MACHINE LEARNING
21CSC305P
Unit-4 – Hidden Markov Models

- **Sequential Data**
- **Markov Models**
- **HMM**
- **Maximum Likelihood for the HMM**
- **The Forward and Backward Algorithm**
- **The Sum-Product Algorithm**
- **Scaling Factors**
- **Viterbi Algorithm**
- **Linear Dynamical Systems**

SEQUENTIAL DATA

- For many applications (i.i.d.) will not work; so, it describes the concept of Sequential dataSequential data through a series of time:
    - Rainfall measurements on successive days at a particular location.
    - Daily values of a currency exchange rate
    - Acoustic features at successive time frames (speech recognition)
- Sequential data other than time series:
    - Sequence of Nucleotide base pairs along a strand of DNA
    - Sequence of Characters in English Sentence
- Stationary Sequential Distributions:
    - Data evolves in time; but distribution from which it is generated remains same
- Non-stationary Sequential Distributions:
    - Generative Distribution itself evolving with time
- Predict next value in a time series given observations of the previous values:
    - Financial Forecasting
- Complexity of model would grow without limit as the number of observations increases.
    - Markov Model: Assume that predictions are independent of all but the most recent observations.
- By the introduction of latent variables; lead to state space models:
    - *Hidden Markov Model*: Latent Variables are discrete
    - *Linear Dynamical Systems*: Latent Variable are Gaussian

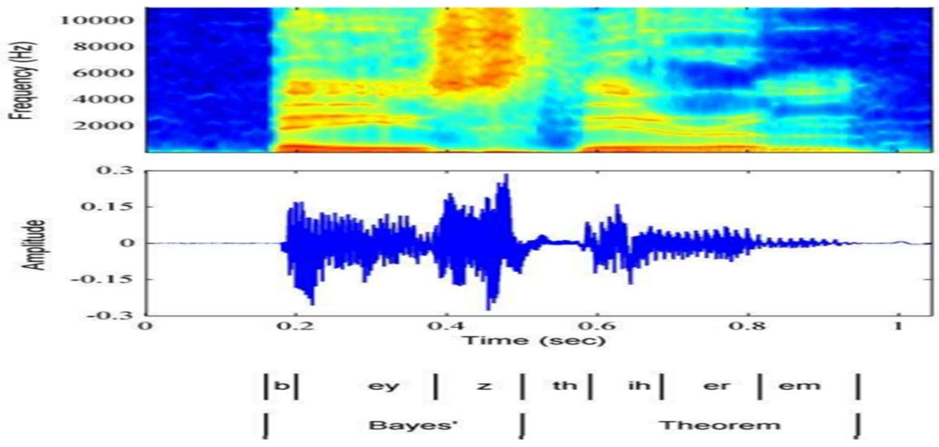
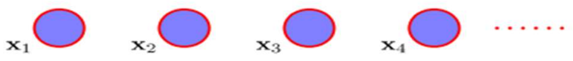


Figure: Example of Spectrogram of the Spoken Word "Bayes" theorem showing a plot of the intensity of the spectral coefficients versus time index.

MARKOV MODELS

- The simplest approach **to modelling a sequence of observations** is to treat them as independent, corresponding to a graph without links as shown in figure below:



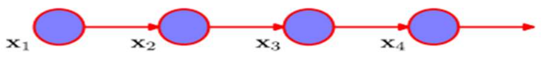
Independent and identically distributed

- Such an approach would fail to exploit the sequential patterns in data.
- A binary variable denoting whether on a particular day is rained or not.
- If we treat the data as i.i.d. then only information we glen from data is relative frequency of rainy days.

- To express such effect in probabilistic model, we need to relax the i.i.d. assumption: ***Product rule to express joint distribution for a sequence of observations***.

$$p(\mathbf{x}_1,\ldots,\mathbf{x}_N)=\prod_{n=1}^{N}p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}).$$

- Assume each of the conditional distributions on right-hand side is independent of all previous observations except the most recent; obtain ***First-Order Markov Chain***as shown in fig below***:***



  - The *Joint Distribution for a sequence of N Observations under this model:*

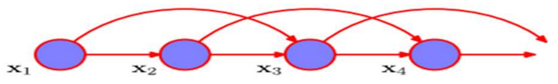
$$p(\mathbf{x}_1,\ldots,\mathbf{x}_N)=p(\mathbf{x}_1)\prod_{n=2}^{N}p(\mathbf{x}_n|\mathbf{x}_{n-1}).$$

  - From the *d-separation property*, conditional distribution for observation x_n, given all of observations upto time n,

$$p(\mathbf{x}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1})=p(\mathbf{x}_n|\mathbf{x}_{n-1})$$

- **Homogenous Markov Model:** If conditional distributions depend on adjustable parameters (whose values might be inferred from a set of training data) then all conditional distributions in chain will share the same values of those parameters.
- **Second-Order Markov Chain:** If we allow the predictions depend also on the previous-but-one value; joint distribution is given by:

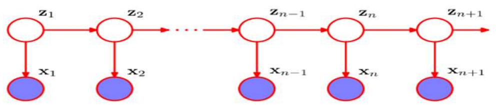
$$p(\mathbf{x}_1,\ldots,\mathbf{x}_N)=p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)\prod_{n=3}^{N}p(\mathbf{x}_n|\mathbf{x}_{n-1},\mathbf{x}_{n-2}).$$



  - **d-separation or by direct evaluation**, conditional distribution of $x_n$given by x_{n-1} and x_{n-2} is independent of all observations x_1,x_2, …., x_{n-3}.

- **Mth Order Markov Chain:** Conditional Distribution for a particular variable depends on the previous M variables. Suppose observations are discrete variables having K states.
  - Conditional distribution $p(x_n| x_{n-1})$ in a first-order Markov chain will be specified by a set of K-1 parameters for each of the K states of x_{n-1} total of K (K-1) parameters. Extend the model to Mth Order Markov Chain; **Joint Distributions: $p(x_n| x_{n-M}, \ldots, x_{n-1})$.**
  - If variables are discrete; conditional distributions are represented by general conditional probability tables: **Number of Parameters= K^{M-1} (K – 1) parameters**.

- For continuous variables; use **Linear Gaussian Conditional Distributions:** each node has a Gaussian distribution whose mean is a linear function of its parents. This is known an **Autoregressive Model or AR model.**
- An alternative approach is to use a parametric model for **$p(x_n | x_{n-M}, ..., x_{n-1})$ Neural Network;** sometimes called ***tapped delay line.***
- Suppose we wish to build a model for sequences that is not limited by Markov Assumption to any order and yet that can be specified using a limited number of free parameters; so add ***latent variables.*** For each observation x_n, corresponding latent variable z_n(State Space Model as shown in figure below):



- If satisfies the key conditional independence property **that z_{n-1} and z_{n+1} are independent given z_n:**

$$\mathbf{z}_{n+1} \perp\!\!\!\perp \mathbf{z}_{n-1} \mid \mathbf{z}_n.$$

- The **joint distribution is given by:**

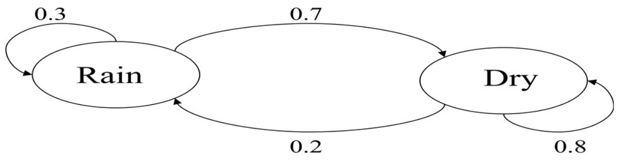
$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = p(\mathbf{z}_1) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n).$$

Markov Model

- **Set of states**: $\{s_1, s_2, \ldots, s_N\}$
- **Process moves from one state to another generating a sequence of states** :

  $s_{i1}, s_{i2}, \ldots, s_{ik}, \ldots$
- **Markov chain property:** probability of each subsequent state depends only on what was the previous state: $P(s_{ik} | s_{i1}, s_{i2}, \ldots, s_{ik-1}) = P(s_{ik} | s_{ik-1})$
- To define Markov model, the following probabilities have to be specified:
  - **transition probabilities** $a_{ij} = P(s_i | s_j)$
  - **initial probabilities** $\pi_i = P(s_i)$

Example of Markov Model:



- **States** : 'Rain' and 'Dry'
- **Transition probabilities:**

  P('Rain'|'Rain')=0.3

  P('Dry'|'Rain')=0.7

  P('Rain'|'Dry')=0.2

  P('Dry'|'Dry')=0.8
- **Initial probabilities**: say

  P('Rain')=0.4

  P('Dry')=0.6

Q: Suppose we want to calculate a probability of a sequence of states in our example: {'Dry','Dry','Rain',Rain'}.

Solution:

P({'Dry','Dry','Rain',Rain'}) = P('Dry') *P('Dry'|'Dry') *P('Rain'|'Dry') *P('Rain'|'Rain')
= 0.6 * 0.8*0.2*0.3

HMM

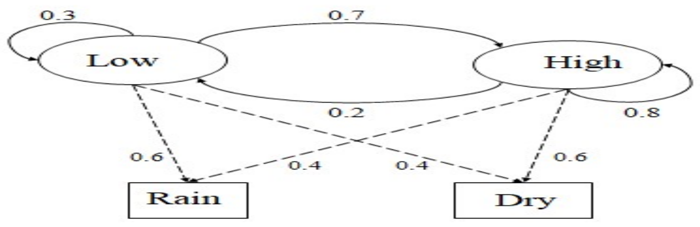
- The **Hidden Markov Model (HMM)** is a statistical model that is used to describe the probabilistic relationship between a sequence of observations and a sequence of hidden states.
- An HMM consists of two types of variables: **hidden states and observations**.
  - The **hidden states** are the underlying variables that generate the observed data, but they are not directly observable.
  - The **observations** are the variables that are measured and observed.
- Markov Model is widely used in ***Speech Recognition, Natural Language Modelling, On-line Handwriting Recognition, Analysis of Biological sequences such as Proteins, DNA.***
- The Hidden Markov Model (HMM) is the relationship between the hidden states and the observations using two sets of probabilities: the transition probabilities and the emission probabilities.
  - The **transition probabilities** describe the probability of transitioning from one hidden state to another.
  - The **emission probabilities** describe the probability of observing an output given a hidden state.

Hidden Markov Model

- **Set of hidden states**: $\{s_1, s_2, \dots, s_N\}$
- **Process moves from one state to another generating a sequence of states**:

  $s_{i1}, s_{i2}, \dots, s_{ik}, \dots$
- **Set of Observations:** $\{o_1, o_2, \dots, o_k\}$
- **Markov chain property:** probability of each subsequent state depends only on what was the previous state: $P(s_{ik} \mid s_{i1}, s_{i2}, \dots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$
- To define Markov model, the following probabilities have to be specified:
  - **transition probabilities** $a_{ij} = P(s_i \mid s_j)$
  - **initial probabilities** $\pi_i = P(s_i)$
  - **emission probabilities** $P(o_i \mid s_i)$

Example of Hidden Markov Model:



- **Two states:** 'Low' and 'High' atmospheric pressure.
- **Two observations**: 'Rain' and 'Dry'.

- **Transition probabilities:**
  - P('Low'|'Low')=0.3
  - P('High'|'Low')=0.7
  - P('Low'|'High')=0.2
  - P('High'|'High')=0.8
- **Emission/Observation probabilities:**
  - P('Rain'|'Low')=0.6
  - P('Dry'|'Low')=0.4
  - P('Rain'|'High')=0.4
  - P('Dry'|'High')=0.6
- **Initial probabilities:** say
  - P('Low')=0.4
  - P('High')=0.6

Calculation of observation sequence probability:

- Suppose we want to calculate a probability of a sequence of observations in our example, {'Dry','Rain'}.
- Consider all possible hidden state sequences:

P({'Dry','Rain'}) = P({'Dry','Rain'} , {'Low','Low'}) + P({'Dry','Rain'} , {'Low','High'}) + P({'Dry','Rain'} , {'High','Low'}) + P({'Dry','Rain'} , {'High','High'})

where first term is :

P({'Dry','Rain'} , {'Low','Low'}) = P({'Dry','Rain'} | {'Low','Low'}) P({'Low','Low'})
= P('Dry'|'Low')P('Rain'|'Low') P('Low')P('Low'|'Low)
= 0.4*0.6*0.4*0.3

Similary, Calculate others

Select maximum probability from among them.

If we had a set of states, we could calculate the probability of the sequence. But we don't have the states. All we have are a sequence of observations.

Problem Associated with HMM model:

- **Problem 1 (Evaluation):** Given an HMM and a sequence of observations, what is the probability that the observations are generated by the model? ***Forward-Backward algorithm solve this problem.***
- **Problem 2 (Decoding):** Given an HMM and a sequence of observations, what is the most likely state sequence in the model that produced the observations? ***Viterbi algorithm solve this problem.***
- **Problem 3 (Learning):** Given an observation sequence, how should the model parameters be adjusted to maximize the probability of the evaluation problem? ***Baum-Welch algorithm solve this problem***

MAXIMUM LIKELIHOOD FOR THE HMM

- If we have observed a data set X = {x1,..., xN }, we can determine the parameters of an HMM using maximum likelihood. The likelihood function is obtained from the joint distribution by marginalizing over the latent variables:

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}).$$

- The **number of terms in the summation grows exponentially with the length** of the **chain**. We cannot perform the summations explicitly because there are N variables to be summed over, each of which has K states, resulting in a total of K^N terms.
- **Direct maximization of the likelihood function** will therefore lead to complex expressions with no closed-form solutions, as was the case for simple mixture models.
- The **EM algorithm** starts with some **initial selection for the model parameters**, which we denote by $\boldsymbol{\theta}^{\text{old}}$.
- **In the E step**, we take these parameter values and find the *posterior distribution of the latent variables* **$p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$.** We then use this **posterior distribution** to evaluate the expectation of the logarithm of the complete-data likelihood function, as a function of the parameters θ, to give the function **$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$** defined by

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}).$$

  - We shall use ***$\gamma(z_n)$ to denote the marginal posterior distribution of a latent variable z_n***, and ***$\xi(z_{n-1}, z_n)$ to denote the joint posterior distribution of two successive latent variables***, so that:

$$\begin{aligned}\gamma(\mathbf{z}_n) &= p(\mathbf{z}_n|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \\ \xi(\mathbf{z}_{n-1}, \mathbf{z}_n) &= p(\mathbf{z}_{n-1}, \mathbf{z}_n|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}).\end{aligned}$$

- **In the M step, we maximize $Q(\theta, \theta^{\text{old}})$ with respect to the parameters $\theta = \{\pi, A, \varphi\}$ in which we treat $\gamma(z_n)$ and $\xi(z_{n-1}, z_n)$ as constant**. Maximization with respect to π and A is easily achieved using appropriate Lagrange multiplier the results:

$$\pi_k = \frac{\gamma(z_{1k})}{\sum_{j=1}^{K} \gamma(z_{1j})}$$

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nk})}{\sum_{l=1}^{K} \sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nl})}.$$

THE FORWARD AND BACKWARD ALGORITHM

The Forward-Backward Algorithm is a key technique used in Hidden Markov Models (HMMs) for computing the probabilities of hidden states given a sequence of observed events. It involves two main steps: the forward pass and the backward pass.

Forward Algorithm

1. **Initialization**:
   - For each state s_i, initialize the forward probability:

$$\alpha_1(i) = \pi_i \,.\, b_i(o_1)$$

where π_i is the initial probability of state s_i, $b_i(o_1)$ is the emission probability of the observed symbol o_1 from state s_i.

2. **Induction**:
   - For each time step t from 2 to T (length of the observation sequence), update $\alpha_t(i)$:

$$\alpha_t(i) = \sum_j \alpha_{t-1}(j) \cdot a_{ji} \cdot b_i(o_t)$$

Where α_{ij} is the transition probability from state s_j to state s_i, and $b_i\ (o_t)$ is the emission probability of the observed symbol o_t from state s_i.

3. **Termination**:
   - The probability of the entire observation sequence is given by:

$$P(O) = \sum_i \alpha_T(i)$$

Backward Algorithm

1. **Initialization**: For the final step T, set the backward probability for all states to 1.
   - Set $\beta_T(i)=1$ for all states s_i.
2. **Induction**:
   - For each time step t from T−1 to 1, update $\beta_t(i)$:

$$\beta_t(i) = \sum_j a_{ij} \cdot b_j(o_{t+1}) \cdot \beta_{t+1}(j)$$

Where α_{ij} = transition probability from state i to state j

$b_j\ (O_{t+1})$ Probability of observing O_{t+1} given state j

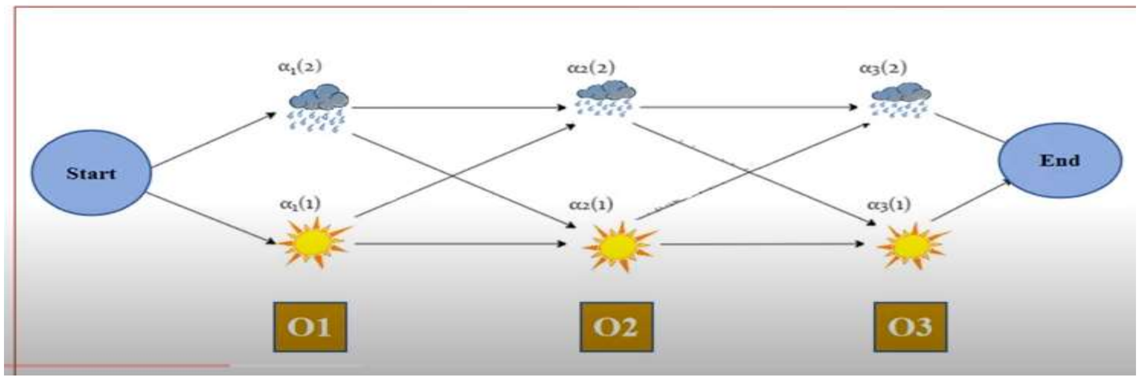
β_{t+1} (j)= Probability of observation sequence from t+1 to T give state j at time t+1

3. **Termination**:

The probability of the observation sequence O given the model is:

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i \cdot b_i(O_1) \cdot \beta_1(i)$$

Forward Algorithm Question



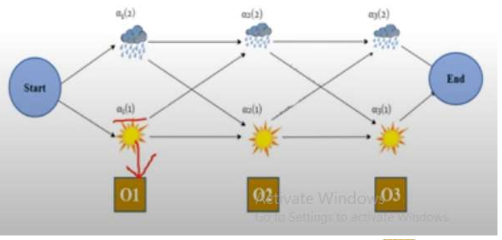
$\pi_1 = 0.6$

$\pi_2 = 0.4$ $\longleftarrow$ **Initial Probabilities**

$$a = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{array}{c|cc} & S & R \\ \hline S & 0.7 & 0.3 \\ R & 0.4 & 0.6 \end{array} \longleftarrow \textbf{Transition Probabilities}$$

$$b = \begin{bmatrix} b_1(O_1) & b_1(O_2) & b_1(O_3) \\ b_2(O_1) & b_2(O_2) & b_2(O_3) \end{bmatrix} = \begin{array}{c|ccc} S & \underline{0.1} & \underline{0.4} & \underline{0.5} \\ R & \underline{0.7} & \underline{0.2} & \underline{0.1} \end{array} \longleftarrow \textbf{Observation Probabilities}$$

Consider the Observation sequence O=[O$_1$, O$_2$, O$_3$]



Step 1:

$$\boxed{\alpha_1(1) = \underline{\pi_1} \cdot \underline{b_1(O_1)} = 0.6 \cdot 0.1 = \underline{0.06}}$$

$$\alpha_1(2) = \pi_2 \cdot b_2(O_1) = 0.4 \cdot 0.7 = 0.28$$

Step 2: For t=2 O1

$$\alpha_2(1) = [\underline{\alpha_1(1) \cdot a_{11}} + \underline{\alpha_1(2) \cdot a_{21}}] \cdot b_1(O_2) = [0.06 \cdot 0.7 + 0.28 \cdot 0.4] \cdot 0.4 = 0.0616$$

$$\alpha_2(2) = [\alpha_1(1) \cdot a_{12} + \alpha_1(2) \cdot a_{22}] \cdot b_2(O_2) = [0.06 \cdot 0.3 + 0.28 \cdot 0.6] \cdot 0.2 = 0.0372$$

Step 3: For t=3 O1

$$\alpha_3(1) = [\alpha_2(1) \cdot a_{11} + \alpha_2(2) \cdot a_{21}] \cdot b_1(O_3) = [0.0616 \cdot 0.7 + 0.0372 \cdot 0.4] \cdot 0.5 = 0.029$$

$$\alpha_3(2) = [\alpha_2(1) \cdot a_{12} + \alpha_2(2) \cdot a_{22}] \cdot b_2(O_3) = [0.0616 \cdot 0.3 + 0.0372 \cdot 0.6] \cdot 0.1 = 0.00408$$

Answer can be found out by summing up the α values

$$\boxed{\alpha_3(1) + \alpha_3(2) = 0.029 + 0.00408 = 0.03308}$$

<u>Backward Algorithm Question</u>

Step 1: Initial Probabilties

$\beta_3(1)=1$

$\beta_3(2)=1$

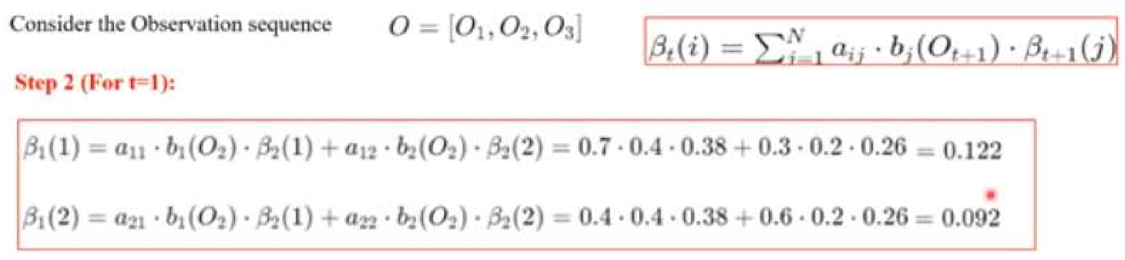
Consider the Observation sequence $O = [O_1, O_2, O_3]$

$$\boxed{\beta_t(i) = \sum_{j=1}^{N} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)}$$

Step 2 (For t=2):

$$\beta_2(1) = a_{11} \cdot b_1(O_3) \cdot \beta_3(1) + a_{12} \cdot b_2(O_3) \cdot \beta_3(2) = 0.7 \cdot 0.5 \cdot 1 + 0.3 \cdot 0.1 \cdot 1 = 0.38$$

$$\beta_2(2) = a_{21} \cdot b_1(O_3) \cdot \beta_3(1) + a_{22} \cdot b_2(O_3) \cdot \beta_3(2) = 0.4 \cdot 0.5 \cdot 1 + 0.6 \cdot 0.1 \cdot 1 = 0.26$$



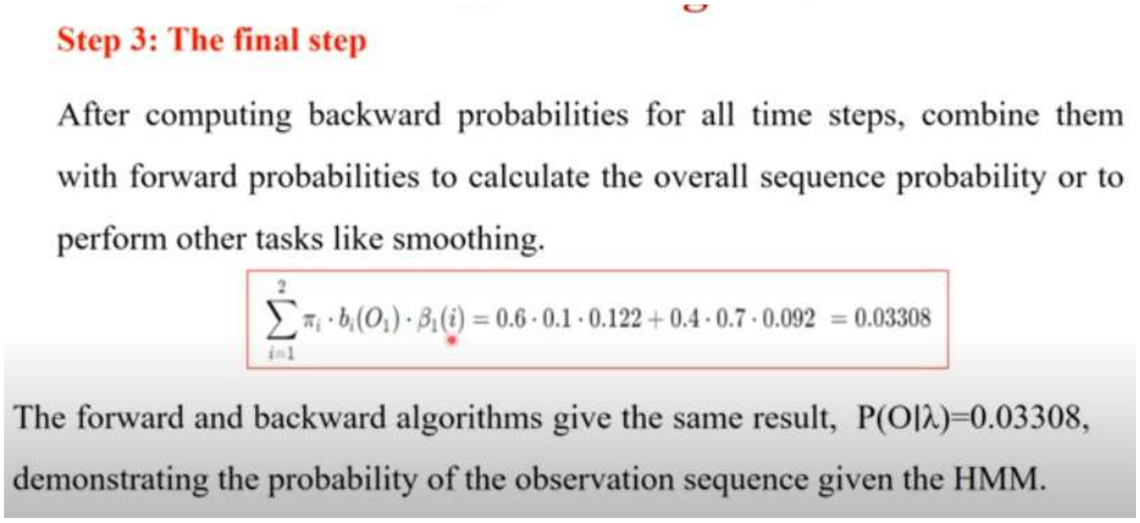
Consider the Observation sequence $O = [O_1, O_2, O_3]$

$$\boxed{\beta_t(i) = \sum_{j=1}^{N} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)}$$

Step 2 (For t=1):

$$\beta_1(1) = a_{11} \cdot b_1(O_2) \cdot \beta_2(1) + a_{12} \cdot b_2(O_2) \cdot \beta_2(2) = 0.7 \cdot 0.4 \cdot 0.38 + 0.3 \cdot 0.2 \cdot 0.26 = 0.122$$

$$\beta_1(2) = a_{21} \cdot b_1(O_2) \cdot \beta_2(1) + a_{22} \cdot b_2(O_2) \cdot \beta_2(2) = 0.4 \cdot 0.4 \cdot 0.38 + 0.6 \cdot 0.2 \cdot 0.26 = 0.092$$



Step 3: The final step

After computing backward probabilities for all time steps, combine them with forward probabilities to calculate the overall sequence probability or to perform other tasks like smoothing.

$$\sum_{i=1}^{2} \pi_i \cdot b_i(O_1) \cdot \beta_1(i) = 0.6 \cdot 0.1 \cdot 0.122 + 0.4 \cdot 0.7 \cdot 0.092 = 0.03308$$

The forward and backward algorithms give the same result, $P(O|\lambda)=0.03308$, demonstrating the probability of the observation sequence given the HMM.

THE SUM-PRODUCT ALGORITHM

The problem can be solved in three steps:

Step 1: Forward Algorithm (Forward Pass)

Step 2: Backward Algorithm (Backward Pass)

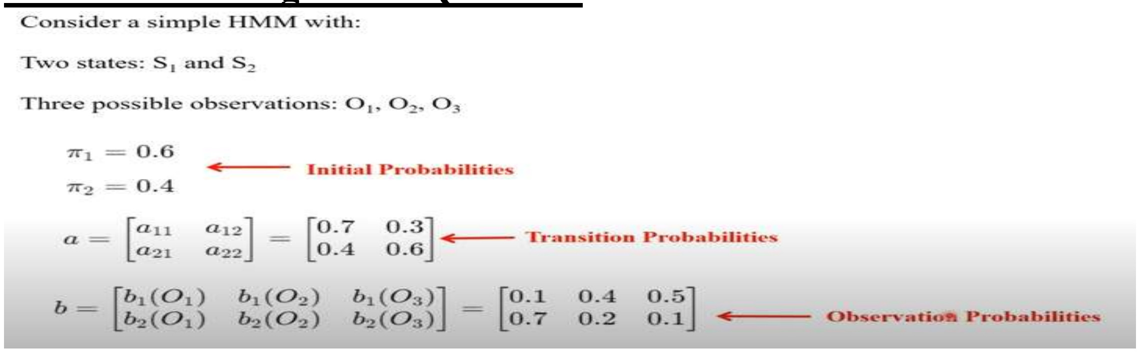
Step 3: Combining Forward and Backward Probabilities

The forward and backward probabilities are combined to compute the posterior probability of particular state at a specific time, given the entire observation sequence.

$\gamma_t(i)$ represents the probability of being in state i at time t, given the entire observation sequence O.

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{P(O|\lambda)}$$

Sum Product Algorithm Question



Consider a simple HMM with:

Two states: S_1 and S_2

Three possible observations: O_1, O_2, O_3

$$\pi_1 = 0.6 \quad \pi_2 = 0.4 \longleftarrow \text{Initial Probabilities}$$

$$a = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \longleftarrow \text{Transition Probabilities}$$

$$b = \begin{bmatrix} b_1(O_1) & b_1(O_2) & b_1(O_3) \\ b_2(O_1) & b_2(O_2) & b_2(O_3) \end{bmatrix} = \begin{bmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{bmatrix} \longleftarrow \text{Observation Probabilities}$$

Step 1: Forward algorithm (Forward Pass)

Recursion at t=3

$$\alpha_3(1) = [0.0616 \cdot 0.7 + 0.0372 \cdot 0.4] \cdot 0.5 = 0.029$$

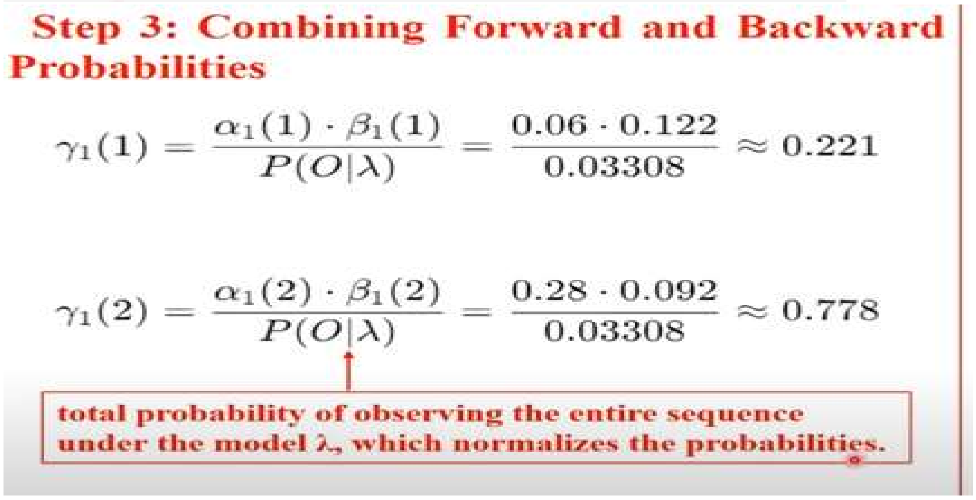
$$\alpha_3(2) = [0.0616 \cdot 0.3 + 0.0372 \cdot 0.6] \cdot 0.1 = 0.00408$$

Step 2: Backward Algorithm (Backward Pass)

Recursion at t=1

$$\beta_1(1) = 0.7 \cdot 0.4 \cdot 0.38 + 0.3 \cdot 0.2 \cdot 0.26 = 0.122$$

$$\beta_1(2) = 0.4 \cdot 0.4 \cdot 0.38 + 0.6 \cdot 0.2 \cdot 0.26 = 0.092$$

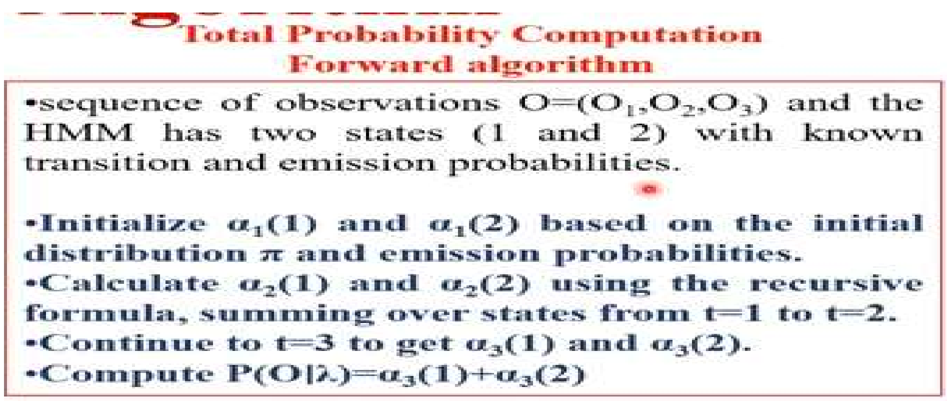


Step 3: Combining Forward and Backward Probabilities

$$\gamma_1(1) = \frac{\alpha_1(1) \cdot \beta_1(1)}{P(O|\lambda)} = \frac{0.06 \cdot 0.122}{0.03308} \approx 0.221$$

$$\gamma_1(2) = \frac{\alpha_1(2) \cdot \beta_1(2)}{P(O|\lambda)} = \frac{0.28 \cdot 0.092}{0.03308} \approx 0.778$$

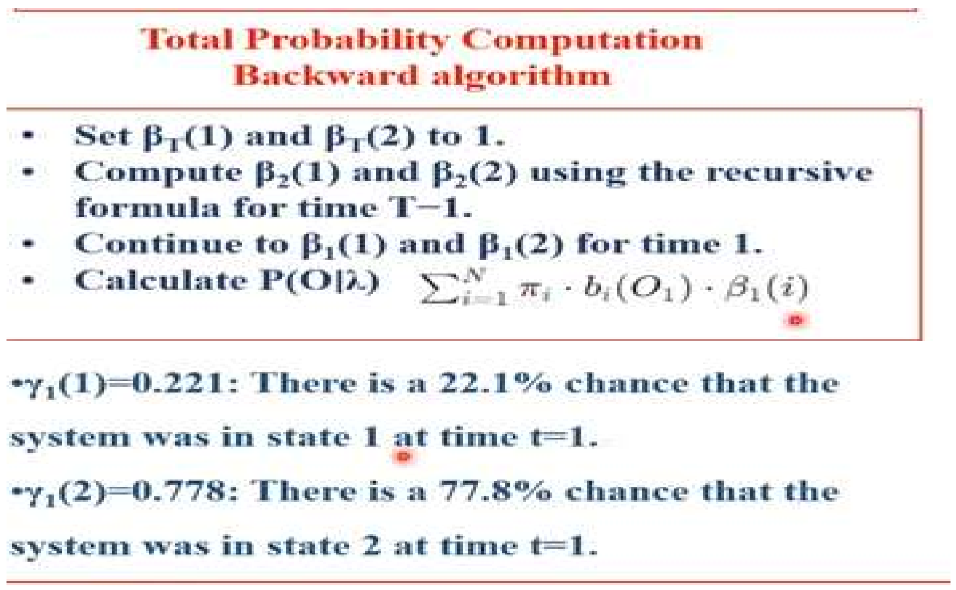
total probability of observing the entire sequence under the model λ, which normalizes the probabilities.



Total Probability Computation
Forward algorithm

- sequence of observations $O=(O_1,O_2,O_3)$ and the HMM has two states (1 and 2) with known transition and emission probabilities.

- **Initialize $\alpha_1(1)$ and $\alpha_1(2)$ based on the initial distribution π and emission probabilities.**
- **Calculate $\alpha_2(1)$ and $\alpha_2(2)$ using the recursive formula, summing over states from t=1 to t=2.**
- **Continue to t=3 to get $\alpha_3(1)$ and $\alpha_3(2)$.**
- **Compute $P(O|\lambda)=\alpha_3(1)+\alpha_3(2)$**



Total Probability Computation
Backward algorithm

- Set $\beta_T(1)$ and $\beta_T(2)$ to 1.
- Compute $\beta_2(1)$ and $\beta_2(2)$ using the recursive formula for time T−1.
- Continue to $\beta_1(1)$ and $\beta_1(2)$ for time 1.
- Calculate P(O|λ) $\sum_{i=1}^{N} \pi_i \cdot b_i(O_1) \cdot \beta_1(i)$

•$\gamma_1(1)=0.221$: There is a 22.1% chance that the system was in state 1 at time t=1.

•$\gamma_1(2)=0.778$: There is a 77.8% chance that the system was in state 2 at time t=1.

THE VITERBI ALGORITHM (VA)

The Viterbi Algorithm was proposed by Andrew J. Viterbi in 1967.
It is a dynamic programming algorithm used to find the most likely sequence of hidden states (or "path") in a hidden Markov model (HMM), given a sequence of observed events.

Input:

- **State Space:** $S=\{s_1, s_2, \dots, s_n\}$
- **Observation sequence:** A sequence of observations (e.g., words in a sentence, acoustic features of a speech signal). $O=\{o_1, o_2, \dots, o_n\}$
- **Hidden states:** A set of possible hidden states (e.g., parts of speech, phonemes).
- **Initial probabilities:** Stores the probability of s_i
- **Transition probabilities**: The probability of transitioning from one hidden state to another. A_{ij} = Probability of transiting from state s_i to state s_j
- **Emission probabilities**: The probability of observing a particular observation given a hidden state. B_{ij} = Probability of transiting from observing o_i from state s_j

Output: The most likely sequence of hidden states.

Steps:

1. **Initialization:**

- Create a trellis (a table) with rows representing the observation sequence and columns representing the hidden states.
- Initialize the first row of the trellis with the probability of starting in each hidden state.

2. **Recursion:**

- For each observation in the sequence:
  - For each hidden state:
    - Calculate the probability of being in that hidden state at that time steps by considering the probabilities of transitioning from previous states and emitting the current observation.

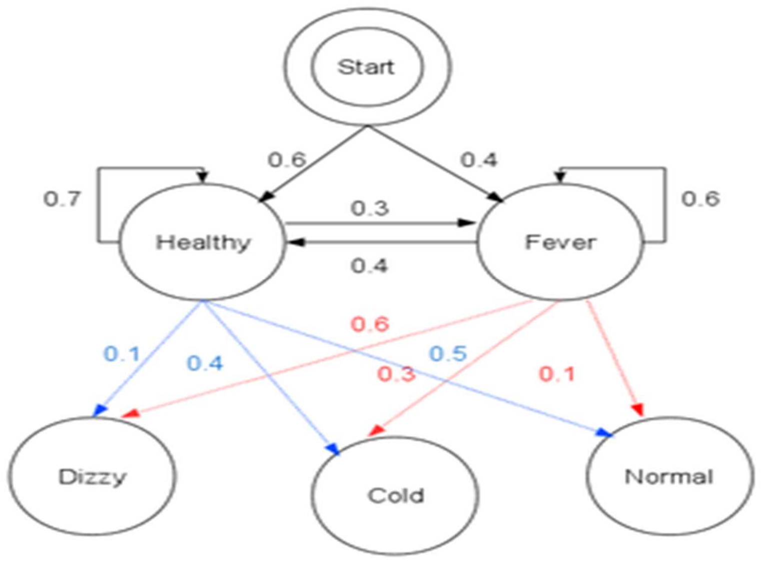
    - Store the calculated probability and the back pointer to the previous state in the trellis.

3. **Termination:**
- Find the most likely final state by selecting the hidden state with the highest probability in the last row of the trellis.
- Trace back the most likely sequence of hidden states using the back pointers.

Advantages:
- Ability to correct wrong bits transmitted by adding redundant information
- State diagram offers a complete description of the System
- Possible to reconstruct lost data

Disadvantages:
- Computations become complex for large number of states
- More bandwidth needed for redundant information

Example:



States:	{‘Healthy’, ‘Fever’}
Observation:	{‘Normal’, ‘Cold’,’Dizzy’}
Initial Probabilities:	P(Healthy)=0.6
	P(Fever)=0.4
Transition Probabilities:	P(Healthy|Healthy)=0.7
	P(Healthy|Fever)=0.4
	P(Fever|Healthy)=0.3
	P(Fever|Fever)=0.6
Emission Probabilities:	P(Dizzy|Healthy)=0.1
	P(Dizzy|Fever)=0.6
	P(Cold|Healthy)=0.4
	P(Cold|Fever)=0.3
	P(Normal|Healthy)=0.5
	P(Normal|Fever)=0.1

A particular patient visits three days in a row, and reports feeling **normal** on the first day, **cold** on the second day, and **dizzy** on the third day.

Observation Sequence: {Normal, Cold, Dizzy}

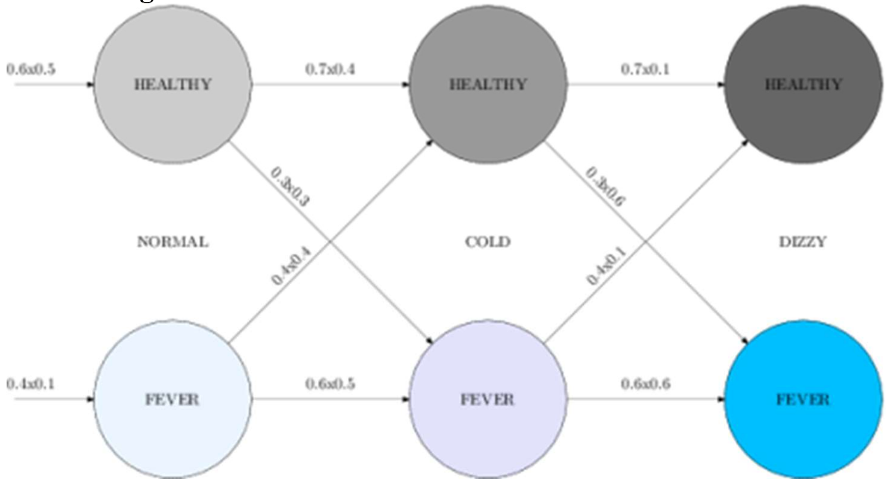
a) The probability that a patient will be healthy on the first day and report feeling normal is 0.6×0.5=0.3.
   The probability that a patient will have a fever on the first day and report feeling normal is 0.4×0.1=0.04.
b) The probabilities for each of the following days can be calculated from the previous day directly:

For example, the highest chance of being **healthy** on the second day and reporting to be **cold**, following reporting being **normal** on the first day, is **the maximum of**

0.3×0.7×0.4=0.084

0.04×0.4×0.4=0.0064

c) **Trellis Diagram**



d) The rest of the probabilities are summarised in the following table:

Day	**1**	**2**	**3**
Observation	Normal	Cold	Dizzy
Healthy	**0.3**	**0.084**	0.00588
Fever	0.04	0.027	**0.01512**

e) From the table, it can be seen that the patient most likely had a fever on the third day. Furthermore, there exists a sequence of states ending on "fever", of which the probability of producing the given observations is 0.**01512**.

SCALING FACTORS

- There is an important ***issue*** that must be addressed before we can make use of the **forward backward algorithm** in practice.
- From the recursion relation:

$$\alpha(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n) \sum_{\mathbf{z}_{n-1}} \alpha(\mathbf{z}_{n-1}) p(\mathbf{z}_n|\mathbf{z}_{n-1}).$$

we note that at each step the new value $\alpha(z_n)$ is obtained from the previous value $\alpha(z_n-1)$ by multiplying by quantities $p(z_n|z_{n-1})$ and $p(x_n|z_n)$.

- These probabilities are often significantly ***less than unity***, as we work our way forward along the chain, the values of $\alpha(z_n)$ can go ***to zero exponentially*** quickly.

- For moderate lengths of chain (say 100 or so), the calculation of the $\alpha(z_n)$ will soon exceed the dynamic range of the computer, even if ***double precision floating point is*** used.
- Simply taking logarithms isn't enough, as we need to compare between sums of small probabilities.
- Store the probabilities **normalised** over the states for a given timestep, keep track of scaling factors.
- **Re-scaled versions of $\alpha(z_n)$ and $\beta(z_n)$:**

$$\widehat{\alpha}(\mathbf{z}_n) = p(\mathbf{z}_n|\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{\alpha(\mathbf{z}_n)}{p(\mathbf{x}_1, \ldots, \mathbf{x}_n)}$$

$$\widehat{\beta}(\mathbf{z}_n) = \frac{p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N|\mathbf{z}_n)}{p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N|\mathbf{x}_1, \ldots, \mathbf{x}_n)}.$$

LINEAR DYNAMICAL SYSTEMS(LDS)

- LDS are a mathematical model used to describe systems that evolve over time according to linear relations between state variables.
- The state of the system at any given time is a hidden (latent) variable, and the observations made from the system are noisy, linear transformations of the state variables. In LDS, the transitions between states and the generation of observations are both linear and Gaussian.
- HMM and LDS are based on **same assumption**: a hidden state variable, of which we can make some noisy measurements.
- HMM uses a ***discrete state variable*** with arbitrary dynamics and arbitrary measurements while LDS uses ***continuous state variable*** with linear Gaussian dynamics and measurements.

LDS Framework:

- **State Equation:**

$$\begin{aligned} \mathbf{s}_t &= \mathbf{A}\mathbf{s}_{t-1} + \mathbf{w}_t \\ \mathbf{w}_t &\sim \mathcal{N}(0, \Gamma) \end{aligned}$$

where:

- s_t is the hidden state at time t,
- A is the state transition matrix,
- W_t is process noise Gaussian

- **Observation Equation:**

$$\begin{aligned} \mathbf{x}_t &= \mathbf{C}\mathbf{s}_t + \mathbf{v}_t \\ \mathbf{v}_t &\sim \mathcal{N}(0, \Sigma) \end{aligned}$$

where:

- y_t is the observation at time t,
- C is the observation matrix,
- $v_t \sim N(0,R)$ is the observation noise (Gaussian).

Inference Problem:

- Because the model is represented by a **tree-structured directed graph**, inference problems can be solved efficiently using the **sum-product algorithm**.
- The ***forward recursions***, analogous to the α messages of the hidden Markov model, are known as the **Kalman filter equations**
- The ***backward recursions***, analogous to the β messages, are known as the **Kalman smoother equations, or the Rauch-Tung-Striebel (RTS) equations**

Forward Recursions Kalman Filter:

The Kalman filter iteratively predicts the hidden state based on the model's dynamics and updates the estimate when new observations become available. It balances between the predicted state and the noisy observations to minimize the estimation error in a probabilistic way.

- **State Prediction**: The predicted state before incorporating the new observation, based on the system's dynamics.
- **Covariance Prediction**: Represents the uncertainty in the predicted state before the new observation.
- **Kalman Gain**: Determines how much the prediction should be adjusted based on the new observation. A larger gain gives more weight to the observation, while a smaller gain gives more weight to the predicted state.
- **Innovation**: Measures how far off the predicted observation is from the actual observation, helping adjust the state estimate.

Backward Recursions Kalman Smoothing:

Kalman smoothing improves upon this by using both past and future observations, leading to more accurate estimates of the hidden states. The smoothing process involves a backward pass after the forward pass to adjust the state estimates by considering future observations.

- **Smoother Gain**: The smoother gain determines how much future information should be used to adjust the filtered state estimate.
- **Smoothed State Estimate**: The smoothed state estimate is a weighted combination of the filtered state estimate and the discrepancy between the smoothed estimate at time t+1 and the predicted state at time t+1.
- **Smoothed Covariance**: The smoothed covariance is reduced compared to the filtered covariance since incorporating future observations typically reduces uncertainty.

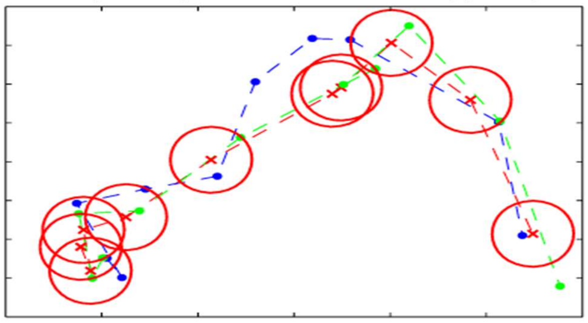


Figure: An illustration of a linear dynamical system being used to track a moving object. The blue points indicate the true positions of the object in a two-dimensional space at successive time steps, the green points denote noisy measurements of the positions, and the red crosses indicate the means of the inferred posterior distributions of the positions obtained by running the Kalman filtering equations.