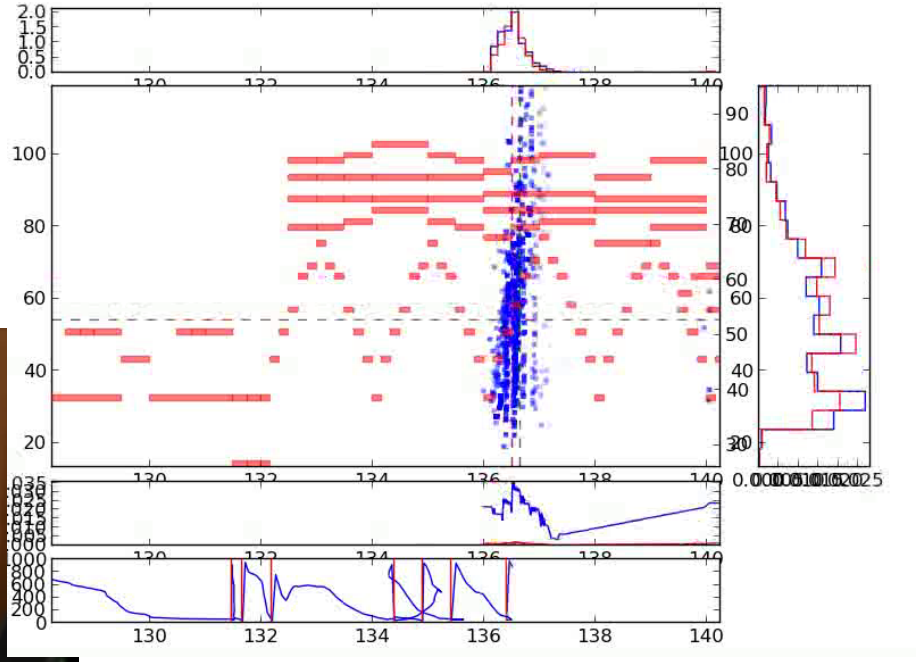
JOHANNES KEPLER UNIVERSITY LINZ

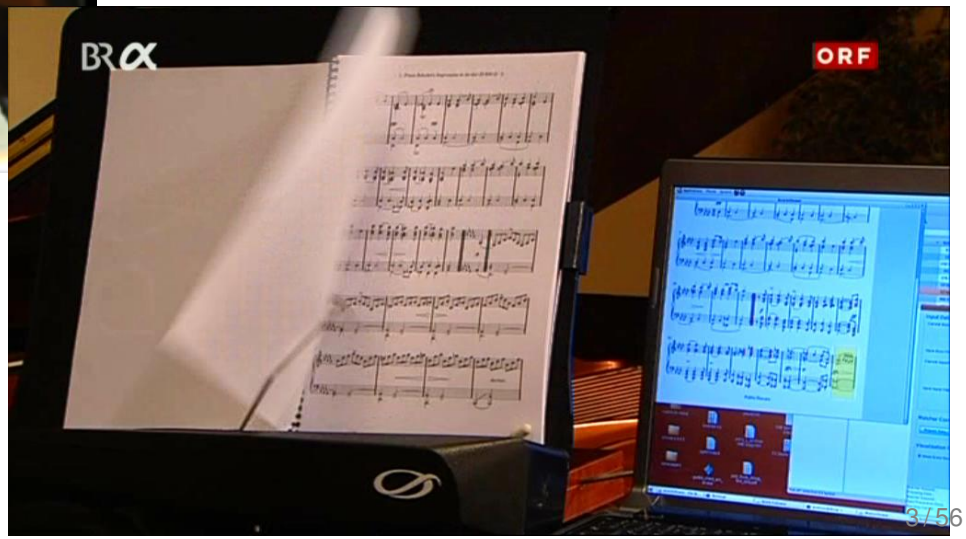
Institute of Computational Perception

November 17, 2025

Presentation partly based on and inspired by [Koller & Friedman, 2009] and [Russell & Norvig, 2021], including the use of some figures from their books and/or lecture slides.

Many thanks to Daphne Koller, Nir Friedman, Stuart Russell, and Peter Norvig for making these available (`pgm.stanford.edu`; `aima.cs.berkeley.edu`).


2.0
1.5
1.0
0.5
0.0
100
80
60
40
20
90
100
80
70
60
50
40
30
130
132
134
136
138
140


BR
ORF

Goals of this Lecture

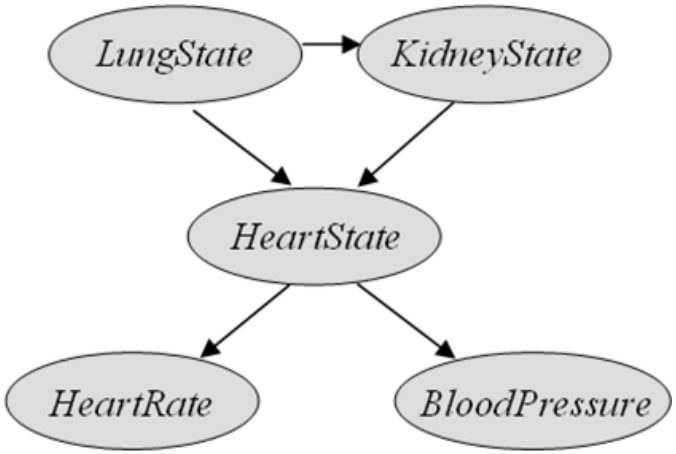
- Discuss how to model processes that develop over time
- Introduce simplifying assumptions to make modelling feasible
- Introduce the general class of **Dynamic Bayesian Networks (DBNs)**
- Discuss intractability of exact inference in unrestricted DBNs
- Introduce a simplified class of models: **State-Observation Models**
- Describe different inference tasks in S-O Models
- Derive general algorithms for 3 general inference tasks
- Explain why we need to restrict ourselves to even simpler models (☞ next sections)

Outline

Motivating Example

Consider monitoring a patient in an intensive care unit

- Obtain sensor readings (heart rate, blood pressure, respiration, ECG, ...) via some measurement sensors
- Goal: reason about patient's (hidden) internal state: state of heart, lungs, ...
- Model joint distribution over these variables as a Bayesian Network:



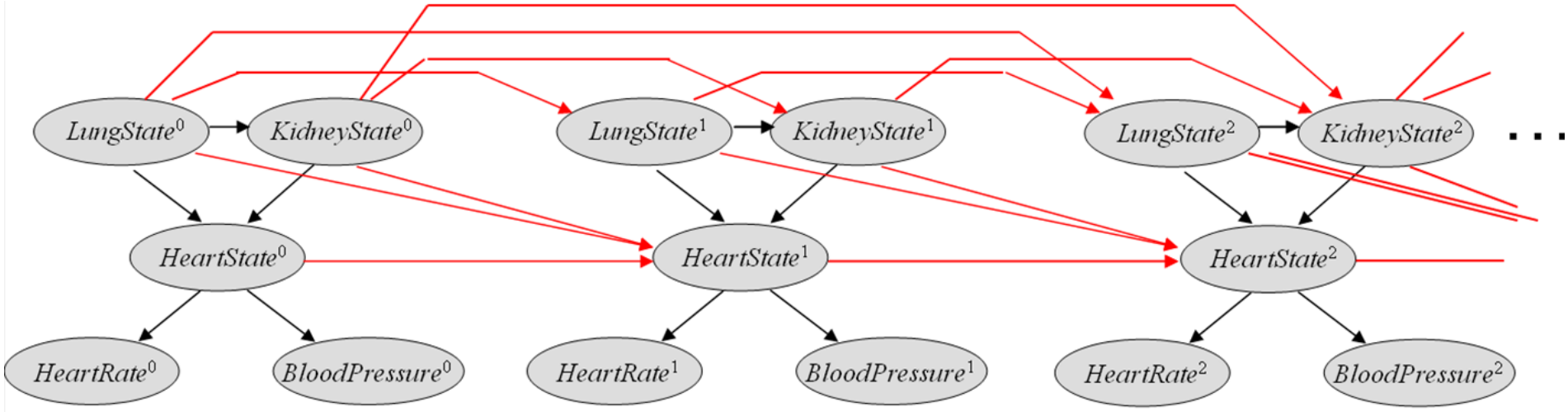
Given measurements of heart rate, blood pressure etc., this network permits us to infer the most likely state of the interior organs ... but:

- This only models a specific moment.
- A patient's state is not static!

More Realistic Setting

System "patient + intensive care unit" is a dynamic system whose state changes over time, and whose state we wish to track and reason about:

- Variables may have different values at different time points
- Need separate new instantiation of each variable for each time point
- State at time t may depend on previous states
- Dependencies between variables at different time points!



☞ **Need to reason about a huge network of arbitrary length ...**

Dynamic Bayesian Networks (DBNs)

- are compact ("template-based") models of temporal processes that
- capture a system's dynamics (development over time)
- permit us to model distributions over **sequences** of system states ("trajectories") of arbitrary length.

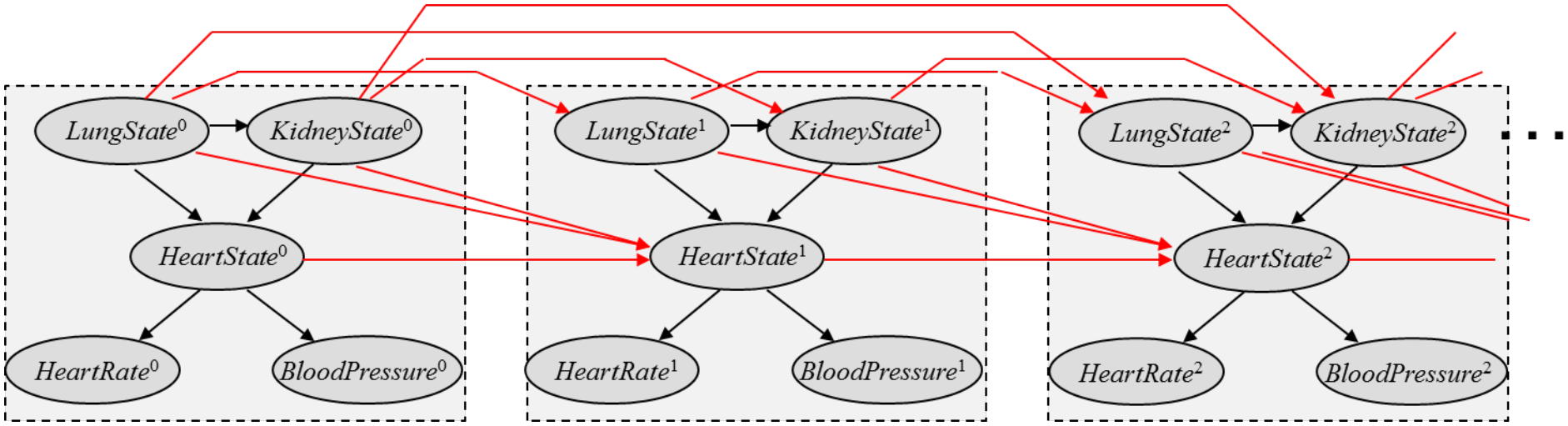
Modelling approach:

- Define a set of variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ to describe system's state
- Model system state **at time point** t as a Bayesian Network
- Each time point t has its own network over $\mathcal{X}$
- Same network structure for each time point t
- Each variable X_i has a specific **instantiation** $X_i^{(t)}$ for each t
- X_i (the "mother" of all variables $X_i^{(t)}$) is called **template variable**
- Temporal dependencies are modelled by edges between networks at different time points t

Notation:

- For a set of variables $\boldsymbol{X} \subseteq \mathcal{X}$, will write $\boldsymbol{X}^{(t_1:t_2)} (t_1 < t_2)$ to denote the set of variables $\{\boldsymbol{X}^{(t)} : t \in [t_1, t_2]\}$
- $\boldsymbol{x}^{(t_1:t_2)}$ is an assignment of values to this set of variables.

Dynamic Bayesian Networks (DBNs)



The Complexity Problem:

- A "possible world" (atomic event) in our probability space is now a **trajectory** = an assignment of values to all variables $\mathcal{X}^{(1:T)}$ for some duration T
- Network represents a **joint distribution over such trajectories** (i.e., over all possible sequences of system states)
- Trajectories can become arbitrarily long
- Huge probability space!
- Simplifying assumptions needed to make this tractable ...

Simplifying Assumption 1: Discrete Time

Assumption 1: Discrete Time

- Timeline is discretised into *time slices*
- System state is taken at regularly spaced intervals with step size Δ
- $\mathcal{X}^{(0)}, \mathcal{X}^{(1)}, \ldots \mathcal{X}^{(t)}$ are the variables that represent the system's state and history at time $t \cdot \Delta$

Consequences:

- **Finite** (though large) set of random variables
- Can write distribution over trajectories in a compact form (by the chain rule):

$$P(\mathcal{X}^{(0:T)}) = P(\mathcal{X}^{(0)}, \mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(T)}) = P(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(0:t)})$$

- Distribution over trajectories is a product of conditional distributions over the variables in each time slice, given *all the preceding* time slices

Simplifying Assumption 2: The Markov Assumption

Assumption 2: The Markov Property

- *"The future is conditionally independent of the past, given the present."*

Definition

We say that a dynamic system over template variables $\mathcal{X}$ satisfies the **Markov Assumption**[a] if, for all $t \geq 0$,

$$(\mathcal{X}^{(t+1)} \perp \mathcal{X}^{(0:t-1)} \mid \mathcal{X}^{(t)})$$

Such a system is called a (first-order) **Markovian system**.

[a] named after Andrei Andreyevich Markov, Russian mathematician, 1856-1922.

Notes:

- this is a *very* strong and limiting assumption
- not satisfied in many practical applications (e.g., ICU)
- but needed to make the problem tractable

☞ Be aware of this!

Simplifying Assumption 2: The Markov Assumption

In Words:

- In a Markovian system, the system state at time $t+1$ only depends on the state at time t, and not on any information from earlier states:

$$P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(0:t)}) = P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$$

- In a Markovian graphical model, there are no arrows into $\mathcal{X}^{(t+1)}$ from variables in time slices $t-1$ or earlier.

Consequence:

Definition

A Markovian Network Model represents the following **distribution over state trajectories:**

$$\boxed{P(\mathcal{X}^{(0:T)})} = P(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(0:t)})$$

$$= \boxed{P(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})}$$

Simplifying Assumption 3: Stationarity

Assumption 3: Stationarity

- The laws governing the system's behaviour do not change over time
- They are the same in each time step
- The system has "**stationary dynamics**".

Definition

We say that a Markovian dynamic system is **stationary**[a] (or **time invariant**) if $P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$ is the same for all t.

[a] Not to be confused with a *static* system!

Consequence:

- The CPD tables look the same in all time steps
- Can represent the whole process (system over time) using a *single* **transition model** $P(\mathcal{X}' \mid \mathcal{X}) = P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$ for all $t \geq 0$

Simplifying Assumption 3: Stationarity

Remember:

- A Markovian Network Model represents the following distribution:

$$P(\mathcal{X}^{(0:T)}) \quad = \quad P(\mathcal{X}^{(0)}) \prod_{t=0}^{T-1} P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$$

Consequence:

Full Markovian DBN with stationary dynamics can be characterised by

1. the **Initial State Distribution** $P(\mathcal{X}^{(0)})$
   (= a Bayesian Network over variables $\mathcal{X}^{(0)}$) and
2. the **Transition Model** $P(\mathcal{X}' \mid \mathcal{X})$:
   = a Bayesian Network over variables $\mathcal{X}' \cup \mathcal{X} = \mathcal{X}^{(t+1)} \cup \mathcal{X}^{(t)}$
   with CPDs for the $\mathcal{X}'$ with parents in $\mathcal{X}'$ and possibly in $\mathcal{X}$

Example Problem: Vehicle Localisation and Tracking

Consider a Vehicle Tracking Task:

- A moving car tries to track its current position and velocity using data from a (possibly faulty) sensor (e.g., a GPS receiver).

Model the world state at any time t using 5 variables:

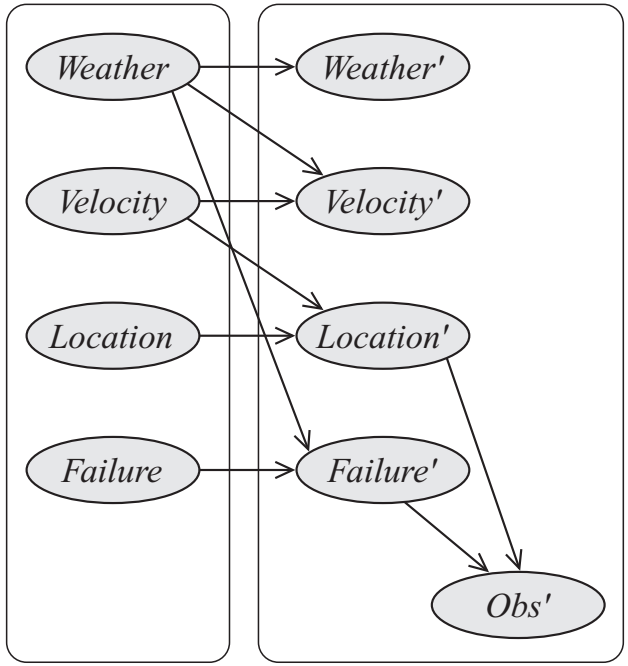
$Location^{(t)}$	the car's current location
$Velocity^{(t)}$	the car's current velocity
$Weather^{(t)}$	the current weather
$Failure^{(t)}$	the failure status of the sensor ($working/defect$)
$Obs^{(t)}$	the current observation (e.g., GPS coordinates)

- Have one such set of variables for each time point t
- Joint distribution over all these sets defines a *probability distribution over trajectories and observations* of the car.

Example Problem: Vehicle Localisation and Tracking

Some Modelling Assumptions for the Transition Model:

- Current observation (GPS) depends on (unobserved) true location of the car, and on error status of the sensor (GPS receiver)
- Bad weather increases chances of sensor to fail
- True car location depends on previous position and (previous) velocity
- The weather at any time is correlated with the weather at the previous time point (e.g., has a certain tendency to stay the same)

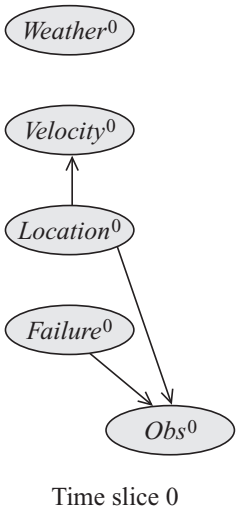


Dynamic Bayesian Networks

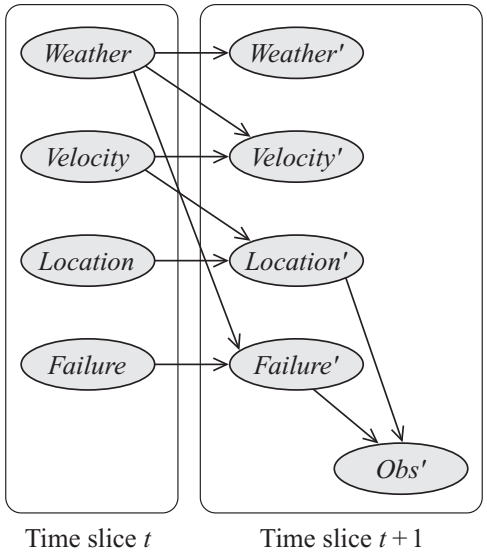
Definition

A **DYNAMIC BAYESIAN NETWORK (DBN)** is a pair $\langle \mathcal{B}_0, \mathcal{B}_\rightarrow \rangle$, where

- $\mathcal{B}_0$ is a Bayesian network over $\mathcal{X}^{(0)}$, representing the **distribution over initial states** $P(\mathcal{X}^{(0)})$
- $\mathcal{B}_\rightarrow$ is a **two-timeslice network** that describes the system dynamics, i.e., the **transition model** $P(\mathcal{X}' \mid \mathcal{X})$



Initial State Model $\mathcal{B}_0$



Transition Model $\mathcal{B}_\rightarrow$ *(structure only)*

Unrolling a DBN

Definition

Given a Dynamic Bayesian Network (DBN) $\langle \mathcal{B}_0, \mathcal{B}_{\rightarrow} \rangle$, for any desired time span $T \geq 0$, the **probability distribution** over the trajectories $\mathcal{X}^{(0:T)}$ is defined by an **Unrolled Bayesian Network** where, for any $i = 1, \ldots, n$:

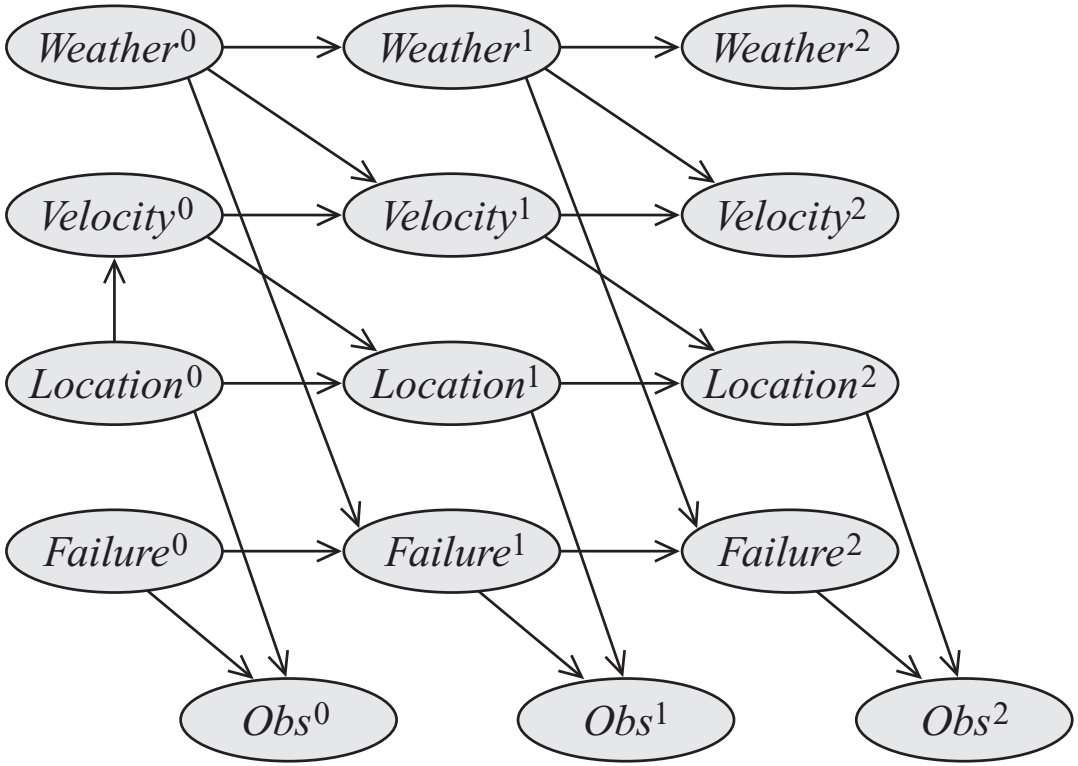
- the dependencies and CPD of $X_i^{(0)}$ are as defined in $\mathcal{B}_0$, and
- the dependencies and CPD of $X_i^{(t)}$, for $t > 0$, are as defined in $\mathcal{B}_{\rightarrow}$

Notes:

- All instantiations of a variable $X_i^{(t)}$ for $t > 0$ have the same dependency structure and the same CPD
- A DBN is a compact representation from which we can generate an infinite set of Bayesian networks (one for every $T > 0$) by **unrolling** the network

Example Problem: Vehicle Localisation and Tracking

Network unrolled over 3 time steps:



Inference in Dynamic Bayesian Networks

Remember the general inference task in Bayesian networks:

Compute $P(\boldsymbol{X} \mid \boldsymbol{E} = \boldsymbol{e})$ for some query variables $\boldsymbol{X} \subseteq \mathcal{X}$ and evidence $\boldsymbol{E} = \boldsymbol{e}$.

Example queries for our car tracking model:

- Given all the sensor readings $obs^{(0:t)}$ so far, from the beginning, and the current weather $w^{(t)}$, where is the car now?

$$P(Loc^{(t)} \mid obs^{(0)}, ..., obs^{(t)}, w^{(t)}) = ?$$

- Given the car's current velocity $vel^{(t)}$ and the current sensor reading $obs^{(t)}$, where will the car be in 2 seconds, and how fast will it be?

$$P(Loc^{(t+2)}, Vel^{(t+2)} \mid obs^{(t)}, vel^{(t)}) = ?$$

- Given the last 5 observations, how likely is it that the sensor is defect?

$$P(Fail^{(t)} \mid obs^{(t-4)}, \ldots, obs^{(t)}) = ?$$

Inference in Dynamic Bayesian Networks

$$P(Loc^{(t+2)}, Vel^{(t+2)} \mid obs^{(t)}, vel^{(t)}) = ?$$

In principle, this is straightforward:

- Unroll ("instantiate") the network to desired length T
- Use a standard Bayes Net inference algorithm (e.g., Variable Elimination) to compute the answer.

Problems:

- Unrolled network will be huge $\Rightarrow$ inference intractable
- In temporal settings, we might be interested in other types of reasoning than in static models
- In particular: may want to perform *online reasoning*, as the system evolves
- Will want to consider only inference tasks that do not require us to explicitly represent the full unrolled network

Inference in Unconstrained DBNs

Too complex to be treated in detail in this class.

The short version:

Exact inference in unconstrained Dynamic Bayesian Networks is computationally extremely expensive (intractable)!

In the following:

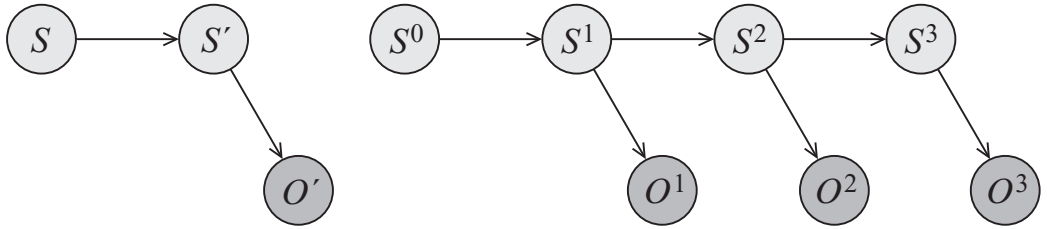
- Focus on two more restricted (simpler) classes of DBNs: Hidden Markov Models and Kalman Filters
- ... both of which are members of the family of ☞ **State-Observation Models**.

An Important Sub-Class of DBNs: State-Observation Models

Basic Idea:

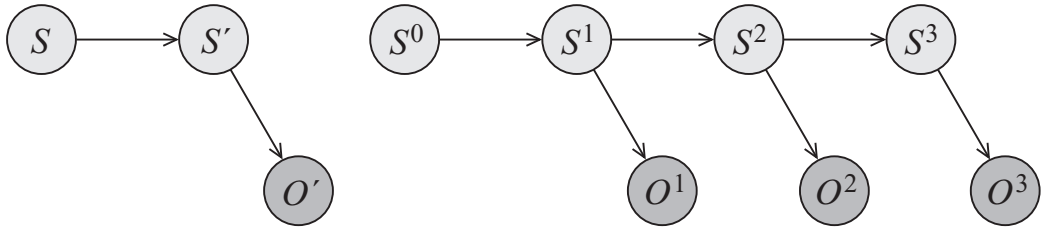
- Think of a system with an **internal state** that evolves over time, according to some rules (system dynamics)
- The state itself is not observable, but we can observe/measure some **effects** that are produced, probabilistically, by the system, depending on its state
- Especially appropriate when our observations are obtained by some (possibly noisy, unreliable) sensors
- Example: Patient in intensive care unit.
- Split variables into two subsets:
  State Variables $\boldsymbol{S}$ (unobservable) and **Observation Variables** $\boldsymbol{O}$:

$$\boxed{\mathcal{X} = \boldsymbol{S} \cup \boldsymbol{O}}$$



(a) (b)

State-Observation Models



(a) (b)

Permits separation into

- a model of the **system dynamics**: describes development of the internal system state S over time – the transitions between system states
- an **observation model**: describes the behaviour of our sensors (i.e., how internal states may lead to specific sensor measurements O)

Definition

A **State-Observation Model** is a DBN that consists of three components:

1. an **Initial State Model** $P(S^{(0)})$
2. a **State Transition Model** $P(S' \mid S)$
3. an **Observation Model** $P(O \mid S)$

Independence Assumptions in State-Observation Models

Assumption 1:

The **state transitions satisfy the Markov assumption**:

$$(\boldsymbol{S}^{(t+1)} \perp \boldsymbol{S}^{(0:t-1)} \mid \boldsymbol{S}^{(t)})$$
$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{S}^{(0:t)}) = P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{S}^{(t)})$$

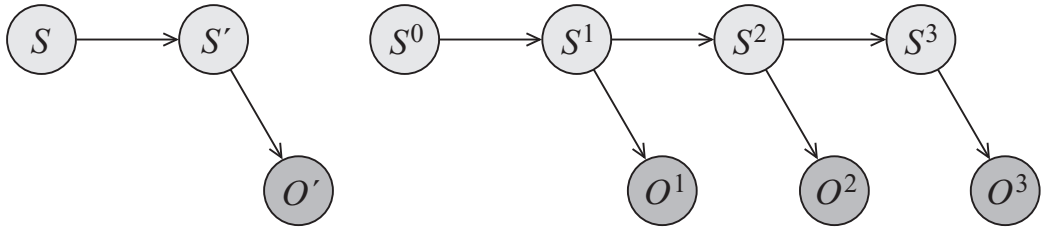
Assumption 2:

The observations at time t are conditionally independent of previous states and observations, given the current state.
In other words: **the current observations depend only on the current state.**

$$(\boldsymbol{O}^{(t)} \perp \boldsymbol{S}^{(0:1-t)}, \boldsymbol{O}^{(0:1-t)} \mid \boldsymbol{S}^{(t)})$$
$$P(\boldsymbol{O}^{(t)} \mid \boldsymbol{S}^{(0:t)}, \boldsymbol{O}^{(0:t-1)}) = P(\boldsymbol{O}^{(t)} \mid \boldsymbol{S}^{(t)})$$

State-Observation Models as Distributions

This is reflected in the graph structure:[1]



(a) (b)

Direct consequence (by Chain Rule for Bayesian Networks):

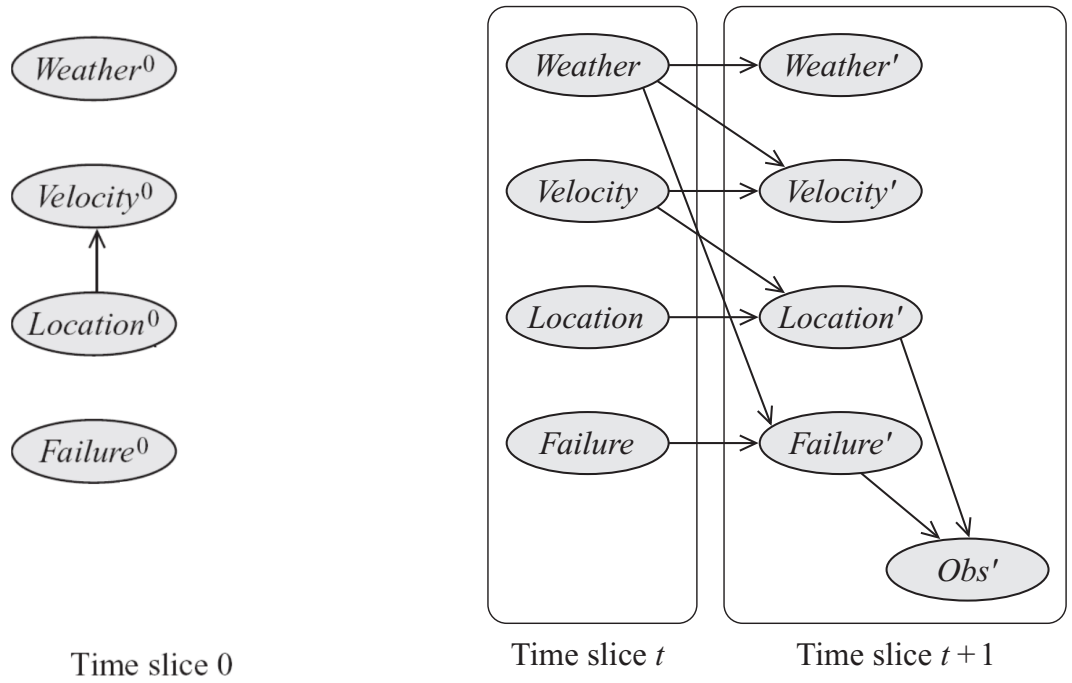
Definition

For any finite T, the **JOINT DISTRIBUTION** represented by a **State-Observation Model** over variables $\mathcal{X}^{(0:T)} = \boldsymbol{S}^{(0:T)} \cup \boldsymbol{O}^{(1:T)}$ is

$$P(\boldsymbol{S}^{(0)}, \ldots, \boldsymbol{S}^{(T)}, \boldsymbol{O}^{(1)}, \ldots, \boldsymbol{O}^{(T)}) = P(\boldsymbol{S}^{(0)}) \prod_{i=1}^{T} P(\boldsymbol{S}^{(i)} | \boldsymbol{S}^{(i-1)}) P(\boldsymbol{O}^{(i)} | \boldsymbol{S}^{(i)})$$

[1] For various technical reasons, we will assume, from now on, that there is no observation at time slice 0; observations start coming in with $t = 1$.

Is Our Car Tracking Network a State-Observation Model?



... only if we consider $Weather$ a hidden state variable ...

Three Common Inference Tasks in S-O Models

FILTERING/TRACKING: Compute our belief about the current system state $\boldsymbol{S}^{(t)}$, given all of the observations made so far:

$$P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)}) = ?$$

"Tracking" then means maintaining this belief state over time, on-line.

SMOOTHING: Compute the posterior distribution over the system state at time t, given all of the evidence $\boldsymbol{o}^{(1:T)}$ over some *longer* trajectory ($t < T$)

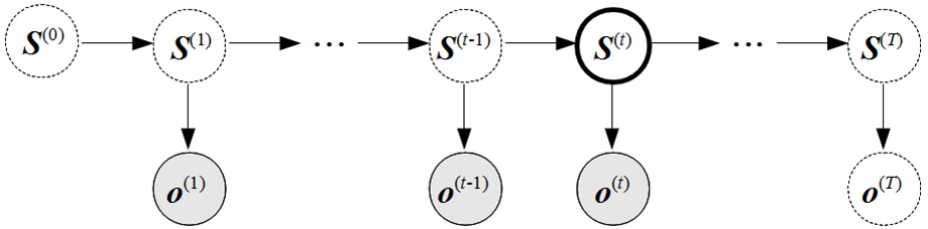
$$P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:T)}) = ?$$

PREDICTION: Given all the observations up to t, predict the distribution over some future state(s) at $t + k > t$:

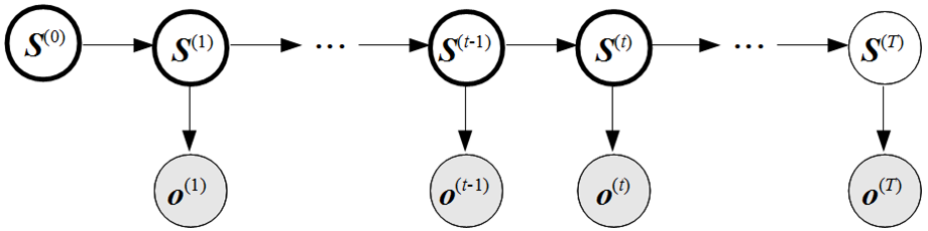
$$P(\boldsymbol{S}^{(t+k)} \mid \boldsymbol{o}^{(1:t)}) = ?$$

Three Common Inference Tasks in S-O Models

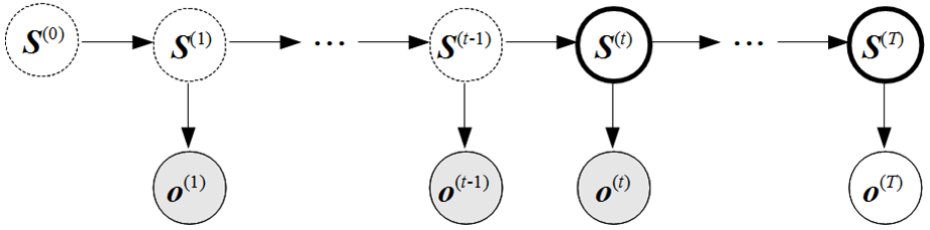
FILTERING/TRACKING: $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)}) = ?$



SMOOTHING: $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:T)}) = ?$



PREDICTION: $P(\boldsymbol{S}^{(t+k)} \mid \boldsymbol{o}^{(1:t)}) = ?$



Filtering vs. Smoothing

Question:

Why would $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$ and $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:T)})$ be different?

Consider vehicle tracking task:

- Assume current and recent sensor readings $o^{(t-k:t)}$ imply that car has stopped moving
- Most probable conclusion at time t: car stopped or is broken
- But if later observations $o^{(t+1)}, o^{(t+2)}, \ldots$ indicate that the car is far ahead of where it was at t, then the hypothesis that at time t it was broken, was probably wrong
- More likely explanation in hindsight: a temporary sensor failure.

☞ **"Hindsight": Later information may change our belief about previous states that the system might have gone through.**

Exact Inference in S-O Models (1): Filtering

Inference Task 1: FILTERING ("maintaining the belief state")

- Compute $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$

NAIVE ALGORITHM:

- Unroll network over the first t time slices
- Use a standard inference algorithm to compute $P(\boldsymbol{S}^{(t)}|\boldsymbol{o}^{(1:t)})$ from the resulting network, using Full Joint Distribution:

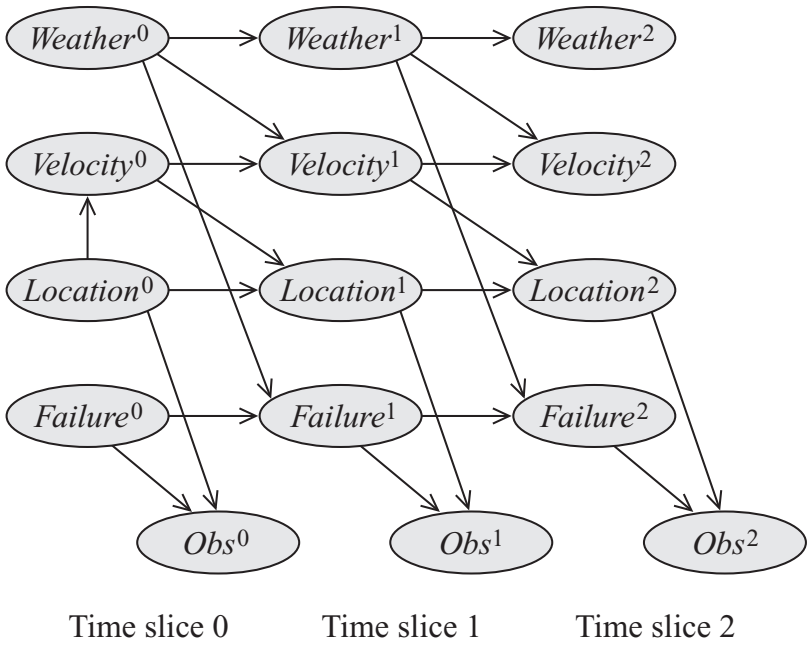
$$P(\boldsymbol{S}^{(0:T)}, \boldsymbol{O}^{(1:T)}) = P(\boldsymbol{S}^{(0)}) \prod_{t=1}^{T} P(\boldsymbol{S}^{(t)}|\boldsymbol{S}^{(t-1)}) P(\boldsymbol{O}^{(t)}|\boldsymbol{S}^{(t)})$$

Problems:

- Must sum out over a large number of hidden variables (all $S_j^{(i)} \in \boldsymbol{S}^{(i)}$, for $i = 0 \ldots t-1$)
- Must keep the entire history of observations $\boldsymbol{o}^{(1)}, \boldsymbol{o}^{(2)}, \ldots, \boldsymbol{o}^{(t)}$
- This grows worse with growing t ...

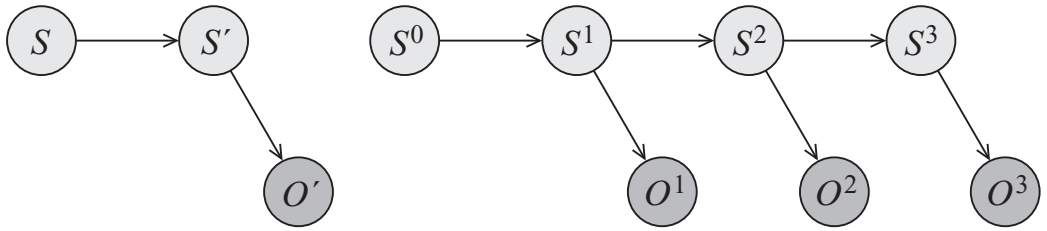
Exact Inference in S-O Models (1): Filtering

Example: Compute $P(Loc^{(100)} \mid \langle obs^{(1)}, obs^{(2)}, \ldots, obs^{(100)} \rangle)$



- Must sum out over $4 \times 100 + 3$ hidden variables!
  (all the $Weathers$, $Velocities$, $Locations$, $Failures$ for $t = 0 \ldots 99$)
- Computationally intractable.

Filtering: An Efficient, Recursive Algorithm



(a) (b)

Idea: Exploit linear structure (independencies) in State-Observation Models

1. Distribution over next state $\boldsymbol{S}^{(t+1)}$ depends only on current state $\boldsymbol{s}^{(t)}$:

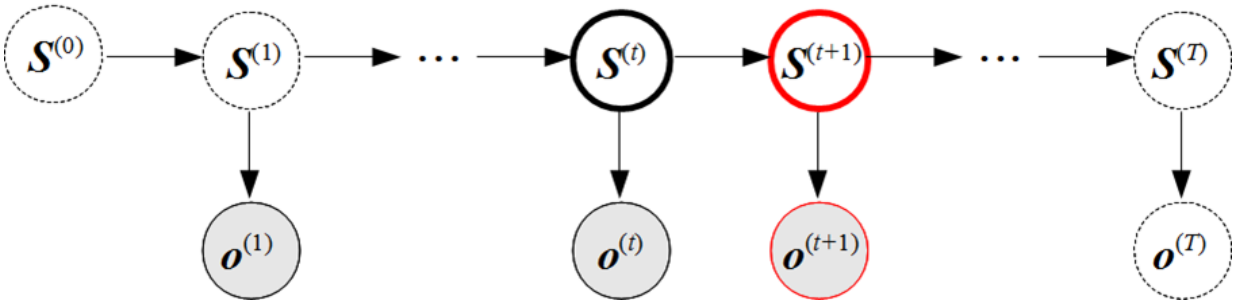
$$\boxed{(\boldsymbol{S}^{(t+1)} \perp \boldsymbol{S}^{(0:t-1)}, \boldsymbol{O}^{(0:t)} \mid \boldsymbol{S}^{(t)})}$$

   (Note also independence of $\boldsymbol{S}^{(t+1)}$ from previous observations $\boldsymbol{O}^{(0:t)}$!)

2. Observations $\boldsymbol{o}^{(t)}$ at time t depend only on state $\boldsymbol{s}^{(t)}$:

$$\boxed{(\boldsymbol{O}^{(t)} \perp \boldsymbol{S}^{(0:1-t)}, \boldsymbol{O}^{(0:t-1)} \mid \boldsymbol{S}^{(t)})}$$

Filtering: An Efficient, Recursive Algorithm



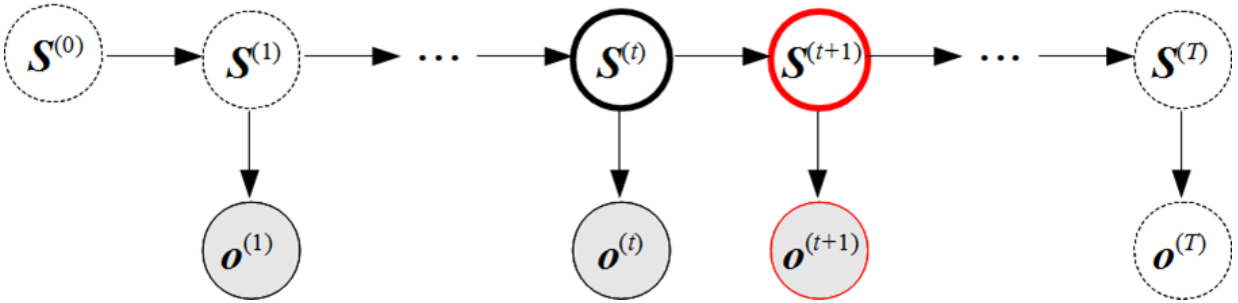
This inspires a Recursive (Inductive) Update Algorithm:

Assume the distribution $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$ can be computed.

Show how to compute $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)})$ from $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$ in two steps:

1. **Propagate state forward:** compute
$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)}) \text{ from } P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$$
2. **Take into account new observation $\boldsymbol{o}^{(t+1)}$:** compute
$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)}) \text{ from } P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})$$

Filtering: An Efficient, Recursive Algorithm



Initialisation:

$$P(\boldsymbol{S}^{(0)} \mid \boldsymbol{o}^{(1:0)}) = P(\boldsymbol{S}^{(0)})$$

is given directly by the CPD tables of the time slice 0 model $\mathcal{B}_0$.

Notes:

- From now on, assume that there is no observation in first time slice $t = 0$
- Observations start at $t = 1$
- Interpret $\boldsymbol{o}^{(1:0)}$ as the empty observation sequence $\{\}$

Filtering: An Efficient, Recursive Algorithm

Inductive Step 1: State Forward Propagation

$$\boxed{P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})} = \sum_{\boldsymbol{s}^{(t)}} P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{s}^{(t)}, \boldsymbol{o}^{(1:t)}) P(\boldsymbol{s}^{(t)} \mid \boldsymbol{o}^{(1:t)}) \quad (1)$$

$$= \boxed{\sum_{\boldsymbol{s}^{(t)}} P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{s}^{(t)}) P(\boldsymbol{s}^{(t)} \mid \boldsymbol{o}^{(1:t)})} \quad (2)$$

(1): law of total probability (conditional version)

(2): conditional independence of state from previous observations.

In Words:

The probability of being in a specific state $\boldsymbol{s}^{(t+1)}$ at time $t+1$ after having observed $\boldsymbol{o}^{(1:t)}$ up until time t is the sum, over all possible previous states $\boldsymbol{s}^{(t)}$, of the system having been in state $\boldsymbol{s}^{(t)}$ at time t (given the observations $\boldsymbol{o}^{(1:t)}$), times the probability of the system moving from state $\boldsymbol{s}^{(t)}$ to $\boldsymbol{s}^{(t+1)}$ in one step (the transition probability $P(\boldsymbol{s}^{(t+1)}|\boldsymbol{s}^{(t)})$).

Filtering: An Efficient, Recursive Algorithm

Inductive Step 1: State Forward Propagation

$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)}) = \sum_{\boldsymbol{s}^{(t)}} P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{s}^{(t)}) P(\boldsymbol{s}^{(t)} \mid \boldsymbol{o}^{(1:t)})$$

- $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$ is known (inductive assumption)
- $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{s}^{(t)})$ is given by the transition model $P(\boldsymbol{S}' | \boldsymbol{S})$

$\Rightarrow$ $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})$ is effectively computable.

☞ $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})$ is called the **Prior Belief State** at time $t+1$

("prior": ***before*** we have seen the next observation $\boldsymbol{o}^{(t+1)}$)

Filtering: An Efficient, Recursive Algorithm

Inductive Step 2: Conditioning – Account for New Observation

$$\boxed{P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)})} = P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)}, \boldsymbol{o}^{(t+1)}) \tag{3}$$

$$= \frac{P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)}, \boldsymbol{o}^{(1:t)}) P(\boldsymbol{S}^{(t+1)} | \boldsymbol{o}^{(1:t)})}{P(\boldsymbol{o}^{(t+1)} | \boldsymbol{o}^{(1:t)})} \tag{4}$$

$$= \frac{P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)}) P(\boldsymbol{S}^{(t+1)} | \boldsymbol{o}^{(1:t)})}{P(\boldsymbol{o}^{(t+1)} | \boldsymbol{o}^{(1:t)})} \tag{5}$$

$$= \boxed{\frac{1}{Z} P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)}) P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})} \tag{6}$$

(4): Conditional version of Bayes' rule: $P(A|B, Z) = P(B|A, Z)P(A|Z)/P(B|Z)$
(5): Conditional independence of $\boldsymbol{o}^{(t+1)}$ *and* $\boldsymbol{o}^{(1:t)}$*, given* $\boldsymbol{S}^{(t+1)}$
(6): Renormalisation

In Words:

The probability of being in state $\boldsymbol{s}^{(t+1)}$ at time $t+1$ after having observed all $\boldsymbol{o}^{(1:t+1)}$ (including $\boldsymbol{o}^{(t+1)}$) is proportional to the probability of being in $\boldsymbol{s}^{(t+1)}$ after having observed $\boldsymbol{o}^{(1:t)}$, times the probability that $\boldsymbol{s}^{(t+1)}$ would then produce observation $\boldsymbol{o}^{(t+1)}$.

Filtering: An Efficient, Recursive Algorithm

Inductive Step 2: Conditioning – Account for New Observation

$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)}) \quad = \quad \frac{1}{Z} P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)}) P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})$$

- $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)}) =$ prior belief state computed in previous step
- $P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)})$ is given by observation model $P(\boldsymbol{O}|\boldsymbol{S})$

$\Rightarrow$ $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)})$ is effectively computable.

☞ $P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)})$ is called the **Posterior Belief State**

("posterior": ***after*** we have taken into account the new observation $\boldsymbol{o}^{(t+1)}$)

Summary: The General Filtering Algorithm

General Filtering Algorithm for State-Observation-Models

Initialisation: Start with initial state distribution

$$P(\boldsymbol{S}^{(0)} \mid \boldsymbol{o}^{(1:0)}) = P(\boldsymbol{S}^{(0)} \mid \{\}) = P(\boldsymbol{S}^{(0)})$$

Forward propagation: For $t = 0$ to $T - 1$ do:

1. Propagate state distribution forward:

$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)}) = \sum_{\boldsymbol{s}^{(t)}} P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{s}^{(t)}) P(\boldsymbol{s}^{(t)} \mid \boldsymbol{o}^{(1:t)})$$

2. Condition on new observation:

$$P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t+1)}) = \frac{1}{Z} P(\boldsymbol{o}^{(t+1)} \mid \boldsymbol{S}^{(t+1)}) P(\boldsymbol{S}^{(t+1)} \mid \boldsymbol{o}^{(1:t)})$$

Termination: Return distribution

$$P(\boldsymbol{S}^{(T)} \mid \boldsymbol{o}^{(1:T)})$$

A More Abstract View: The "Forward Algorithm"

In each iteration, the algorithm

- takes the current filtered state distribution $P(\boldsymbol{S}^{(t)}|\boldsymbol{o}^{(1:t)})$
- propagates it one step forward via the transition model
- updates (conditions) with the new observation, and
- renormalises to obtain the new filtered distribution at $t+1$.

Abstract View:

- View current filtered state distribution $P(\boldsymbol{S}^{(t)}|\boldsymbol{o}^{(1:t)})$ as a ***"message"*** $\mathrm{f}^{(1:t)}$ that is propagated forward and updated in each time step
- Think of the two steps *state propagation* and *conditioning* (without the renormalisation) as being encapsulated in a function FORWARD
- (This function will be re-used in later algorithms..)

The "Forward Algorithm" for Filtering

The FORWARD ALGORITHM for Filtering

Initialisation:

$$\mathsf{f}^{(1:0)} = P(\boldsymbol{S}^{(0)})$$

Recursion: For $t = 0$ to $T - 1$ do:

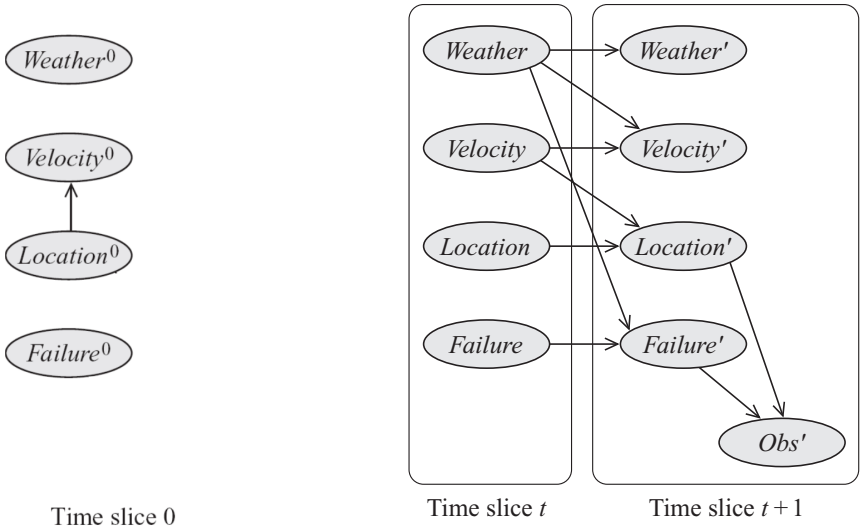
$$\mathsf{f}^{(1:t+1)} = 1/Z \times \textsc{Forward}(\mathsf{f}^{(1:t)}, \boldsymbol{o}^{(t+1)})$$

Termination: Return $\mathsf{f}^{(1:T)}$ as the posterior state distribution at time T.

Function FORWARD($\mathsf{f}^{(1:t)}, \boldsymbol{o}^{(t+1)}$)

$$\text{return} \quad \underbrace{P(\boldsymbol{o}^{(t+1)}|\boldsymbol{S}^{(t+1)}) \underbrace{\sum_{\boldsymbol{s}^{(t)}} P(\boldsymbol{S}^{(t+1)}|\boldsymbol{s}^{(t)}) \mathsf{f}^{(1:t)}_{\boldsymbol{s}}}_{\text{State Forward Propagation}}}_{\text{Conditioning}}$$

A Forward Step in our Car Tracking Network



1. **Initialisation:**

$$\mathsf{f}^{(1:0)} = P(W^{(0)}, V^{(0)}, L^{(0)}, F^{(0)}) = P(W^{(0)})P(L^{(0)})P(V^{(0)}|L^{(0)})P(F^{(0)})$$

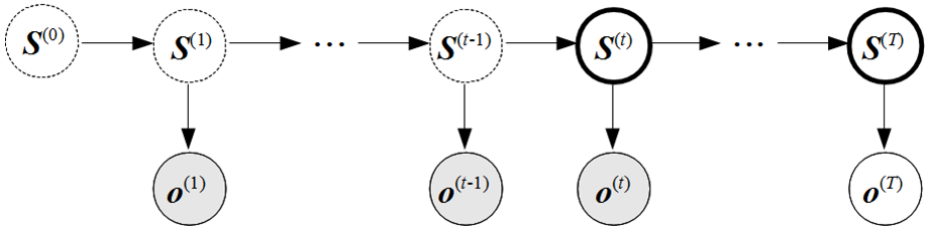
2. **State Propagation:**

$$\begin{aligned} \boldsymbol{U} &= P(W^{(1)}, V^{(1)}, L^{(1)}, F^{(1)} \mid \{\}) \\ &= \sum_{w,v,l,f} P(W^{(1)}|w)P(V^{(1)}|w,v)P(L^{(1)}|v,l)P(F^{(1)}|w,f) \times \mathsf{f}^{(1:0)}_{w,v,l,f} \end{aligned}$$

3. **Conditioning and Re-normalisation:**

$$\mathsf{f}^{(1:1)} = P(W^{(1)}, V^{(1)}, L^{(1)}, F^{(1)} \mid \{o^{(1)}\}) = \frac{1}{Z} \times P(o^{(1)} \mid L^{(1)}, F^{(1)}) \times \boldsymbol{U}$$

Exact Inference in S-O Models (2): Prediction



Inference Task 2: PREDICTION

- Compute $P(S^{(t+k)} \mid o^{(1:t)})$, for some $k > 0$
- Needed to anticipate future situations

Notes:

- Prediction can be viewed as **filtering without new evidence**
- One-step prediction is obvious: equivalent to the state propagation step in the Forward Algorithm (see above)
- Given a prediction for the state distribution at time $t + k$, the prediction for $t + k + 1$ can be easily computed via another state propagation step.

☞ **Simple inductive algorithm** (see next slide)

Exact Inference in S-O Models (2): Prediction

General Prediction Algorithm for State-Observation Models

Starting Point:

- Posterior state distribution $P(\boldsymbol{S}^{(t)} \mid \boldsymbol{o}^{(1:t)})$ at current time t

Inductive Step:

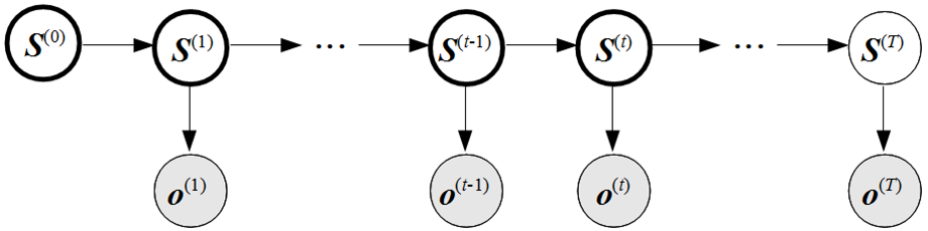
- Given $P(\boldsymbol{S}^{(t+k)} \mid \boldsymbol{o}^{(1:t)})$ (where $k \geq 0$),
- Predict next distribution as

$$P(\boldsymbol{S}^{(t+k+1)} \mid \boldsymbol{o}^{(1:t)}) = \sum_{\boldsymbol{s}^{(t+k)}} \underbrace{P(\boldsymbol{S}^{(t+k+1)} | \boldsymbol{s}^{(t+k)})}_{\text{Transition Model } P(\boldsymbol{S}' | \boldsymbol{S})} P(\boldsymbol{s}^{(t+k)} | \boldsymbol{o}^{(1:t)})$$

Notes:

- No new observations
- Prediction of future state distributions relies only on the *Transition Model*
- Predictions will soon become very unreliable
- ... and converge to the **stationary distribution** of the Markov process defined by the Transition Model.

Exact Inference in S-O Models (3): Smoothing



Inference Task 3: SMOOTHING

- Compute $P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)})$, for some $k < T$

Notes:

- Smoothing = computing the distribution over past states k, given evidence up to the present T
- Needed for better interpretation of previous situations, and for learning (will see that in the chapter on ☞ Hidden Markov Models)
- Will be solved by the *Forward-Backward Algorithm*.

Exact Inference in S-O Models (3): Smoothing

Inference Task 3: SMOOTHING

- Compute $P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)})$, for some $k < T$

Trick: Divide the task into two parts – the evidence up to k, and the evidence from $k+1$ to T:

$$\underline{P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)})} = \underline{P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:k)}, \boldsymbol{o}^{(k+1:T)})} \quad (7)$$

$$= 1/Z \times P(\boldsymbol{o}^{(k+1:T)} \mid \boldsymbol{S}^{(k)}, \boldsymbol{o}^{(1:k)}) P(\boldsymbol{S}^{(k)} | \boldsymbol{o}^{(1:k)}) \quad (8)$$

$$= 1/Z \times P(\boldsymbol{o}^{(k+1:T)} \mid \boldsymbol{S}^{(k)}) P(\boldsymbol{S}^{(k)} | \boldsymbol{o}^{(1:k)}) \quad (9)$$

$$= 1/Z \times \underbrace{P(\boldsymbol{S}^{(k)} | \boldsymbol{o}^{(1:k)})}\ \underbrace{P(\boldsymbol{o}^{(k+1:T)} | \boldsymbol{S}^{(k)})} \quad (10)$$

$$= \underline{1/Z \times \quad \mathsf{f}^{(1:k)} \quad \times \quad \mathsf{b}^{(k+1:T)}} \quad (11)$$

(8): (conditional version of) Bayes' rule
(9): conditional independence of future from previous observations, given state
(10): $A \cdot B = B \cdot A$ ...

Exact Inference in S-O Models (3): Smoothing

$$
\begin{aligned}
P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)}) &= 1/Z \times \underbrace{P(\boldsymbol{S}^{(k)} | \boldsymbol{o}^{(1:k)})}_{\mathsf{f}^{(1:k)}} \underbrace{P(\boldsymbol{o}^{(k+1:T)} | \boldsymbol{S}^{(k)})}_{\mathsf{b}^{(k+1:T)}} \\
&= 1/Z \times \mathsf{f}^{(1:k)} \times \mathsf{b}^{(k+1:T)}
\end{aligned}
$$

Notes:

- $\mathsf{f}^{(1:k)}$ is our **forward message** from before
- $\mathsf{b}^{(k+1:T)}$ will be called **backward message**
- $\mathsf{b}^{(k+1:T)}$ can be computed recursively, starting from the end of the trajectory and running backwards from T (see next slide for a derivation).

☞ Both messages can be computed recursively, with fixed effort per step – one forward in time, the other backward.

☞ Leads to the **Forward-Backward Algorithm** (see below).

Computing the Backward Message

$$\underline{\mathrm{b}^{(k+1:T)}} = \underline{P(\boldsymbol{o}^{(k+1:T)} \mid \boldsymbol{S}^{(k)})} \tag{12}$$

$$= \sum_{\boldsymbol{s}^{(k+1)}} P(\boldsymbol{o}^{(k+1:T)} | \boldsymbol{S}^{(k)}, \boldsymbol{s}^{(k+1)}) P(\boldsymbol{s}^{(k+1)} | \boldsymbol{S}^{(k)}) \tag{13}$$

$$= \sum_{\boldsymbol{s}^{(k+1)}} P(\boldsymbol{o}^{(k+1:T)} | \boldsymbol{s}^{(k+1)}) P(\boldsymbol{s}^{(k+1)} | \boldsymbol{S}^{(k)}) \tag{14}$$

$$= \sum_{\boldsymbol{s}^{(k+1)}} P(\boldsymbol{o}^{(k+1)}, \boldsymbol{o}^{(k+2:T)} | \boldsymbol{s}^{(k+1)}) P(\boldsymbol{s}^{(k+1)} | \boldsymbol{S}^{(k)}) \tag{15}$$

$$= \sum_{\boldsymbol{s}^{(k+1)}} \underbrace{P(\boldsymbol{o}^{(k+1)} | \boldsymbol{s}^{(k+1)})}_{\text{Observation Model}} \underbrace{P(\boldsymbol{o}^{(k+2:T)} | \boldsymbol{s}^{(k+1)})}_{\underline{\mathrm{b}^{(k+2:T)}}} \underbrace{P(\boldsymbol{s}^{(k+1)} | \boldsymbol{S}^{(k)})}_{\text{Transition Model}} \tag{16}$$

(13): conditioning on $\boldsymbol{S}^{(k+1)}$ (law of total probability)
(14): conditional independence of observation from previous state, given current state
(15): writing $\boldsymbol{o}^{(k+1:T)}$ as $(\boldsymbol{o}^{(k+1)}, \boldsymbol{o}^{(k+2:T)})$
(16): conditional independence of $\boldsymbol{o}^{(k+1)}$ and $\boldsymbol{o}^{(k+2:T)}$, given $\boldsymbol{S}^{(k+1)}$.

Computing the Backward Message

$$\begin{aligned}\underline{\mathsf{b}^{(k+1:T)}} &= P(\boldsymbol{o}^{(k+1:T)} \mid \boldsymbol{S}^{(k)}) \\ &= \sum_{\boldsymbol{s}^{(k+1)}} \underbrace{P(\boldsymbol{s}^{(k+1)}|\boldsymbol{S}^{(k)})}_{\text{Transition Model}} \underbrace{P(\boldsymbol{o}^{(k+1)}|\boldsymbol{s}^{(k+1)})}_{\text{Observation Model}} \underbrace{P(\boldsymbol{o}^{(k+2:T)}|\boldsymbol{s}^{(k+1)})}_{\underline{\mathsf{b}^{(k+2:T)}}}\end{aligned}$$

In Words:

The probability of observing the remaining observations $\boldsymbol{o}^{(k+1:T)}$, given that at time k we are in state $\boldsymbol{s}^{(k)}$, is the sum, over all possible successor states $\boldsymbol{s}^{(k+1)}$, of the probability of moving from $\boldsymbol{s}^{(k)}$ to $\boldsymbol{s}^{(k+1)}$, then observing $\boldsymbol{o}^{(k+1)}$, and then observing the rest $\boldsymbol{o}^{(k+2:T)}$ from there onwards.

☞ Think of this as being encapsulated in a function BACKWARD:

$$\boxed{\mathsf{b}^{(k+1:T)} = \textsc{Backward}(\mathsf{b}^{(k+2:T)}, \boldsymbol{o}^{(k+1)})}$$

Smoothing: The Forward-Backward Algorithm

The FORWARD-BACKWARD ALGORITHM for Smoothing

Goal: Compute $P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)})$, for all $k = 0, \ldots, T$

- **Initialise f:** $\underline{\mathsf{f}^{(1:0)} = P(\boldsymbol{S}^{(0)})}$
- **Forward Pass:** For $k = 1$ to T compute (and **store**):

$$\underline{\mathsf{f}^{(1:k)} = 1/Z \times \textsc{Forward}(\mathsf{f}^{(1:k-1)}, \boldsymbol{o}^{(k)})}$$

- **Initialise b:** $\underline{\mathsf{b}^{(T+1:T)} = P(\boldsymbol{o}^{(T+1:T)} \mid \boldsymbol{S}^{(T)}) = \mathbf{1}}$ (a vector of all 1's)[a]
- **Backward Pass:** For $k = T$ downto 1 do
  1. Compute smoothed posterior distribution at k:

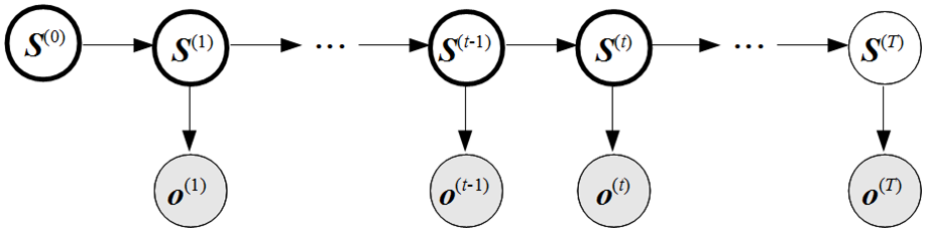
$$\boxed{P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)}) = 1/Z \times \mathsf{f}^{(1:k)} \times \mathsf{b}^{(k+1:T)}}$$

  2. Pass backward message backwards:

$$\underline{\mathsf{b}^{(k:T)} = \textsc{Backward}(\mathsf{b}^{(k+1:T)}, \boldsymbol{o}^{(k)})}$$

[a] The probability of the observing the empty sequence $\boldsymbol{o}^{(T+1:T)} = \{\}$ after any final state is 1.

Smoothing: The Forward-Backward Algorithm



In Words:

- **Forward Pass:** go forward in time, estimate probability distribution over current state, given observations so far: $\mathsf{f}^{(1:k)} = P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:k)})$
- **Backward Pass:** go backward in time, in each step correct forward estimate $\mathsf{f}^{(1:k)}$ by considering what happened later (evidence $\boldsymbol{o}^{k+1:T}$):

$$\begin{aligned} P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:T)}) &= 1/Z \times \underbrace{P(\boldsymbol{S}^{(k)} | \boldsymbol{o}^{(1:k)})}_{\mathsf{f}^{(1:k)}} \underbrace{P(\boldsymbol{o}^{(k+1:T)} | \boldsymbol{S}^{(k)})}_{\mathsf{b}^{(k+1:T)}} \\ &= 1/Z \times \mathsf{f}^{(1:k)} \times \mathsf{b}^{(k+1:T)} \end{aligned}$$

Note:

- Forward message $\mathsf{f}^{(1:k)} = P(\boldsymbol{S}^{(k)} \mid \boldsymbol{o}^{(1:k)})$ is a proper *probability distribution* over possible states $\boldsymbol{S}^{(k)}$
- Backward message $\mathsf{b}^{(k+1:T)} = P(\boldsymbol{o}^{(k+1:T)} \mid \boldsymbol{S}^{(k)})$ is a *likelihood function:* probability of seeing future data $\boldsymbol{o}^{(k+1:T)}$, given model and current state $\boldsymbol{s}^{(k)}$

☞ need to *re-normalise* by $1/Z$.

Is this getting too abstract for you?

Will see examples of these algorithms in the simpler setting of Hidden Markov Models (HMMs) ...

Summary: Exact Inference in State-Observation Models

There are fixed-effort, recursive algorithms for exact inference in State-Observation Models:

- Filtering: the Forward Algorithm
- Prediction: simple application of the Transition Model
- Smoothing: the Forward-Backward Algorithm.

Problem:

- In practice, these problems are still often intractable:
- Size of belief state (forward message) $P(\boldsymbol{S}^{(t)}|\boldsymbol{o}^{(1:t)})$ is *exponential* in the number of variables $|\boldsymbol{S}^{(t)}| = |\boldsymbol{S}|$,
- and we need to sum over all these possible states in each inference step.

Solutions:

- Look at **simpler, tractable models:**
  ☞ Hidden Markov Models (HMMs), Kalman Filters, ...
- Look at methods for **approximate reasoning:**
  ☞ *Particle Filters*.

What you should remember of this section

- Why temporal models are needed
- Why unrestricted unrolled temporal models would be intractable
- The 3 simplifying assumptions needed to make the concept practicable
- Definition of Dynamic Bayesian Networks (DBNs)
- Concept of State-Observation Models
- The main inference tasks in State-Observation Models
- Basic idea of the forward algorithm, and the forward-backward algorithm
- That inference in general State-Observation Models is still intractable.

Literature

Koller, Daphne and Friedman, Nir (2009).
Probabilistic Graphical Models: Principles and Techniques. Cambridge, MA: MIT Press.

Russell, Stuart J. and Norvig, Peter (2003).
Artificial Intelligence: A Modern Approach. Englewood Cliffs, NJ: Prentice Hall.