EE613
Machine Learning for Engineers

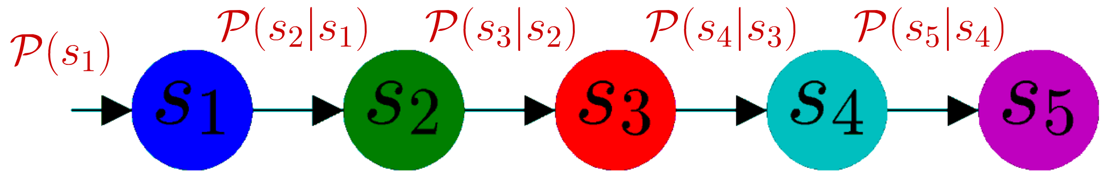
HIDDEN MARKOV MODELS

Sylvain Calinon
Robot Learning & Interaction Group
Idiap Research Institute
Dec. 5, 2019

Outline

- Markov models
- Hidden Markov model (HMM)
- Forward-backward algorithm
- Viterbi decoding (dynamic programming)
- Hidden semi-Markov model (HSMM)
- HMM with dynamic features (Trajectory-HMM)

Markov models



With a **first order Markov model**, the joint distribution of a sequence of states is assumed to be of the form

$$\mathcal{P}(s_1, s_2, \ldots, s_T) = \mathcal{P}(s_1) \prod_{t=2}^{T} \mathcal{P}(s_t|s_{t-1})$$

and we thus have

$$\mathcal{P}(s_t|s_1, s_2, \ldots, s_{t-1}) = \mathcal{P}(s_t|s_{t-1})$$

In most applications, the conditional distributions $\mathcal{P}(s_t|s_{t-1})$ will be assumed to be **stationary** (**homogeneous Markov chain**).

Markov models - Parameters

K *possible states*

The **initial state distribution** is defined by

$$\Pi_i = \mathcal{P}(s_1 = i) \quad \text{with} \quad \sum_{i=1}^{K} \Pi_i = 1$$

A **transition matrix** $\boldsymbol{A}$ is defined, with elements

$$a_{i,j} = \mathcal{P}(s_{t+1} = j \mid s_t = i)$$

defining the probability of getting from state i to state j in one step.

Constraint: each row of the matrix sums to one, $\sum_{j=1}^{K} a_{i,j} = 1$.

Markov models in language modeling

We define the state space to be all the words in English or some other language.

The marginal probabilities $\mathcal{P}(s_t = k)$ are called **unigram** statistics.

For a first-order Markov model, $\mathcal{P}(s_t = k \mid s_{t-1} = j)$ is called a **bigram** model.

For a second-order Markov model, $\mathcal{P}(s_t = k \mid s_{t-1} = j, s_{t-2} = i)$ is called a **trigram** model, etc.

In the general case, these are called ***n*-gram** models.

Markov models in language modeling

Sentence completion
The model can predict the next word given the previous words in a sentence. This can be used to reduce the amount of typing required (e.g., mobile devices).

Data compression
The model can be used to define an encoding scheme, by assigning codewords to more probable strings. The more accurate the predictive model, the fewer the number of bits is required to store the data.

Text classification
The model can be used as a class-conditional density and/or generative classifier.

Automatic writing
The model can be used to sample from $\mathcal{P}(s_1, s_2, \ldots, s_t)$ to generate artificial text.

Markov models in language modeling

SAYS IT’S NOT IN THE CARDS LEGENDARY RECONNAISSANCE BY ROLLIE DEMOCRACIES UNSUSTAINABLE COULD STRIKE REDLINING VISITS TO PROFIT BOOKING WAIT HERE AT MADISON SQUARE GARDEN COUNTY COURTHOUSE WHERE HE HAD BEEN DONE IN THREE ALREADY IN ANY WAY IN WHICH A TEACHER …

Example of text generated from a 4-gram model, trained on a corpus of 400 million words.

The first 4 words are specified by hand, the model generates the 5th word, and then the results are fed back into the model.

Source: http://www.fit.vutbr.cz/~imikolov/rnnlm/gen-4gram.txt

MLE of transition matrix in Markov models

A Markov model is described by $\boldsymbol{\Theta}^{\text{MM}} = \{\{a_{i,j}\}_{j=1}^{K}, \Pi_i\}_{i=1}^{K}$, where the transition probabilities $a_{i,j}$ are stored in a matrix $\boldsymbol{A}$.

The maximum likelihood estimate (MLE) of the parameters can be computed with the normalized counts (details in Appendix)

$$\hat{\Pi}_i = \frac{N_i}{\sum_{k=1}^{K} N_k}, \qquad \hat{a}_{i,j} = \frac{N_{i,j}}{\sum_{k=1}^{K} N_{i,k}}$$

These results can be extended to higher order Markov models, but since an n-gram models has $O(K^n)$ parameters, special care needs to be taken with overfitting.

For example, with a bi-gram model and 50,000 words in the dictionary, there are 2.5 billion parameters to estimate, and it is unlikely that all possible transitions will be observed in the training data.

Hidden Markov model (HMM)

Python notebook: demo_HMM.ipynb

Matlab code: demo_HMM01.m

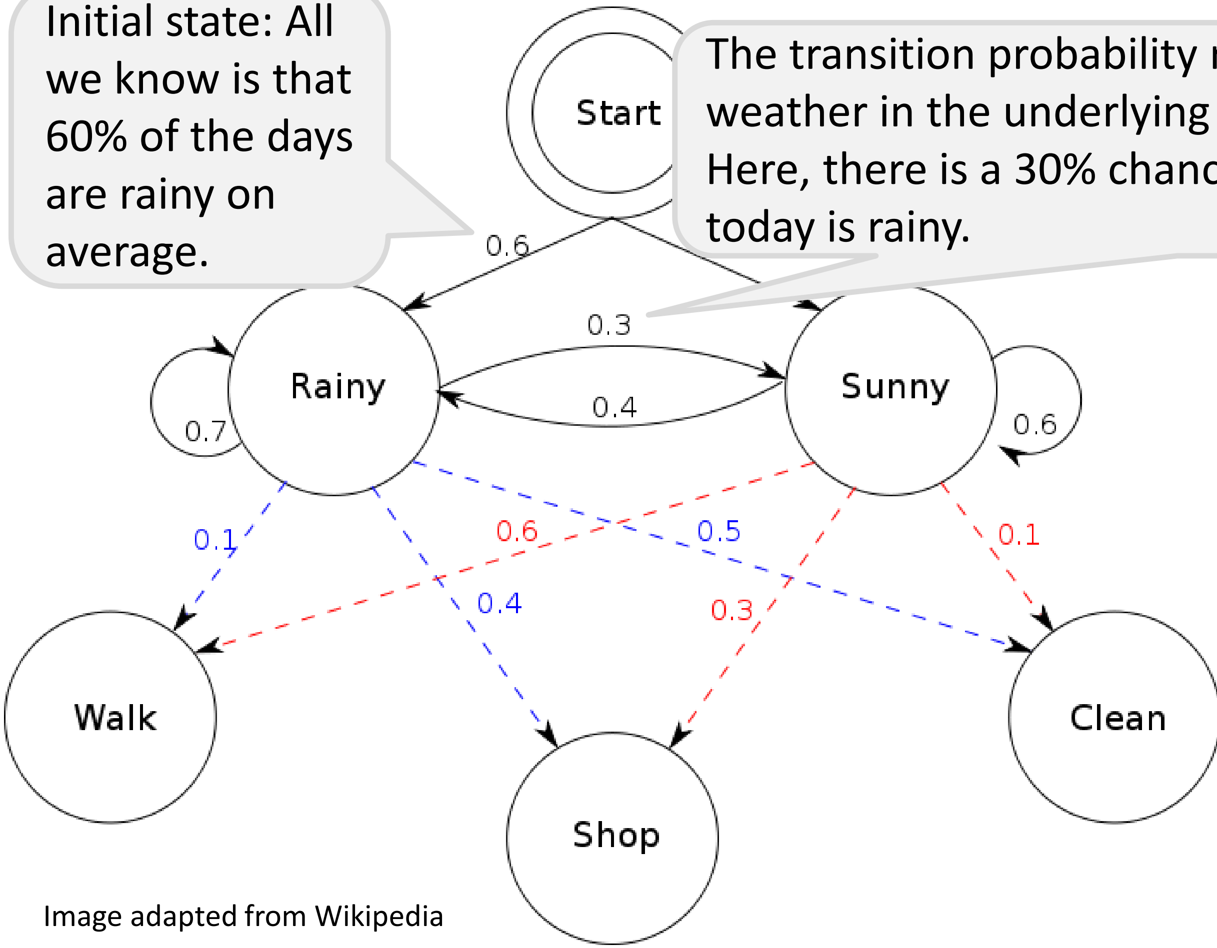
Hidden Markov model (HMM)

In a Markov chain, the state is directly visible to the observer
→ the transition probabilities are the only parameters.

In an HMM, the state is not directly visible, but an output dependent on the state is visible.

Initial state: All we know is that 60% of the days are rainy on average.

The transition probability represents the change of the weather in the underlying Markov chain.
Here, there is a 30% chance that tomorrow will be sunny if today is rainy.



Hidden states

Observed output (emission probability)

Image adapted from Wikipedia

Hidden Markov model (HMM)

You can think of an HMM either as:

- a Markov chain with stochastic measurements
- a GMM with latent <u>variables changing over time</u>

The emission probability represents how likely Bob performs a certain activity on each day.
if it is sunny, there is a 60% chance that he is outside for a walk. If it is rainy, there is a 50% chance that he cleans his apartment, etc.

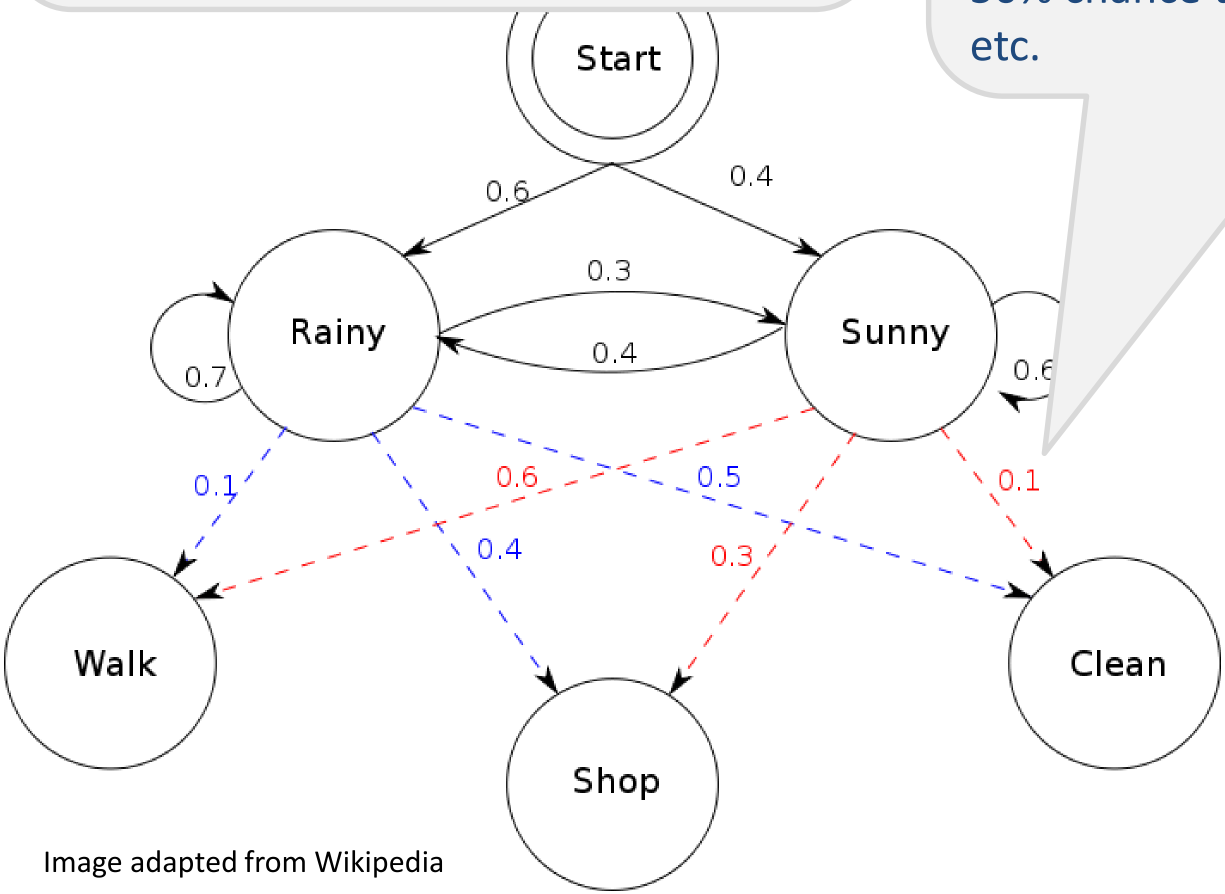


Image adapted from Wikipedia

Hidden states

Observed output (emission probability)

Inference problems associated with HMMs

Probability of an observed sequence

$\mathcal{P}(\boldsymbol{\xi}_{1:T}) = \mathcal{P}(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T) \leftarrow$ Use of *forward* variable

Probability of the latent variables

- **Filtering** $\rightarrow$ Use of *forward* or *backward* variables
  $\mathcal{P}(s_t \,|\, \boldsymbol{\xi}_{1:t}) = \mathcal{P}(s_t \,|\, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t) \leftarrow$ *forward* comp.

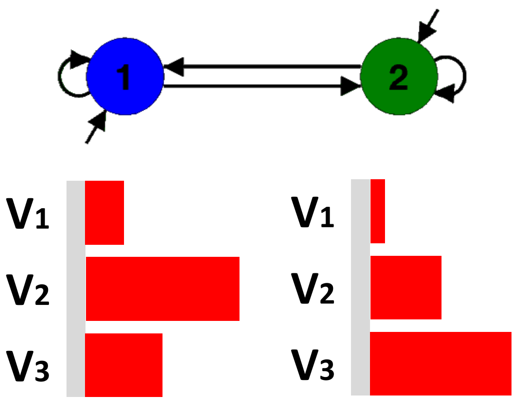
- **Prediction**
  $\mathcal{P}(s_{t+1} \,|\, \boldsymbol{\xi}_{1:t}) = \mathcal{P}(s_{t+1} \,|\, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t) \leftarrow$ *forward* comp.

- **Smoothing** $\rightarrow$ *Forward-backward algorithm*
  $\mathcal{P}(s_t \,|\, \boldsymbol{\xi}_{1:T}) = \mathcal{P}(s_t \,|\, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T)$

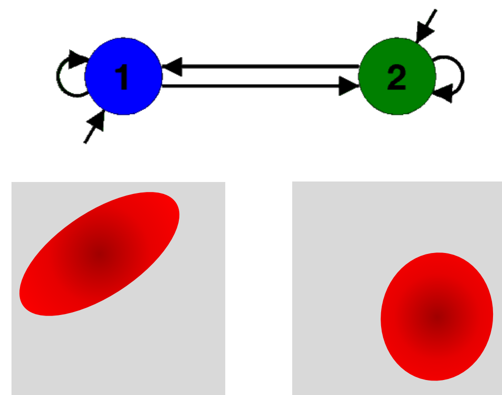
- **MAP estimation** $\rightarrow$ *Viterbi decoding*
  $\mathcal{P}(s_{1:T} \,|\, \boldsymbol{\xi}_{1:T}) = \mathcal{P}(s_1, s_2, \ldots, s_T \,|\, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T)$

Emission/output distributions in HMM

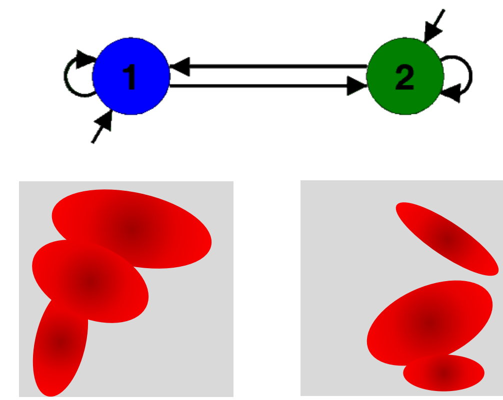
Discrete tables



Gaussian distribution

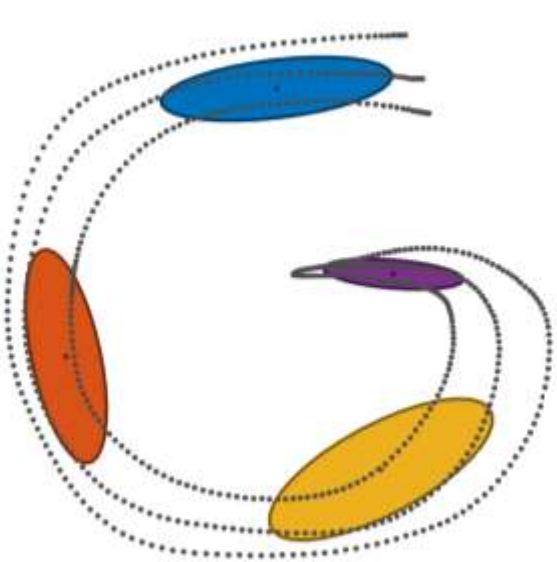


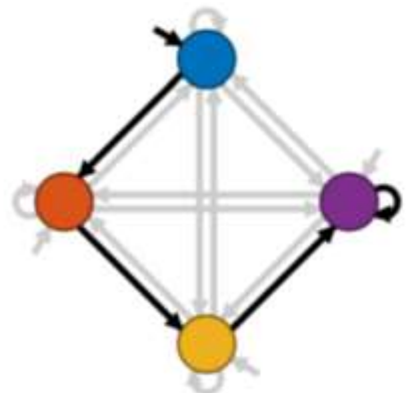
Mixture of Gaussians



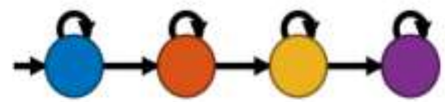
GMM with latent variable $\boldsymbol{z}_t$ depending on the conditional distribution $\mathcal{P}(\boldsymbol{z}_t | \boldsymbol{z}_{t-1})$

Transition matrix structures in HMM

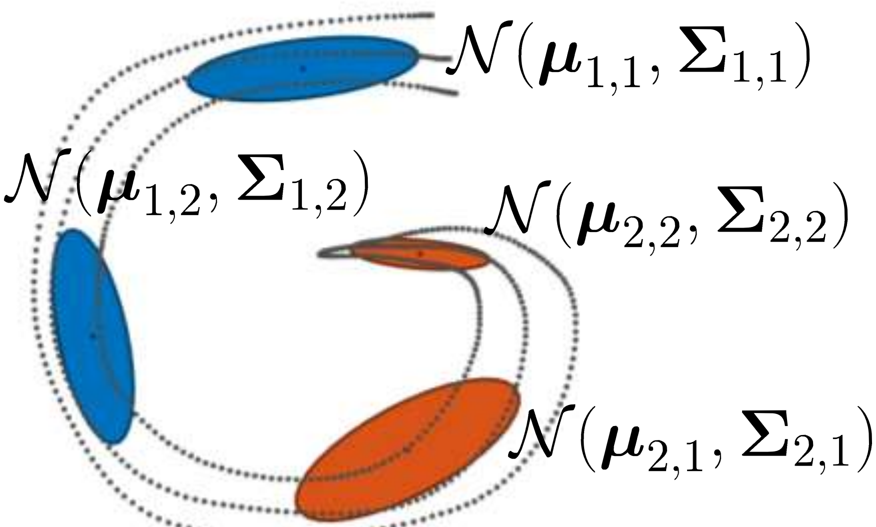




$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix}$$



$$\begin{bmatrix} a_{1,1} & a_{1,2} & 0 & 0 \\ 0 & a_{2,2} & a_{2,3} & 0 \\ 0 & 0 & a_{3,3} & a_{3,4} \\ 0 & 0 & 0 & a_{4,4} \end{bmatrix}$$



$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$$

HMM - Examples of application

HMM is used in many fields as a tool for time series or sequences analysis, and in fields where the goal is to recover a data sequence that is not immediately observable:

Speech recognition
Speech synthesis
Part-of-speech tagging
Natural language modeling
Machine translation
Gene prediction
Molecule kinetic analysis
DNA motif discovery
Alignment of bio-sequences (e.g., proteins)
Metamorphic virus detection
Document separation in scanning solutions

Cryptoanalysis
Activity recognition
Protein folding
Human motion science
Online handwriting recognition
Robotics

and many, many others...

HMM - Examples of application

ξ_t *Observation*
s_t *Hidden state*

Automatic speech recognition

ξ_t can represent features extracted from the speech signal, and s_t can represent the word being spoken. The transition model $P(s_t|s_{t-1})$ represents the language model, and the observation model $P(\xi_t|s_t)$ represents the acoustic model.

Part of speech tagging

ξ_t can represent a word, and s_t represents its part of speech (noun, verb, adjective, etc.)

Activity recognition

ξ_t can represent features extracted from a video frame, and s_t is the class of activity the person is engaged in (e.g., running, walking, sitting, etc.).

Gene finding

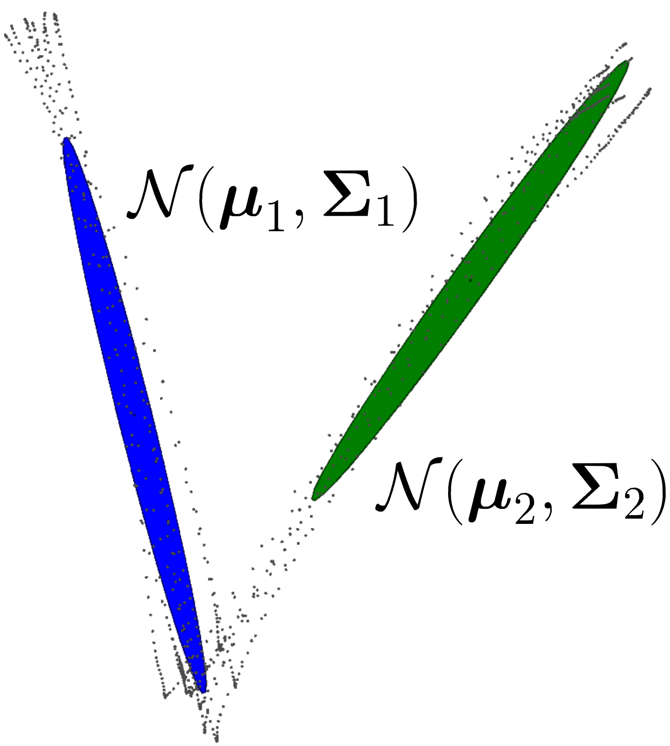
ξ_t can represent the DNA nucleotides (A,T,G,C), and s_t can represent whether we are inside a gene-coding region or not.

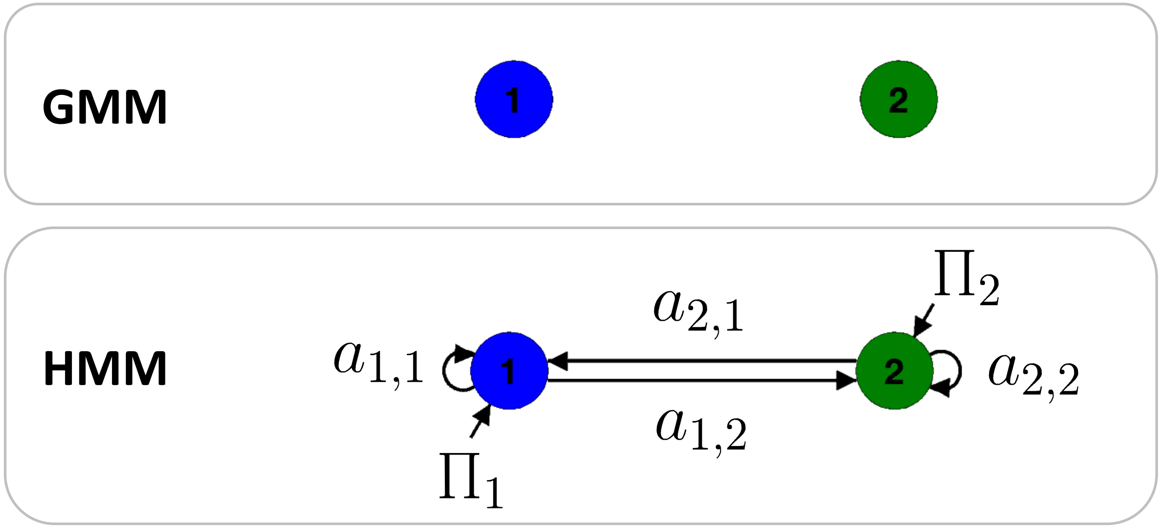
HMM parameters

$$\boldsymbol{\Theta}^{\text{GMM}} = \{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$$

$$\boldsymbol{\Theta}^{\text{HMM}} = \{\{a_{i,j}\}_{j=1}^{K}, \Pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K} \qquad \pi_i = 1$$

From now on, we will consider a single Gaussian as state output





Useful intermediary variables in HMM

Forward variable $\alpha_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

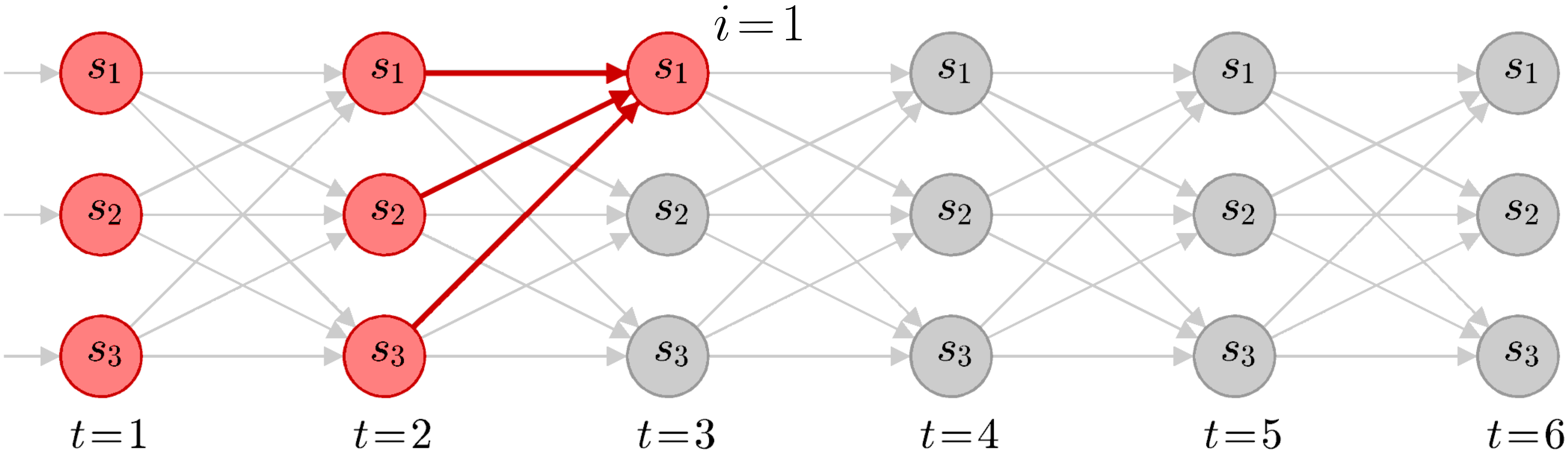
Backward variable $\beta_{t,i}^{\mathrm{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$

Smoothed node marginals $\gamma_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$

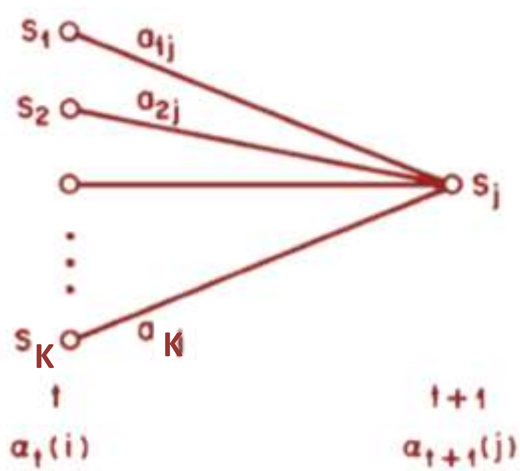
Smoothed edge marginals $\zeta_{t,i,j}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, s_{t+1} = j \mid \boldsymbol{\xi}_{1:T})$

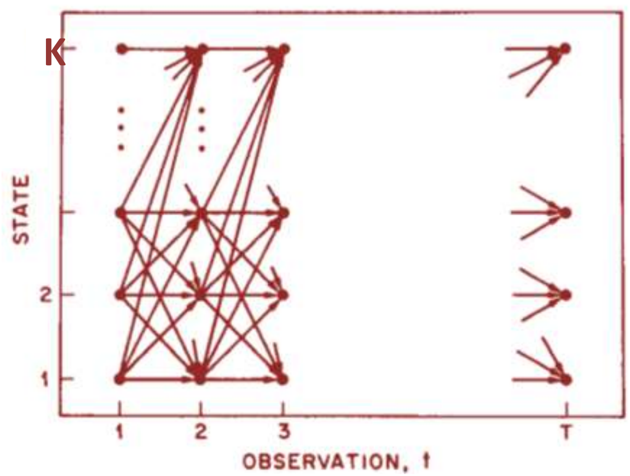
Forward algorithm

$$\alpha^{\text{HMM}}_{t,i} = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})$$



$$\alpha^{\text{HMM}}_{t,i} = \Big(\sum_{j=1}^{K} \alpha^{\text{HMM}}_{t-1,j}\; a_{j,i}\Big)\, \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big) \text{ with } \alpha^{\text{HMM}}_{1,i} = \Pi_i\, \mathcal{N}\big(\boldsymbol{\xi}_1 \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$





Forward algorithm

$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})$

The probability to be in state i at time step t given the partial observation $\boldsymbol{\xi}_{1:t}\!=\!\{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t\}$ can be computed with the **forward variable**

$$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_t) = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})$$

which can be used to compute

$$\mathcal{P}(s_t\!=\!i \,|\, \boldsymbol{\xi}_{1:t}) = \frac{\mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})}{\mathcal{P}(\boldsymbol{\xi}_{1:t})} = \frac{\alpha_{t,i}^{\text{HMM}}}{\sum_{k=1}^{K} \alpha_{t,k}^{\text{HMM}}}$$

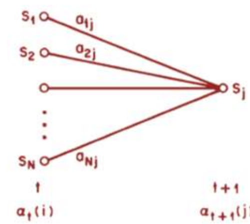
The direct computation would require marginalizing over all possible state sequences $\{s_1, s_2, \ldots, s_{t-1}\}$, which would grow exponentially with t.

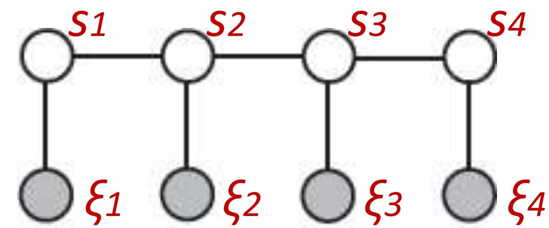
The forward algorithm takes advantage of the conditional independence rules of the HMM to perform the calculation recursively.

Forward algorithm

$\alpha_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

The recursion can be derived by using the chain rule and writing





$$\mathcal{P}(s_t, \boldsymbol{\xi}_{1:t}) = \sum_{s_{t-1}=1}^{K} \mathcal{P}(s_t, s_{t-1}, \boldsymbol{\xi}_{1:t})$$

$\mathcal{P}(a,b) = \mathcal{P}(b|a)\mathcal{P}(a)$

$$= \sum_{s_{t-1}=1}^{K} \mathcal{P}(\boldsymbol{\xi}_t \,|\, s_t, s_{t-1}, \boldsymbol{\xi}_{1:t-1}) \underbrace{\mathcal{P}(s_t, s_{t-1}, \boldsymbol{\xi}_{1:t-1})}_{\mathcal{P}(s_t \,|\, s_{t-1}, \boldsymbol{\xi}_{1:t-1})\, \mathcal{P}(s_{t-1}, \boldsymbol{\xi}_{1:t-1})}$$

Since $\boldsymbol{\xi}_t$ is conditionally dependent only on s_t, and s_t is conditionally dependent only on s_{t-1}, the above relation simplifies to

$$\mathcal{P}(s_t, \boldsymbol{\xi}_{1:t}) = \boxed{\mathcal{P}(\boldsymbol{\xi}_t \,|\, s_t)} \sum_{s_{t-1}=1}^{K} \boxed{\mathcal{P}(s_t \,|\, s_{t-1})} \mathcal{P}(s_{t-1}, \boldsymbol{\xi}_{1:t-1})$$

$\mathcal{P}(\boldsymbol{\xi}_t \,|\, s_t)$ and $\mathcal{P}(s_t \,|\, s_{t-1})$ are the emission and transition probabilities $\rightarrow$ $\mathcal{P}(s_t, \boldsymbol{\xi}_{1:t})$ can be computed from $\mathcal{P}(s_{t-1}, \boldsymbol{\xi}_{1:t-1})$.

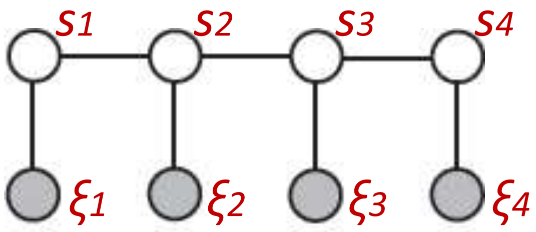
Forward algorithm

$$\alpha^{\text{HMM}}_{t,i} = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})$$

$$\mathcal{P}(s_t, \boldsymbol{\xi}_{1:t}) = \mathcal{P}(\boldsymbol{\xi}_t \,|\, s_t) \sum_{s_{t-1}=1}^{K} \mathcal{P}(s_t \,|\, s_{t-1})\, \mathcal{P}(s_{t-1}, \boldsymbol{\xi}_{1:t-1})$$

The *forward* variable can thus be computed recursively with

$$\alpha^{\text{HMM}}_{t,i} = \Big(\sum_{j=1}^{K} \alpha^{\text{HMM}}_{t-1,j}\; a_{j,i} \Big)\, \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$



by starting from

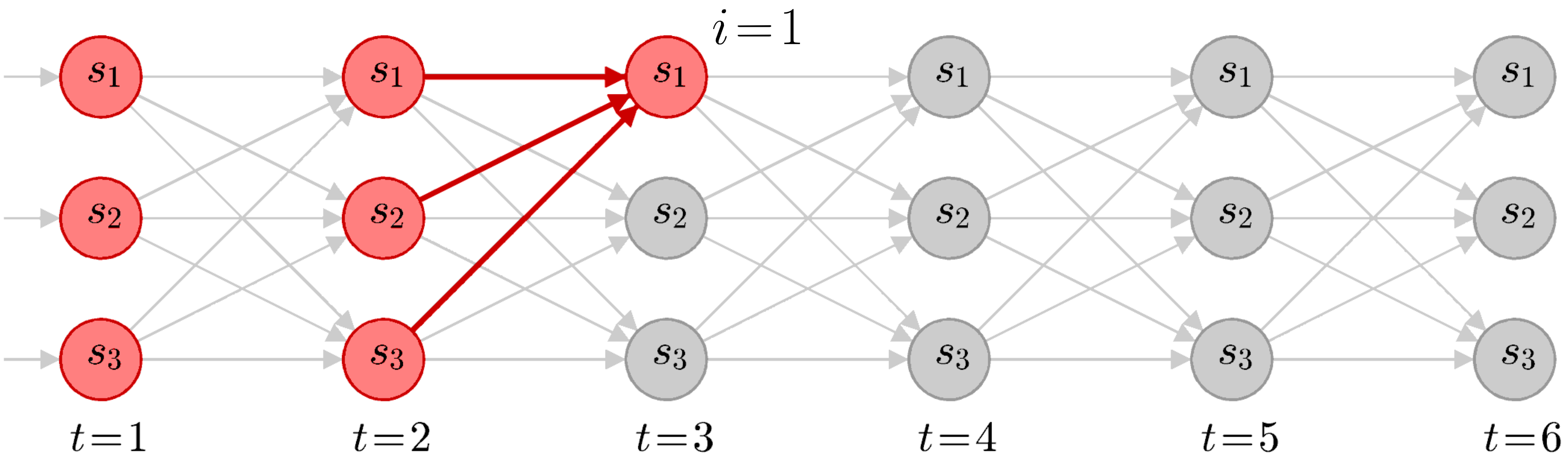
$$\alpha^{\text{HMM}}_{1,i} = \Pi_i\, \mathcal{N}\big(\boldsymbol{\xi}_1 \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

It can be used to evaluate trajectories by computing the likelihood

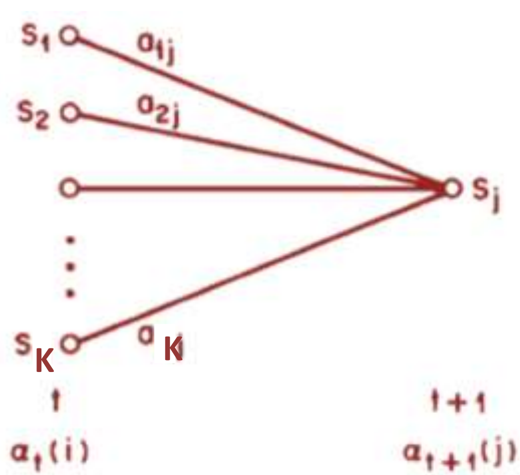
$$\mathcal{P}(\boldsymbol{\xi} \,|\, \boldsymbol{\Theta}^{\text{HMM}}) = \sum_{i=1}^{K} \alpha^{\text{HMM}}_{T,i}$$

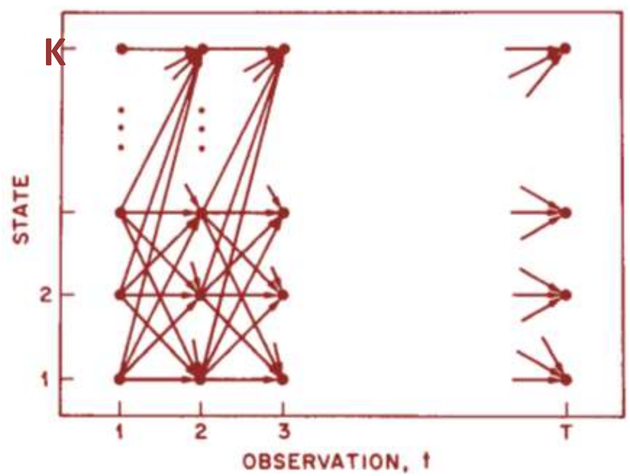
Forward algorithm

$$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$$



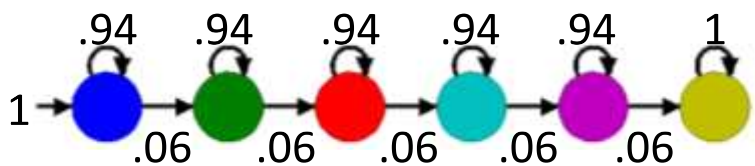
$$\alpha_{t,i}^{\text{HMM}} = \Big(\sum_{j=1}^{K} \alpha_{t-1,j}^{\text{HMM}}\; a_{j,i}\Big)\, \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big) \text{ with } \alpha_{1,i}^{\text{HMM}} = \Pi_i\, \mathcal{N}\big(\boldsymbol{\xi}_1 \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$





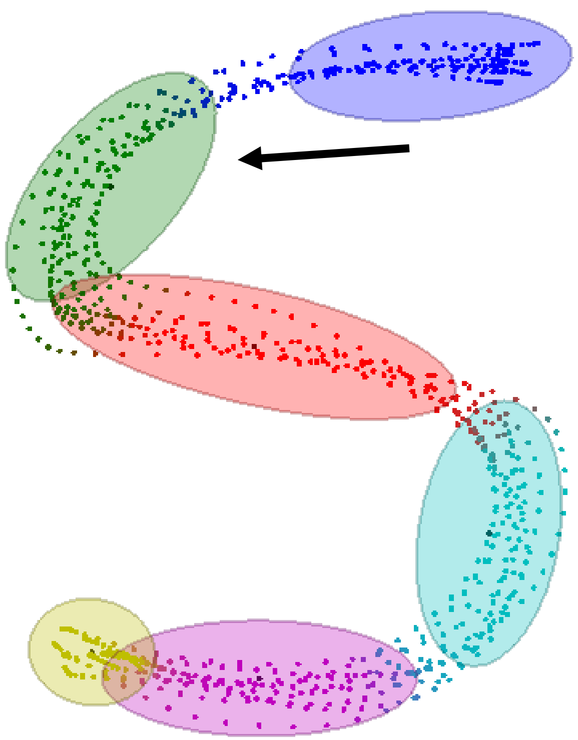
Low influence of transition probabilities w.r.t. emission probabilities in HMM

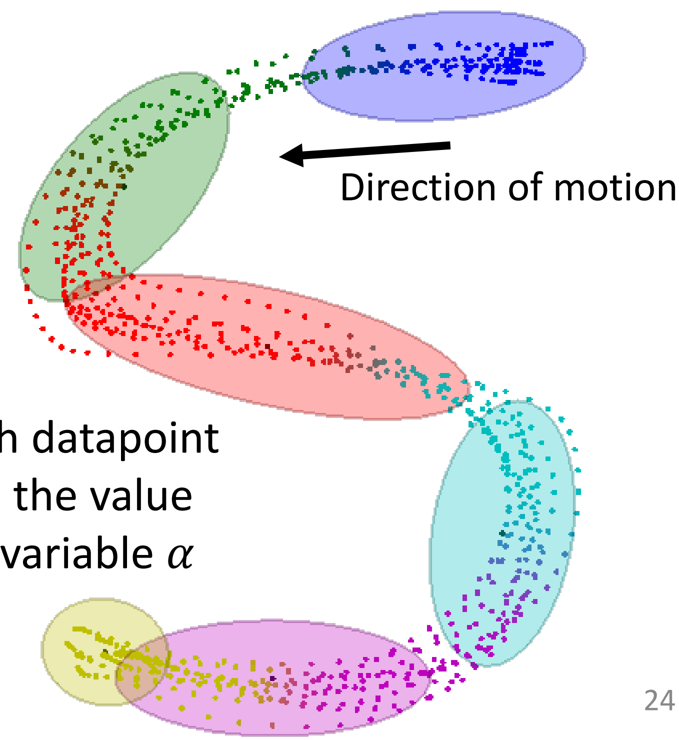
$$\alpha_{t,i}^{\text{HMM}} = \Big(\sum_{j=1}^{K} \alpha_{t-1,j}^{\text{HMM}} \; a_{j,i} \Big) \, \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$



Learned transition probabilities

Transition probabilities manually set





The color of each datapoint corresponds to the value of the forward variable α

Useful intermediary variables in HMM

Forward variable $\alpha_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

Backward variable $\beta_{t,i}^{\mathrm{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$

Smoothed node marginals $\gamma_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$

Smoothed edge marginals $\zeta_{t,i,j}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, s_{t+1} = j \mid \boldsymbol{\xi}_{1:T})$

Backward algorithm

$\beta^{\text{HMM}}_{t,i} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t = i)$

Similarly, we can define a **backward variable** starting from

$$\beta^{\text{HMM}}_{T,i} = 1$$

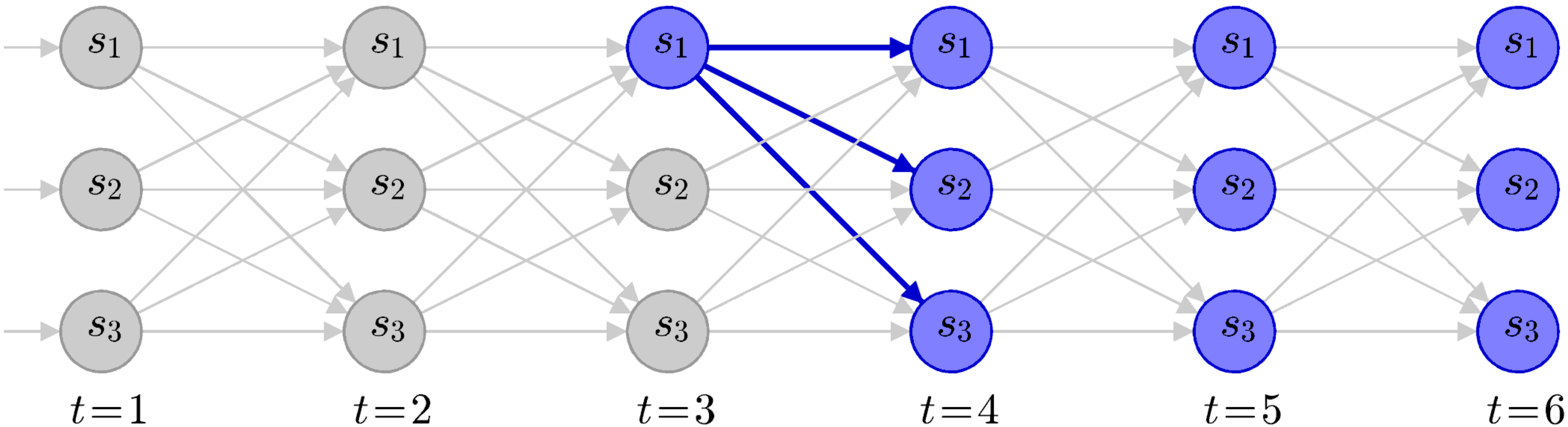
and computed as

$$\beta^{\text{HMM}}_{t,i} = \sum_{j=1}^{K} a_{i,j}\, \mathcal{N}\big(\boldsymbol{\xi}_{t+1} |\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\big)\; \beta^{\text{HMM}}_{t+1,j}$$

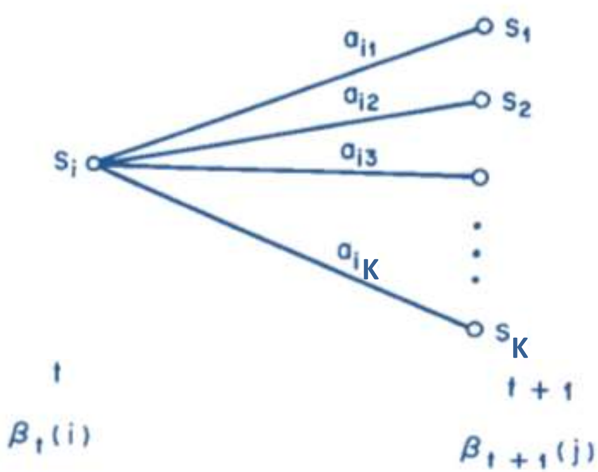
corresponding to the probability of the partial observation $\{\boldsymbol{\xi}_{t+1}, \ldots, \boldsymbol{\xi}_{T-1}, \boldsymbol{\xi}_T\}$, knowing that we are in state i at time step t.

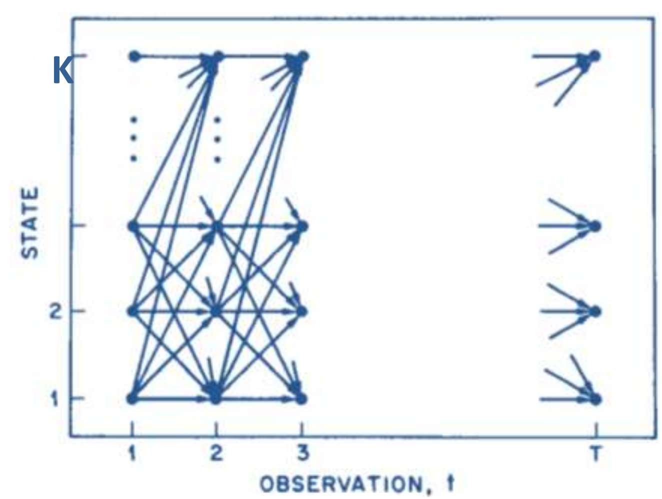
Backward algorithm

$$\beta^{\text{HMM}}_{t,i} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t = i)$$



$$\beta^{\text{HMM}}_{t,i} = \sum_{j=1}^{K} a_{i,j}\, \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\big)\; \beta^{\text{HMM}}_{t+1,j} \quad \text{with} \quad \beta^{\text{HMM}}_{T,i} = 1$$





Useful intermediary variables in HMM

Forward variable $\alpha^{\text{HMM}}_{t,i} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

Backward variable $\beta^{\text{HMM}}_{t,i} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t = i)$

Smoothed node marginals $\gamma^{\text{HMM}}_{t,i} = \mathcal{P}(s_t = i \,|\, \boldsymbol{\xi}_{1:T})$

Smoothed edge marginals $\zeta^{\text{HMM}}_{t,i,j} = \mathcal{P}(s_t = i, s_{t+1} = j \,|\, \boldsymbol{\xi}_{1:T})$

These variable are sometimes called "smoothed values" as they combine forward and backward probabilities in the computation.

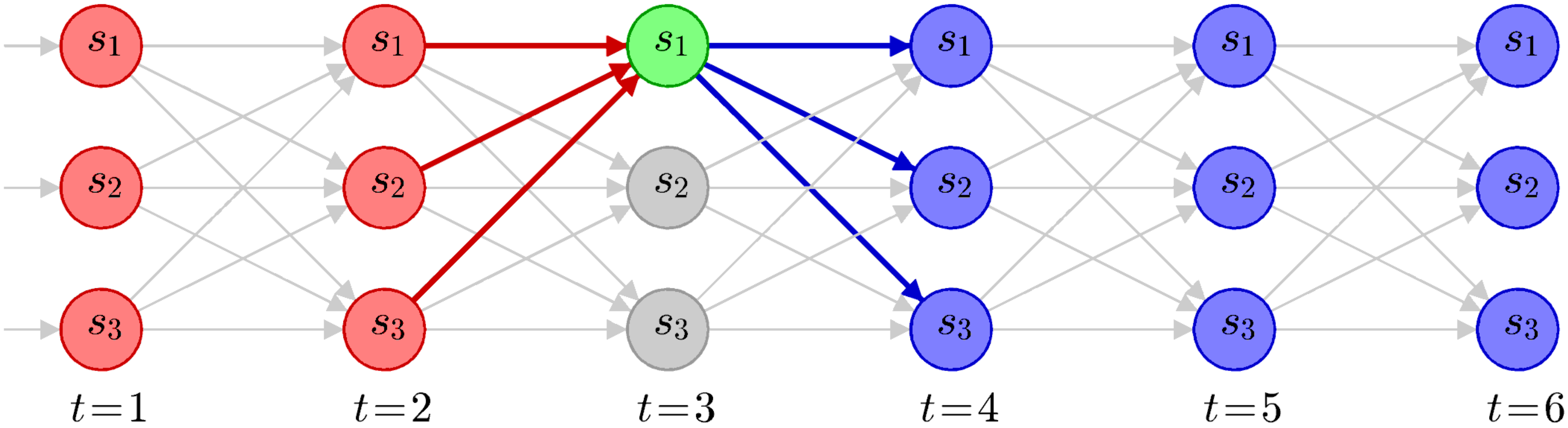
You can think of their roles as passing "messages" from left to right, and from right to left, and then combining the information at each node.

Smoothed node marginals

$$\gamma^{\text{HMM}}_{t,i} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$$

$$\alpha^{\text{HMM}}_{t,i} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$$

$$\beta^{\text{HMM}}_{t,i} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$$



$$\gamma^{\text{HMM}}_{t,i} = \frac{\alpha^{\text{HMM}}_{t,i}\ \beta^{\text{HMM}}_{t,i}}{\sum_{k=1}^{K} \alpha^{\text{HMM}}_{t,k}\ \beta^{\text{HMM}}_{t,k}} = \frac{\alpha^{\text{HMM}}_{t,i}\ \beta^{\text{HMM}}_{t,i}}{\mathcal{P}(\boldsymbol{\xi})}$$

Smoothed node marginals

$\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i \,|\, \boldsymbol{\xi}_{1:T})$

Given the full observation $\boldsymbol{\xi} = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T\}$, the probability of $\boldsymbol{\xi}_t$ to be in state i at time step t is

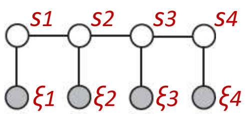
$$\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i \,|\, \boldsymbol{\xi}_{1:T}) = \frac{\mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:T})}{\mathcal{P}(\boldsymbol{\xi}_{1:T})}$$

$\mathcal{P}(b|a) = \frac{\mathcal{P}(a,b)}{\mathcal{P}(a)}$

$\mathcal{P}(a,b) = \mathcal{P}(b|a)\mathcal{P}(a)$

$$= \frac{\mathcal{P}(\boldsymbol{\xi}_{1:T} \,|\, s_t\!=\!i)\ \mathcal{P}(s_t\!=\!i)}{\mathcal{P}(\boldsymbol{\xi}_{1:T})}$$

Conditional independence property



$$= \frac{\mathcal{P}(\boldsymbol{\xi}_{1:t} \,|\, s_t\!=\!i)\ \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t\!=\!i)\ \mathcal{P}(s_t\!=\!i)}{\mathcal{P}(\boldsymbol{\xi}_{1:T})}$$

$\mathcal{P}(a,b) = \mathcal{P}(b|a)\mathcal{P}(a)$

$$= \frac{\mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})\ \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t\!=\!i)}{\mathcal{P}(\boldsymbol{\xi}_{1:T})}$$

$$= \frac{\alpha_{t,i}^{\text{HMM}}\ \beta_{t,i}^{\text{HMM}}}{\sum_{k=1}^{K} \alpha_{t,k}^{\text{HMM}}\ \beta_{t,k}^{\text{HMM}}}$$

$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i, \boldsymbol{\xi}_{1:t})$

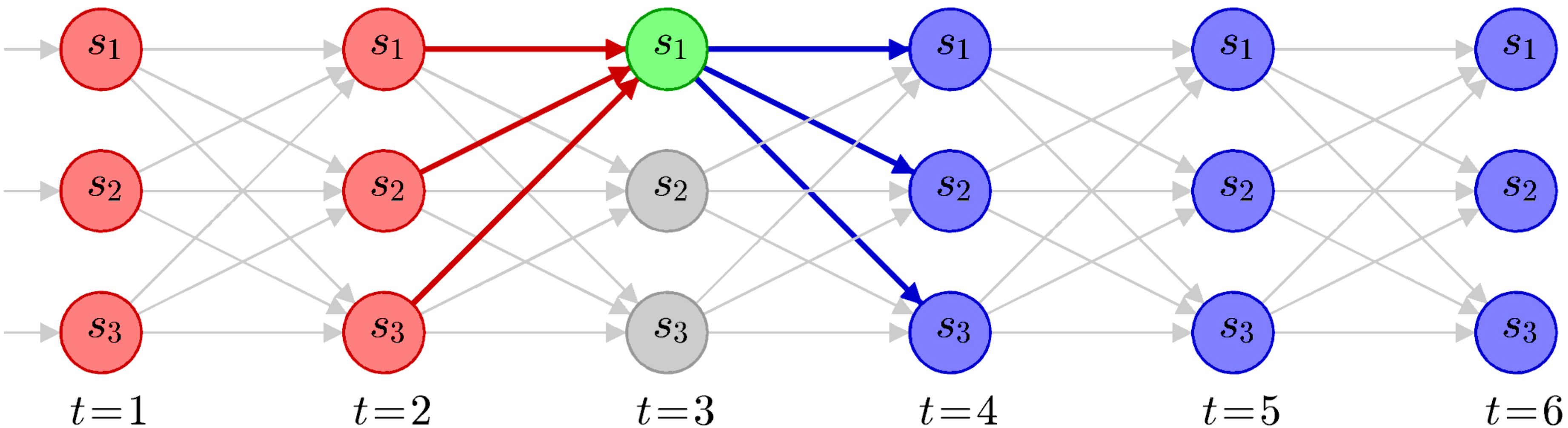
$\beta_{t,i}^{\text{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t\!=\!i)$

Smoothed node marginals

$$\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$$

$$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$$

$$\beta_{t,i}^{\text{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$$



$$\gamma_{t,i}^{\text{HMM}} = \frac{\alpha_{t,i}^{\text{HMM}}\ \beta_{t,i}^{\text{HMM}}}{\sum_{k=1}^{K} \alpha_{t,k}^{\text{HMM}}\ \beta_{t,k}^{\text{HMM}}} = \frac{\alpha_{t,i}^{\text{HMM}}\ \beta_{t,i}^{\text{HMM}}}{\mathcal{P}(\boldsymbol{\xi})}$$

Useful intermediary variables in HMM

Forward variable $\alpha_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

Backward variable $\beta_{t,i}^{\mathrm{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \,|\, s_t = i)$

Smoothed node marginals $\gamma_{t,i}^{\mathrm{HMM}} = \mathcal{P}(s_t = i \,|\, \boldsymbol{\xi}_{1:T})$

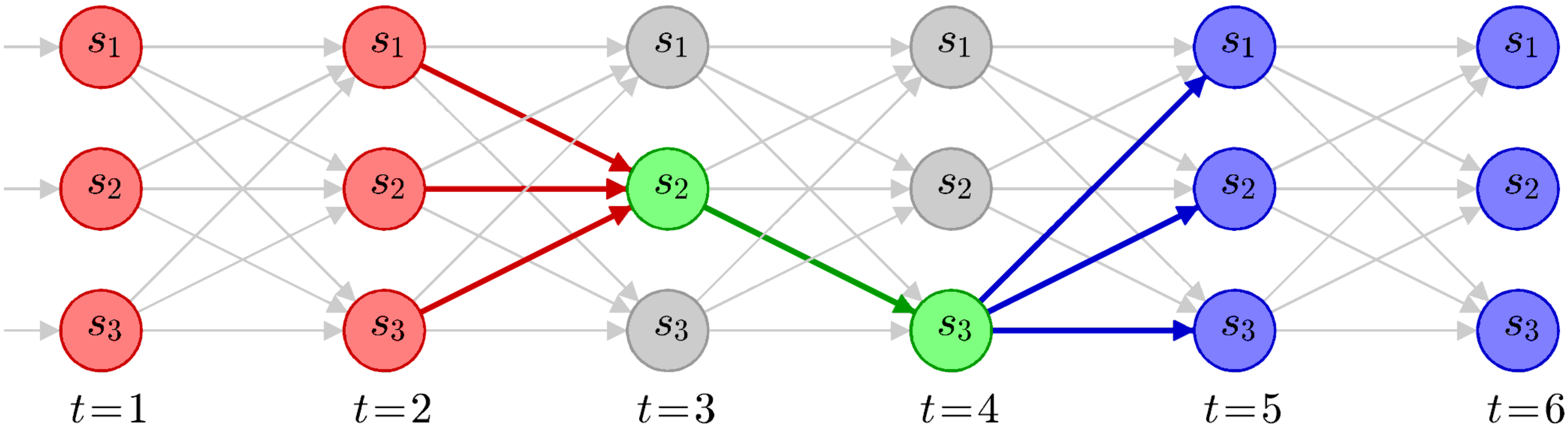
Smoothed edge marginals $\zeta_{t,i,j}^{\mathrm{HMM}} = \mathcal{P}(s_t = i, s_{t+1} = j \,|\, \boldsymbol{\xi}_{1:T})$

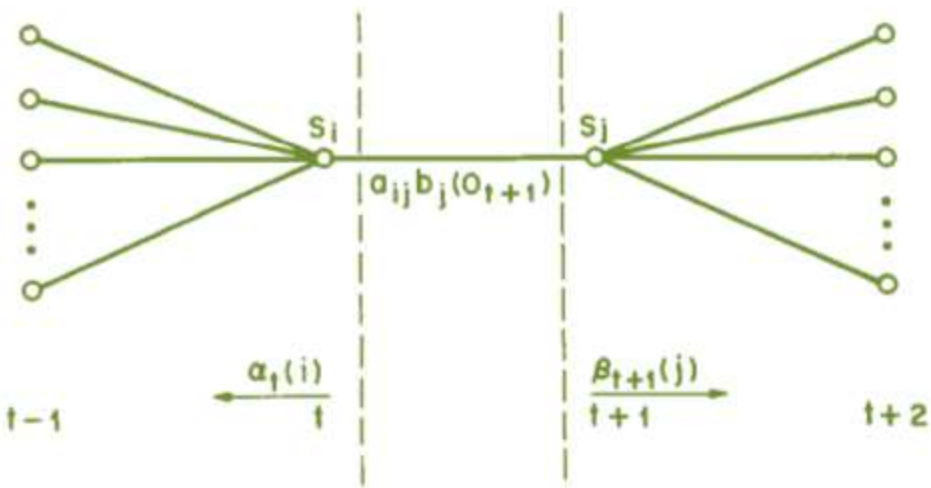
These variable are sometimes called "smoothed values" as they combine forward and backward probabilities in the computation.

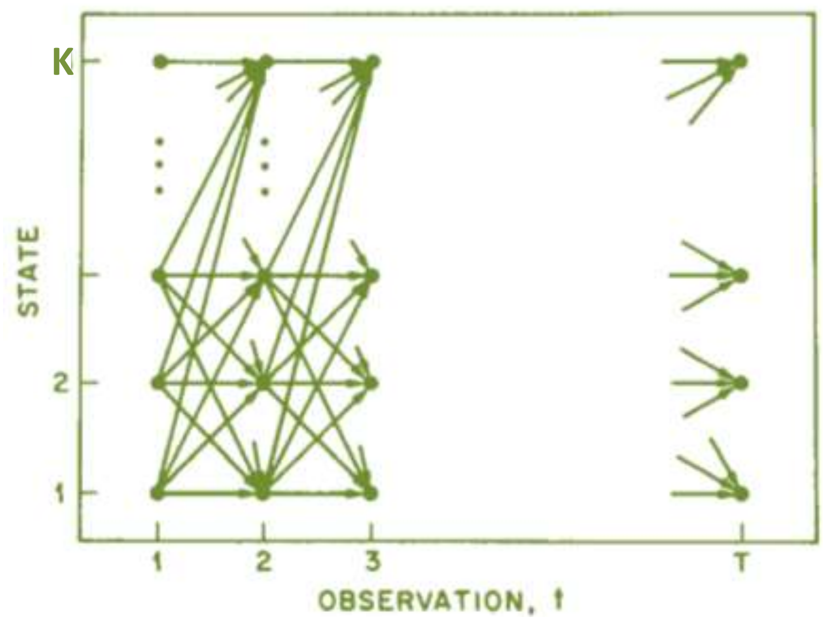
You can think of their roles as passing "messages" from left to right, and from right to left, and then combining the information at each node.

Smoothed edge marginals

$$\zeta^{\mathrm{HMM}}_{t,i,j} = \mathcal{P}(s_t = i, s_{t+1} = j \mid \boldsymbol{\xi}_{1:T})$$







Smoothed edge marginals $\zeta^{\text{HMM}}_{t,i,j} = \mathcal{P}(s_t{=}i, s_{t+1}{=}j \,|\, \boldsymbol{\xi}_{1:T})$

Given the full observation $\boldsymbol{\xi} = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_T\}$, the probability to be in state i at time step t and in state j at time step $t{+}1$ is

$$\zeta^{\text{HMM}}_{t,i,j} = \mathcal{P}\big(s_t = i, s_{t+1} = j \,\big|\, \boldsymbol{\xi}_{1:T}\big)$$

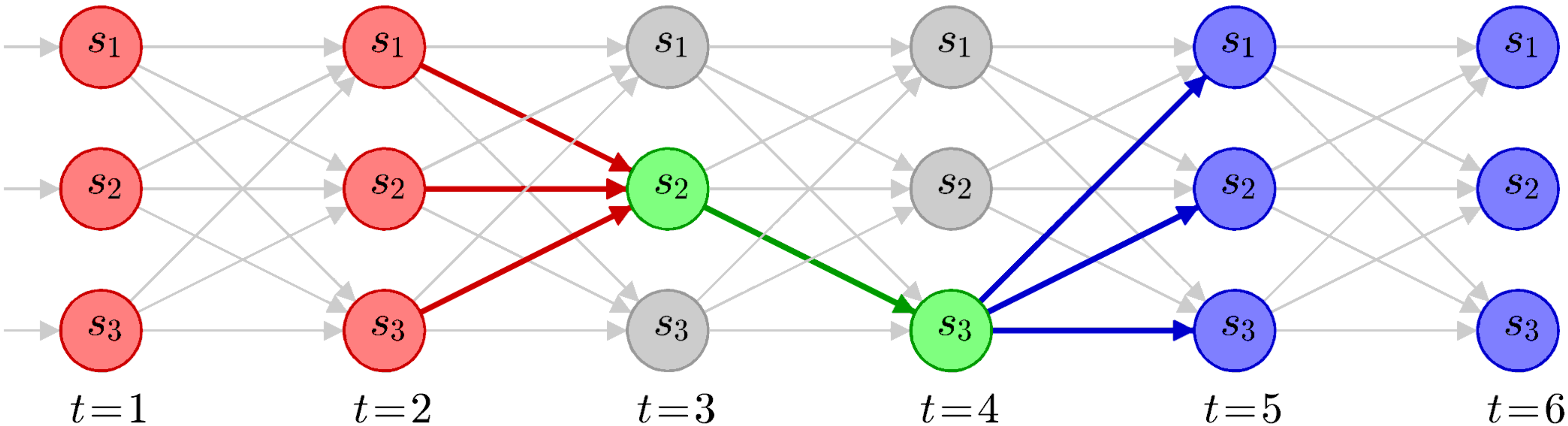
$$= \frac{\mathcal{P}\big(s_t = i, s_{t+1} = j, \boldsymbol{\xi}_{1:T}\big)}{\mathcal{P}\big(\boldsymbol{\xi}_{1:T}\big)}$$

$\mathcal{P}(b|a) = \frac{\mathcal{P}(a,b)}{\mathcal{P}(a)}$

$$= \frac{\alpha^{\text{HMM}}_{t,i}\; a_{i,j}\; \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\big)\; \beta^{\text{HMM}}_{t+1,j}}{\sum_{k=1}^{K} \sum_{l=1}^{K} \alpha^{\text{HMM}}_{t,k}\; a_{k,l}\; \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\big)\; \beta^{\text{HMM}}_{t+1,l}}$$

Note that we have $\gamma^{\text{HMM}}_{t,i} = \sum_{j=1}^{K} \zeta^{\text{HMM}}_{t,i,j}$

Smoothed edge marginals $\zeta^{\text{HMM}}_{t,i,j} = \mathcal{P}(s_t{=}i, s_{t+1}{=}j \,|\, \boldsymbol{\xi}_{1:T})$



$$\zeta^{\text{HMM}}_{t,i,j} = \frac{\alpha^{\text{HMM}}_{t,i}\; a_{i,j}\; \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\big)\; \beta^{\text{HMM}}_{t+1,j}}{\sum_{k=1}^{K}\sum_{l=1}^{K} \alpha^{\text{HMM}}_{t,k}\; a_{k,l}\; \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\big)\; \beta^{\text{HMM}}_{t+1,l}}$$

$$= \frac{\alpha^{\text{HMM}}_{t,i}\; a_{i,j}\; \mathcal{N}\big(\boldsymbol{\xi}_{t+1} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\big)\; \beta^{\text{HMM}}_{t+1,j}}{\mathcal{P}(\boldsymbol{\xi})}$$

EM for HMM

K Gaussians
M trajectories
T_m points per traj.

The expected complete data log-likelihood is

$$\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}}) = \mathbb{E}\left[\sum_{m=1}^{M} \sum_{t=1}^{T_m} \log \mathcal{P}(\boldsymbol{\xi}_{m,t}, \boldsymbol{s}_t \,|\, \boldsymbol{\Theta}) \,\middle|\, \boldsymbol{\xi}, \boldsymbol{\Theta}^{\text{old}} \right]$$

$$= \boxed{\sum_{i=1}^{K} \mathbb{E}[N_i] \log \Pi_i + \sum_{i=1}^{K} \sum_{j=1}^{K} \mathbb{E}[N_{i,j}] \log a_{i,j}}$$

$$\boxed{+ \sum_{m=1}^{M} \sum_{t=1}^{T_m} \sum_{i=1}^{K} \mathcal{P}(s_t \!=\! i \,|\, \boldsymbol{\xi}_m, \boldsymbol{\Theta}^{\text{old}}) \; \log \mathcal{N}(\boldsymbol{\xi}_{m,t} | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}$$

Similar to Markov models

Similar to GMM

with expected counts given by

$$\mathbb{E}[N_i] = \sum_{m=1}^{M} \mathcal{P}(s_{m,1} \!=\! i \,|\, \boldsymbol{\xi}_m, \boldsymbol{\Theta}^{\text{old}}) = \sum_{m=1}^{M} \gamma_{m,1,i}^{\text{HMM}}$$

$$\mathbb{E}[N_{i,j}] = \sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \mathcal{P}(s_{m,t} \!=\! i, s_{m,t+1} \!=\! j \,|\, \boldsymbol{\xi}_m, \boldsymbol{\Theta}^{\text{old}}) = \sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \zeta_{m,t,i,j}^{\text{HMM}}$$

EM for HMM

K Gaussians
M trajectories
T_m points per traj.

By setting

$$\frac{\partial \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}})}{\partial \Pi_i} = 0 \qquad \frac{\partial \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}})}{\partial a_{i,j}} = 0$$

a result similar to the case of Markov models is obtained.

The maximum likelihood estimate of the initial state distribution and transition probability parameters can thus be computed with the normalized counts

$$\hat{\Pi}_i = \frac{\mathbb{E}[N_i]}{\sum_{k=1}^{K} \mathbb{E}[N_k]} = \frac{\mathbb{E}[N_i]}{M}, \qquad \hat{a}_{i,j} = \frac{\mathbb{E}[N_{i,j}]}{\sum_{k=1}^{K} \mathbb{E}[N_{i,k}]} = \frac{\mathbb{E}[N_{i,j}]}{\mathbb{E}[N_i]}$$

with $$\mathbb{E}[N_i] = \sum_{m=1}^{M} \sum_{t=1}^{T_m} \mathcal{P}(s_{m,t} = i \,|\, \boldsymbol{\xi}_m, \boldsymbol{\Theta}^{\text{old}}) = \sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}}$$

EM for HMM - Summary

K Gaussians
M trajectories
T_m points per traj.

M-step:

$$\Pi_i \leftarrow \frac{\sum_{m=1}^{M} \gamma_{m,1,i}^{\text{HMM}}}{M} = \frac{\text{Total number of times in i at time step 1}}{\text{Total number of trajectories}}$$

$$a_{i,j} \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \zeta_{m,t,i,j}^{\text{HMM}}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \gamma_{m,t,i}^{\text{HMM}}} = \frac{\text{Total number of transitions from i to j}}{\text{Total number of times in i (and transit to anything else)}}$$

$$\boldsymbol{\mu}_i \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}} \, \boldsymbol{\xi}_{m,t}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}}}$$

By setting $\frac{\partial \mathcal{Q}(\boldsymbol{\Theta},\boldsymbol{\Theta}^{\text{old}})}{\partial \boldsymbol{\mu}_i} = 0$ and $\frac{\partial \mathcal{Q}(\boldsymbol{\Theta},\boldsymbol{\Theta}^{\text{old}})}{\partial \boldsymbol{\Sigma}_i} = 0$, a result similar to GMM is obtained.

$$\boldsymbol{\Sigma}_i \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}} \, (\boldsymbol{\xi}_{m,t} - \boldsymbol{\mu}_i)(\boldsymbol{\xi}_{m,t} - \boldsymbol{\mu}_i)^{\top}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}}}$$

EM for HMM - Summary

K Gaussians
M trajectories
T_m points per traj.

M-step:

$$\Pi_i \leftarrow \frac{\sum_{m=1}^{M} \gamma_{m,1,i}^{\text{HMM}}}{M}$$

$$a_{i,j} \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \zeta_{m,t,i,j}^{\text{HMM}}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m-1} \gamma_{m,t,i}^{\text{HMM}}}$$

$$\boldsymbol{\mu}_i \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}} \boldsymbol{\xi}_{m,t}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}}}$$

$$\boldsymbol{\Sigma}_i \leftarrow \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}} (\boldsymbol{\xi}_{m,t} - \boldsymbol{\mu}_i)(\boldsymbol{\xi}_{m,t} - \boldsymbol{\mu}_i)^{\top}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_{m,t,i}^{\text{HMM}}}$$

These results can be formally retrieved with EM (also called **Baum-Welch algorithm** in the context of HMM).

The update rules can be interpreted as normalized counts, with several types of weighted averages required in the computation.

Numerical underflow issue in HMM

For long sequences, the forward and backward variables can quickly get very low, likely exceeding the precision range of the computer.

A simple scaling procedure is to multiply $\alpha_{t,i}^{\text{HMM}}$ by a factor independent of i, and divide $\beta_{t,i}^{\text{HMM}}$ by the same factor so that they are cancelled in the forward-backward computation.

The computation can be kept within reasonable bounds by setting the scaling factor

$$c_t = \frac{1}{\sum_{i=1}^{K} \alpha_{t,i}^{\text{HMM}}}$$

Numerical underflow issue in HMM

$$c_t = \frac{1}{\sum_{i=1}^{K} \alpha_{t,i}^{\text{HMM}}}$$

This issue is sometimes not covered in textbooks, although it remains very important for practical implementation of HMM!

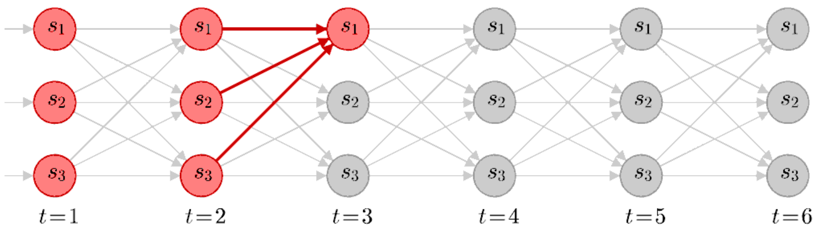
We have by induction

$$\hat{\alpha}_{t,i}^{\text{HMM}} = \left(\prod_{s=1}^{t} c_s\right) \alpha_{t,i}^{\text{HMM}}, \quad \hat{\beta}_{t,i}^{\text{HMM}} = \left(\prod_{s=t}^{T} c_s\right) \beta_{t,i}^{\text{HMM}}$$

With this, the numerator and denominator will cancel out when used in the re-estimation formulas. For example

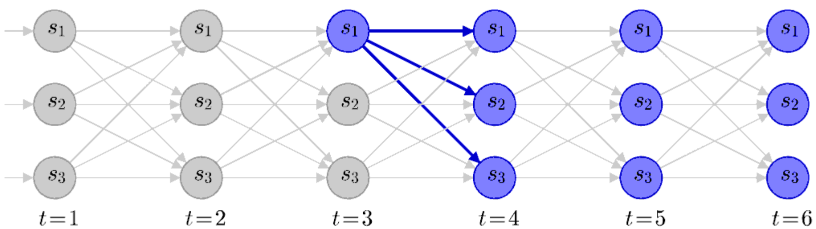
$$\gamma_{t,i}^{\text{HMM}} = \frac{\hat{\alpha}_{t,i}^{\text{HMM}} \hat{\beta}_{t,i}^{\text{HMM}}}{\sum_{k=1}^{K} \hat{\alpha}_{t,k}^{\text{HMM}} \hat{\beta}_{t,k}^{\text{HMM}}} = \frac{(\prod_{s=1}^{t} c_s)(\prod_{s=t}^{T} c_s)\ \alpha_{t,i}^{\text{HMM}} \beta_{t,i}^{\text{HMM}}}{(\prod_{s=1}^{t} c_s)(\prod_{s=t}^{T} c_s)\ \sum_{k=1}^{K} \alpha_{t,k}^{\text{HMM}} \beta_{t,k}^{\text{HMM}}} = \frac{\alpha_{t,i}^{\text{HMM}} \beta_{t,i}^{\text{HMM}}}{\sum_{k=1}^{K} \alpha_{t,k}^{\text{HMM}} \beta_{t,k}^{\text{HMM}}}$$

Summary - Why did we introduce these four intermediary variables in HMM?



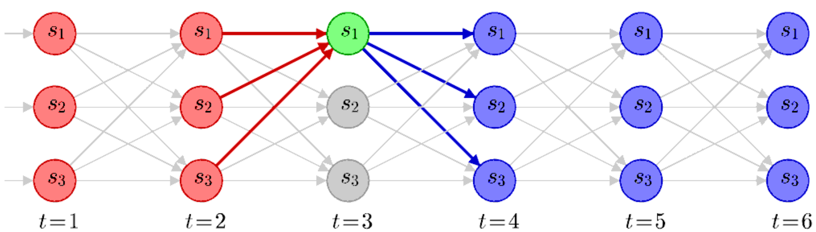
Forward variable

$$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$$



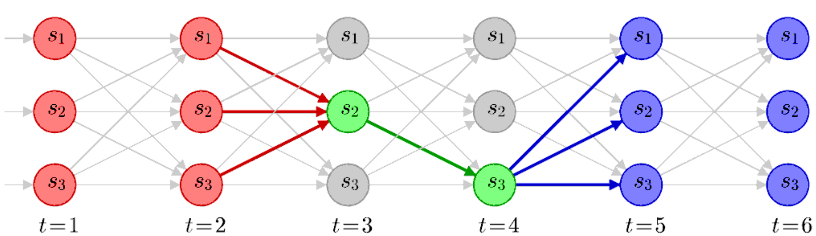
Backward variable

$$\beta_{t,i}^{\text{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$$



Smoothed node marginals

$$\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$$



Smoothed edge marginals

$$\zeta_{t,i,j}^{\text{HMM}} = \mathcal{P}(s_t = i, s_{t+1} = j \mid \boldsymbol{\xi}_{1:T})$$

Summary - Why did we introduce these four intermediary variables in HMM?

How to estimate the parameters of an HMM?

$\rightarrow$ Maximum of expected complete data log-likelihood $\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}})$

How to compute $\frac{\partial \mathcal{Q}}{\partial \Pi_i} = 0$, $\frac{\partial \mathcal{Q}}{\partial a_{i,j}} = 0$, $\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\mu}_i} = 0$ **and** $\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\Sigma}_i} = 0$ **?**

$\rightarrow$ Requires to compute $\zeta_{t,i,j}^{\text{HMM}} = \mathcal{P}(s_t = i, s_{t+1} = j \mid \boldsymbol{\xi}_{1:T})$

$\rightarrow$ Requires to compute $\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i \mid \boldsymbol{\xi}_{1:T})$

How to compute $\zeta_{t,i,j}^{\text{HMM}}$ **and** $\gamma_{t,i}^{\text{HMM}}$ **?**

$\rightarrow$ Requires to compute $\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t = i, \boldsymbol{\xi}_{1:t})$

$\rightarrow$ Requires to compute $\beta_{t,i}^{\text{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+1:T} \mid s_t = i)$

$\max_{\boldsymbol{\Theta}} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\text{old}})$

$\zeta_{t,i,j}^{\text{HMM}}$ $\gamma_{t,i}^{\text{HMM}}$

$\alpha_{t,i}^{\text{HMM}}$ $\beta_{t,i}^{\text{HMM}}$

Viterbi decoding
(MAP vs MPE estimates)

Maximum a posteriori

Most probable explanation

Python notebook: demo_HMM.ipynb

Matlab code: demo_HMM_Viterbi01.m

Viterbi decoding (MAP vs MPE estimates)

Maximum a posteriori — Most probable explanation

The (jointly) most probable sequence of states $\hat{\boldsymbol{s}}^{\text{MAP}}$ is not necessarily the same as the sequence of (marginally) most probable states $\hat{\boldsymbol{s}}^{\text{MPE}}$

$$\hat{\boldsymbol{s}}^{\text{MAP}} = \arg\max_{\{s_1,s_2,\ldots,s_T\}} \mathcal{P}(\boldsymbol{s}|\boldsymbol{\xi})$$

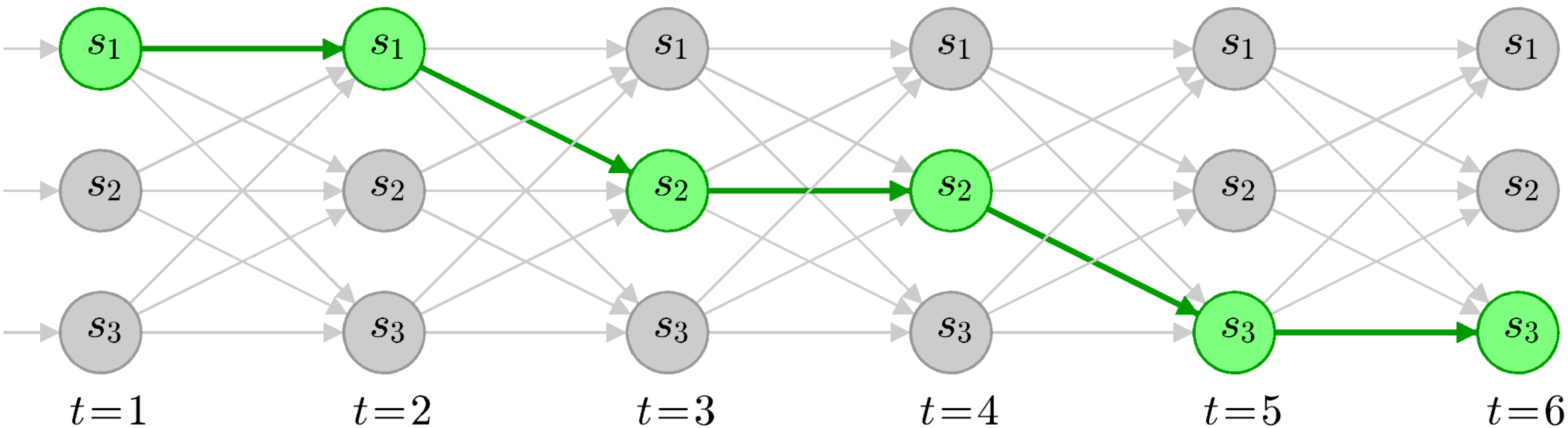
$$\hat{\boldsymbol{s}}^{\text{MPE}} = \left\{ \arg\max_{s_1} \overbrace{\mathcal{P}(s_1|\boldsymbol{\xi})}^{\gamma_1^{\text{HMM}}}, \arg\max_{s_2} \overbrace{\mathcal{P}(s_2|\boldsymbol{\xi})}^{\gamma_2^{\text{HMM}}}, \ldots, \arg\max_{s_T} \overbrace{\mathcal{P}(s_T|\boldsymbol{\xi})}^{\gamma_T^{\text{HMM}}} \right\}$$

$\hat{\boldsymbol{s}}^{\text{MAP}}$ can be computed with the **Viterbi algorithm**, employing the max operator in a forward pass, followed by a backward pass using a **fast traceback procedure** to recover the most probable path.

$\hat{\boldsymbol{s}}^{\text{MPE}}$ can be computed by replacing the sum operator with a max operator in $\boldsymbol{\gamma}^{\text{HMM}}$

$$\gamma_{t,i}^{\text{HMM}} = \mathcal{P}(s_t\!=\!i \,|\, \boldsymbol{\xi}_{1:T})$$

Viterbi decoding - Trellis representation



$$\delta_{t,i} = \max_j \big(\delta_{t-1,j}\ a_{j,i}\big)\,\mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

$$\Psi_{t,i} = \arg\max_j \big(\delta_{t-1,j}\ a_{j,i}\big)$$

$$\hat{s}_t^{\mathrm{MAP}} = \Psi_{t+1,\,\hat{s}_{t+1}^{\mathrm{MAP}}}$$

Viterbi decoding - Algorithm

$$\alpha^{\text{HMM}}_{t,i} = \Big(\sum_{j=1}^{K} \alpha^{\text{HMM}}_{t-1,j}\, a_{j,i}\Big)\, \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

$$\text{with } \alpha^{\text{HMM}}_{1,i} = \Pi_i\, \mathcal{N}\big(\boldsymbol{\xi}_1 \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

Initialization:

$$\delta_{1,i} = \Pi_i\, \mathcal{N}\big(\boldsymbol{\xi}_1 |\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

$$\Psi_{1,i} = 0$$

This is the probability of ending up in state i at time step t by taking the most probable path

Recursion:

$$\delta_{t,i} = \max_j \big(\delta_{t-1,j}\; a_{j,i}\big)\, \mathcal{N}\big(\boldsymbol{\xi}_t |\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

$$\Psi_{t,i} = \arg\max_j \big(\delta_{t-1,j}\; a_{j,i}\big) \quad \forall t \in \{2, 3, \ldots, T\}$$

It tells us the most likely previous state on the most probable path to $s_t = i$

Termination:

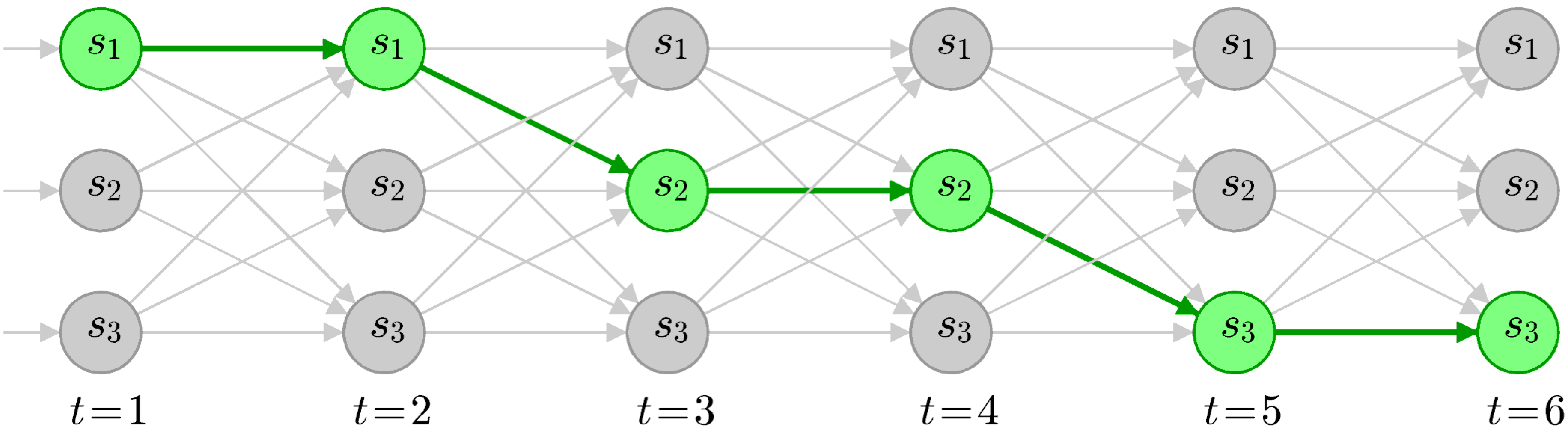
$$\hat{s}^{\text{MAP}}_T = \arg\max_j \delta_{T,j}$$

Backtracking:

$$\hat{s}^{\text{MAP}}_t = \Psi_{t+1,\, \hat{s}^{\text{MAP}}_{t+1}} \qquad \forall t \in \{T-1, T-2, \ldots, 1\}$$

Here, $\delta_{t,i} = \max_{\boldsymbol{s}_{1:t-1}} \mathcal{P}(\boldsymbol{s}_{1:t-1}, s_t = i \,|\, \boldsymbol{\xi}_{1:t})$, and $\Psi_{t,i}$ are state indices that keep track of the states j that maximized $\delta_{t,i}$.

Viterbi decoding - Trellis representation



$$\delta_{t,i} = \max_j \big(\delta_{t-1,j}\ a_{j,i}\big)\, \mathcal{N}\big(\boldsymbol{\xi}_t |\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

$$\Psi_{t,i} = \arg\max_j \big(\delta_{t-1,j}\ a_{j,i}\big)$$

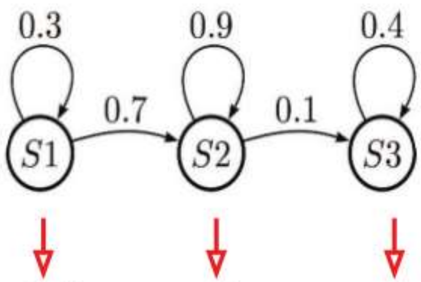
$$\hat{s}_t^{\mathrm{MAP}} = \Psi_{t+1,\, \hat{s}_{t+1}^{\mathrm{MAP}}}$$

Viterbi decoding - Example

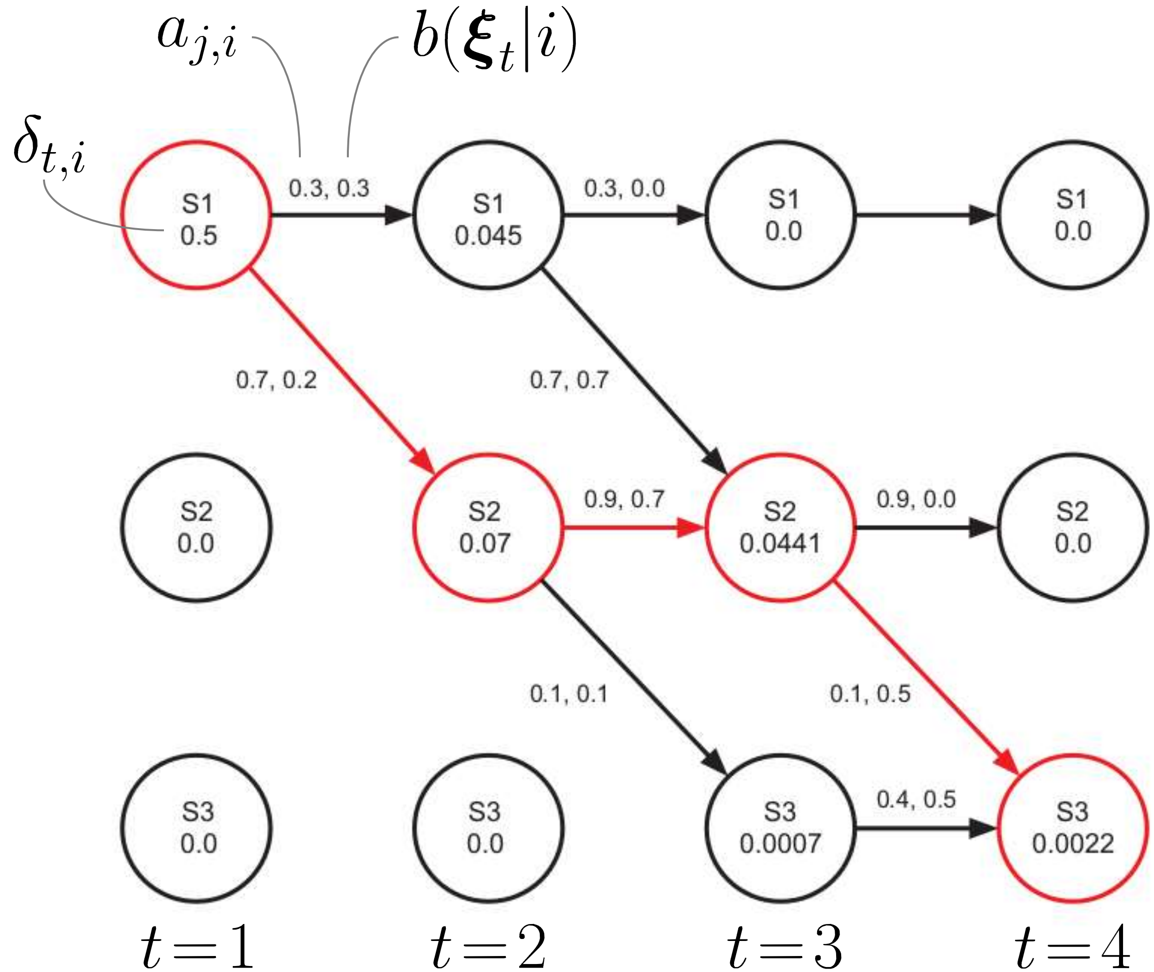
$$\delta_{t,i} = \max_j\left(\delta_{t-1,j}\ a_{j,i}\right)\ b(\boldsymbol{\xi}_t|i)$$

$$\text{with } \delta_{1,i} = \Pi_i\ \mathcal{N}\left(\boldsymbol{\xi}_1|\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)$$

$$\boldsymbol{\Pi} = [1, 0, 0]$$



C1	0.5	0	0
C2	0.2	0	0
C3	0.3	0.2	0
C4	0	0.7	0.1
C5	0	0.1	0
C6	0	0	0.5
C7	0	0	0.4



$$\boldsymbol{\xi} = \{C1, C3, C4, C6\}$$

Image adapted from Kevin P. Murphy (2012), Machine Learning: A Probabilistic Perspective

Numerical underflow issue in Viterbi

Similarly to the forward-backward variables in HMM, we have to take care about potential numerical underflow when implementing Viterbi decoding.

A simple way is to normalize $\delta_{t,i}$ at each time step t with

$$c_t = \frac{1}{\sum_{i=1}^{K} \delta_{t,i}}$$

similarly as in the computation of the forward-backward variables.

Such scaling will not affect the maximum.

Numerical underflow issue in Viterbi

Alternatively, we can work in the log domain. We then have

$$\begin{aligned}\log \delta_{t,i} &= \max_{\boldsymbol{s}_{1:t-1}} \log \mathcal{P}(\boldsymbol{s}_{1:t-1}, s_t\!=\!i \mid \boldsymbol{\xi}_{1:t}) \\ &= \max_j \Big(\log \delta_{t-1,j} + \log a_{i,j} \Big) + \log \mathcal{N}\Big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\Big)\end{aligned}$$

With high dimensional Gaussians as emission distributions, the Viterbi computation with log can result in a **significant speedup**, since computing $\log \mathcal{P}(\boldsymbol{\xi}_t|s_t)$ can be much faster than computing $\mathcal{P}(\boldsymbol{\xi}_t|s_t)$.

It is for this reason that it is also common to use the Viterbi algorithm in the E step of the EM procedure instead of the forward-backward variables when training HMMs with data of high dimension.

Hidden semi-Markov model (HSMM)

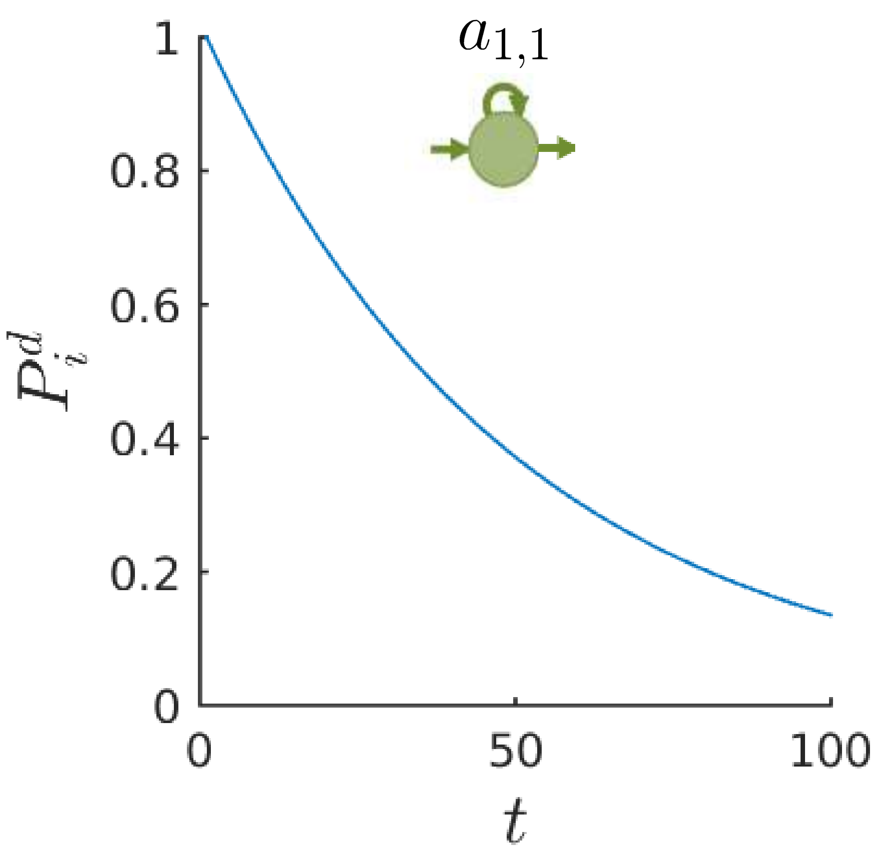
Python notebook: demo_HSMM.ipynb

Matlab code: demo_HSMM01.m

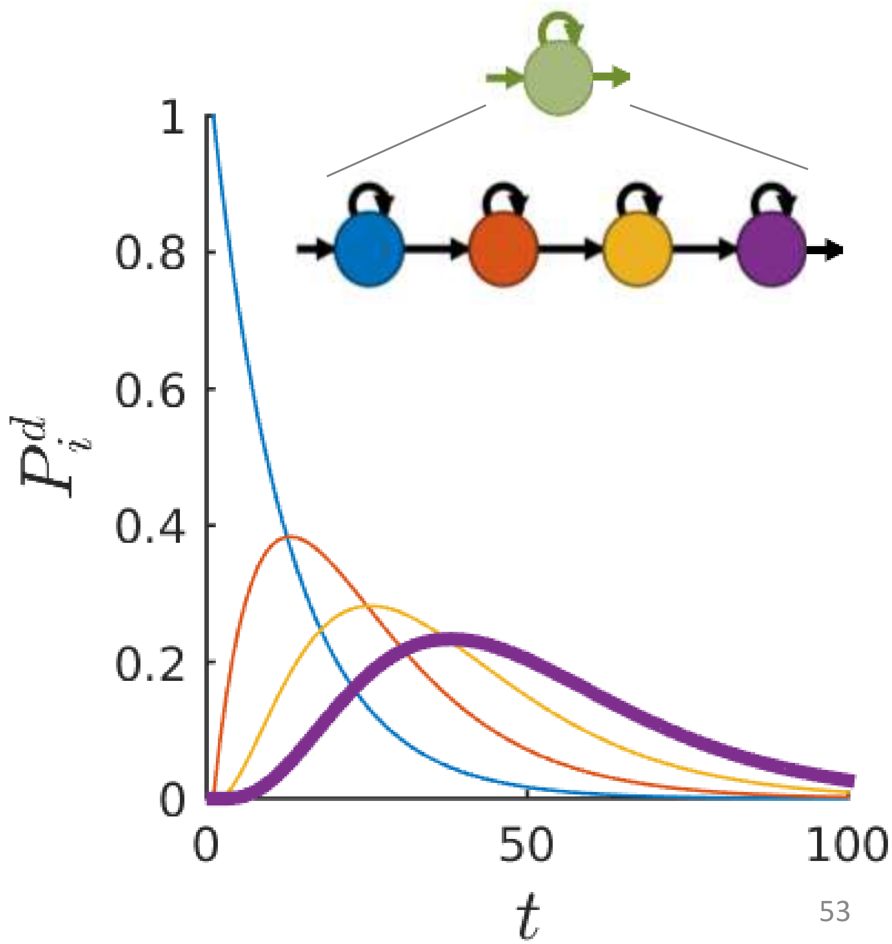
State duration probability in standard HMM

The state duration follows a geometric distribution

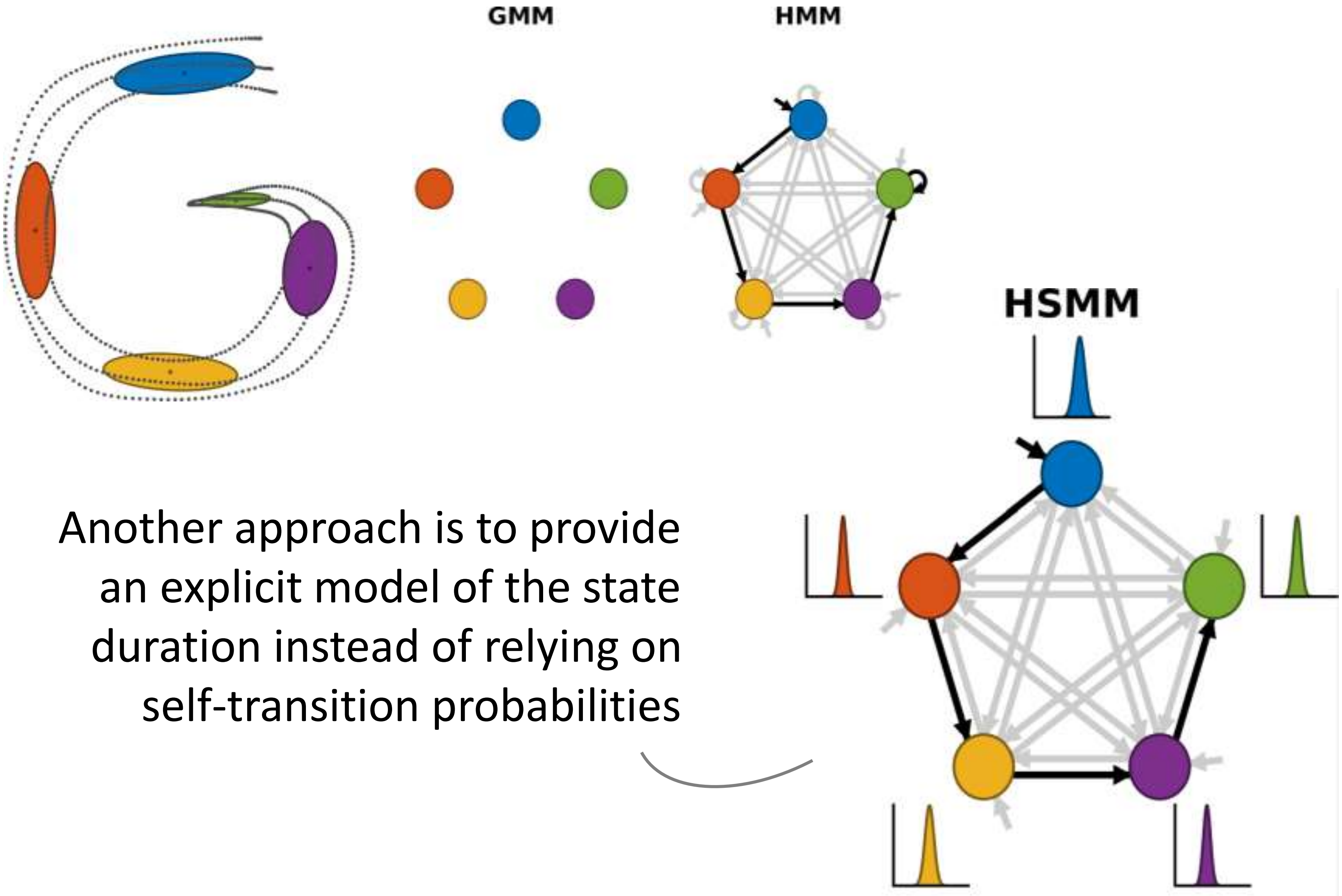
$$\mathcal{P}(d) = a_{i,i}^{d-1}(1 - a_{i,i})$$



By artificially duplicating the number of states while keeping the same emission distribution, other state duration distributions can be modeled



Hidden semi-Markov model (HSMM)



Another approach is to provide an explicit model of the state duration instead of relying on self-transition probabilities

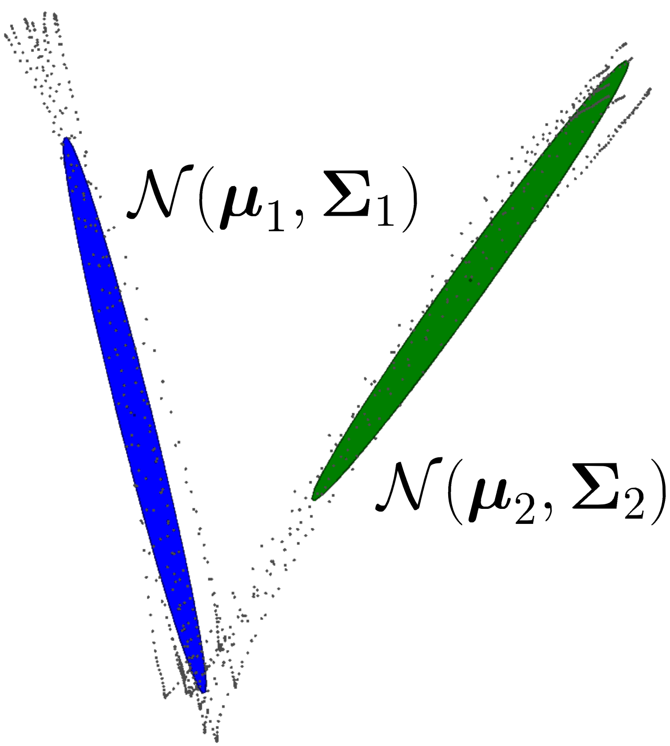
Hidden semi-Markov model (HSMM)

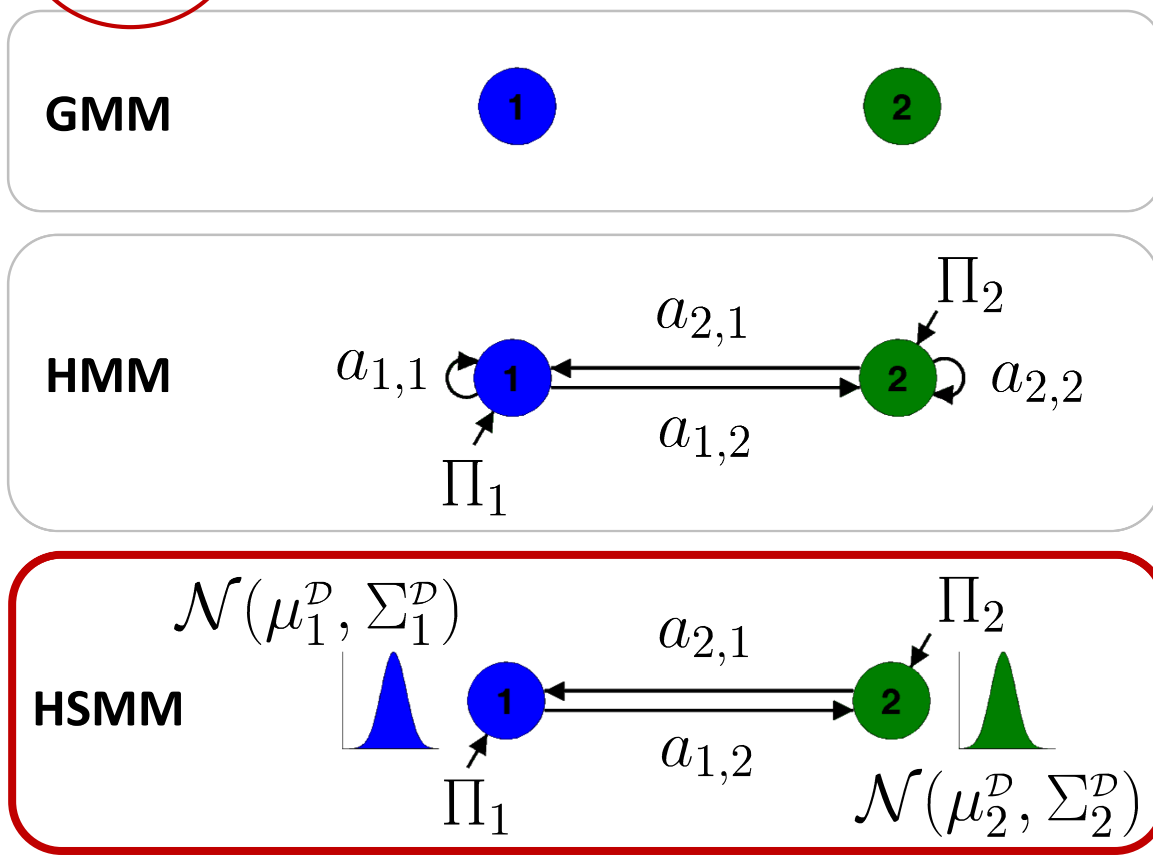
$$\boldsymbol{\Theta}^{\text{GMM}} = \{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$$

Parametric duration distribution

$$\boldsymbol{\Theta}^{\text{HMM}} = \{\{a_{i,j}\}_{j=1}^{K}, \Pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$$

$$\boldsymbol{\Theta}^{\text{HSMM}} = \{\{a_{i,j}\}_{j=1, j\neq i}^{K}, \Pi_i, \mu_i^{\mathcal{D}}, \Sigma_i^{\mathcal{D}}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$$





Hidden semi-Markov model (HSMM)

While the HMM computes the *forward* variable as

$$\alpha_{t,i}^{\text{HMM}} = \sum_{j=1}^{K} \alpha_{t-1,j}^{\text{HMM}} \, a_{j,i} \, \mathcal{N}_{t,i}, \quad \text{with} \quad \mathcal{N}_{t,i} = \mathcal{N}\big(\boldsymbol{\xi}_t \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

the HSMM requires the evaluation of

$$\alpha_{t,i}^{\text{HSMM}} = \sum_{d=1}^{d^{\max}} \sum_{j=1}^{K} \alpha_{t-1,j}^{\text{HSMM}} \, a_{(j,d),(i,d)} \, \mathcal{N}_{(t,d),i}$$

where the system has to keep an history of length $d^{\max}$.

$a_{(j,d),(i,d)}$ is the probability to be in state i at iterations $[t{+}1,\ t{+}d]$ knowing that we were in state j at iterations $[t{-}d{+}1,\ t]$.

$\mathcal{N}_{(t,d),i}$ is the probability to observe $\{\boldsymbol{\xi}_{t-d+1}, \boldsymbol{\xi}_{t-d+2}, \ldots, \boldsymbol{\xi}_t\}$ knowing that we were in state i at iterations $[t{-}d{+}1,\ t]$.

Hidden semi-Markov model (HSMM)

$$\alpha_{t,i}^{\text{HSMM}} = \sum_{d=1}^{d^{\max}} \sum_{j=1}^{K} \alpha_{t-1,j}^{\text{HSMM}} \, a_{(j,d),(i,d)} \, \mathcal{N}_{(t,d),i}$$

An **explicit-duration HSMM** with, for example*, a Gaussian parametrization of the duration $\mathcal{N}_{d,i}^{\mathcal{D}} = \mathcal{N}(d \,|\, \mu_i^{\mathcal{D}}, \Sigma_i^{\mathcal{D}})$ assumes that

$$a_{(j,d),(i,d)} = a_{j,i} \, \mathcal{N}_{d,i}^{\mathcal{D}} \quad \text{and} \quad \mathcal{N}_{(t,d),i} = \prod_{s=t-d+1}^{t} \mathcal{N}_{s,i}$$

which corresponds to the assumption that the state duration is dependent on the current state and independent on the previous state, and that the outputs are conditionally independent.

* *used here only for simplification: other distributions from the exponential family are better suited to model positive counts (e.g.,* ***gamma*** *or* ***log-normal*** *distributions).*

Hidden semi-Markov model (HSMM)

The probability to be in state i at time step t given the partial observation $\boldsymbol{\xi}_{1:t}=\{\boldsymbol{\xi}_1,\boldsymbol{\xi}_2,\ldots,\boldsymbol{\xi}_t\}$ can then be recursively computed

$$\mathcal{P}(s_t=i\,|\,\boldsymbol{\xi}_{1:t}) = \frac{\alpha^{\text{HSMM}}_{t,i}}{\sum_{k=1}^{K}\alpha^{\text{HSMM}}_{t,k}}, \quad \text{with}$$

$$\alpha^{\text{HSMM}}_{t,i} = \sum_{d=1}^{d^{\max}}\sum_{j=1}^{K}\alpha^{\text{HSMM}}_{t-d,j}\; a_{j,i}\, \mathcal{N}^{\mathcal{D}}_{d,i} \prod_{s=t-d+1}^{t} \mathcal{N}_{s,i}, \quad \text{where}$$

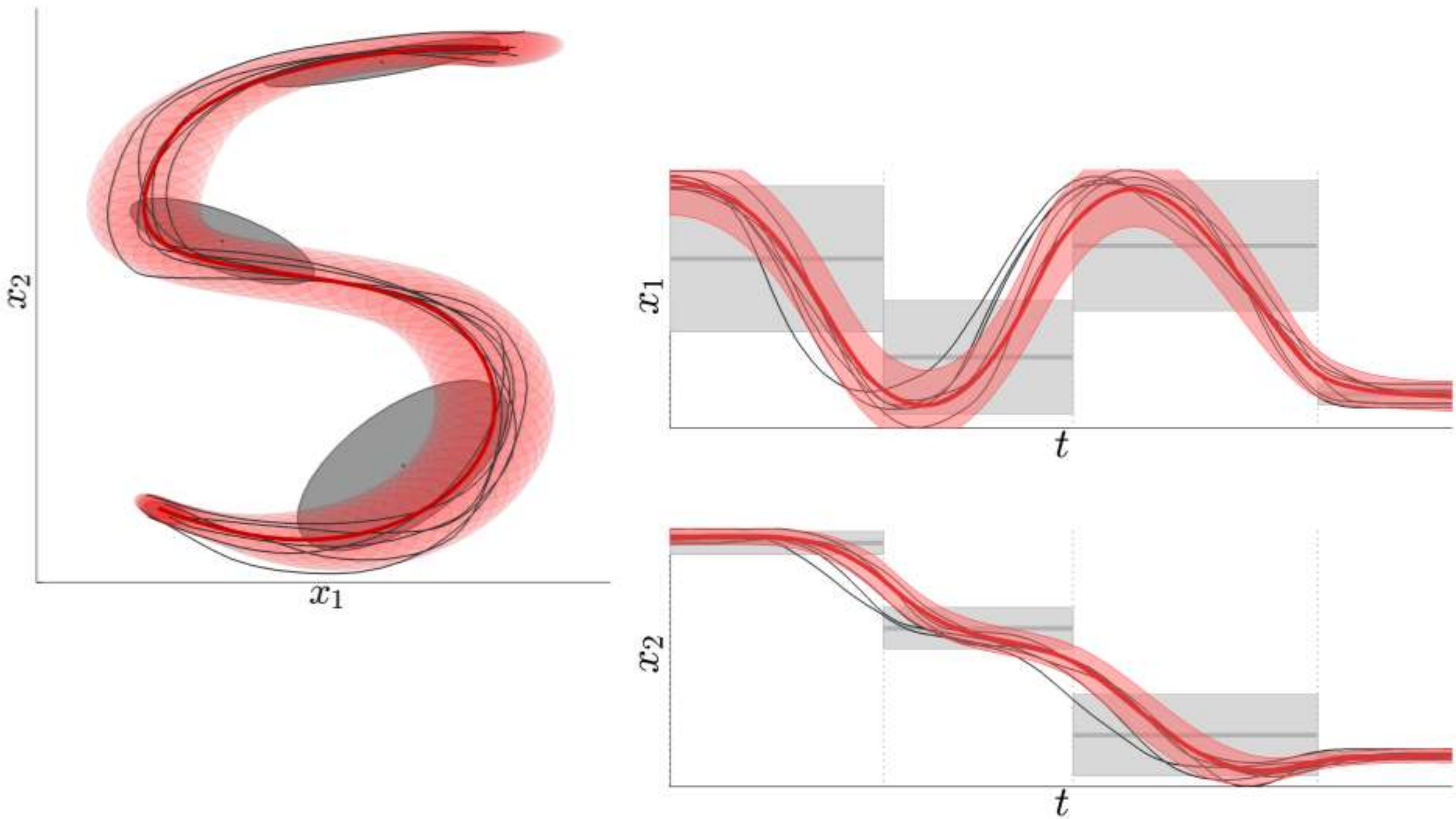
$$\mathcal{N}^{\mathcal{D}}_{d,i} = \mathcal{N}(d\,|\,\mu^{\mathcal{D}}_i, \Sigma^{\mathcal{D}}_i) \quad \text{and} \quad \mathcal{N}_{s,i} = \mathcal{N}\big(\boldsymbol{\xi}_s\,|\,\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\big)$$

HMM with dynamic features

(Trajectory-HMM)

Matlab code: demo_trajHSMM01.m

HMM with dynamic features



HMM with dynamic features

For the encoding of movements, velocity and acceleration can be used as dynamic features. By considering an Euler approximation, the velocity is computed as

$$\dot{\boldsymbol{x}}_t = \frac{\boldsymbol{x}_{t+1} - \boldsymbol{x}_t}{\Delta t}$$

where $\boldsymbol{x}_t$ is a multivariate position vector.

The acceleration is similarly computed as

$$\ddot{\boldsymbol{x}}_t = \frac{\dot{\boldsymbol{x}}_{t+1} - \dot{\boldsymbol{x}}_t}{\Delta t} = \frac{\boldsymbol{x}_{t+2} - 2\boldsymbol{x}_{t+1} + \boldsymbol{x}_t}{\Delta t^2}$$

HMM with dynamic features

$$\dot{\boldsymbol{x}}_t = \frac{\boldsymbol{x}_{t+1} - \boldsymbol{x}_t}{\Delta t}, \quad \ddot{\boldsymbol{x}}_t = \frac{\boldsymbol{x}_{t+2} - 2\boldsymbol{x}_{t+1} + \boldsymbol{x}_t}{\Delta t^2}$$

A vector $\boldsymbol{\zeta}_t$ will be used to represent the concatenated position, velocity and acceleration vectors at time step t

$$\boldsymbol{\zeta}_t = \begin{bmatrix} \boldsymbol{x}_t \\ \dot{\boldsymbol{x}}_t \\ \ddot{\boldsymbol{x}}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ -\frac{1}{\Delta t}\boldsymbol{I} & \frac{1}{\Delta t}\boldsymbol{I} & \boldsymbol{0} \\ \frac{1}{\Delta t^2}\boldsymbol{I} & -\frac{2}{\Delta t^2}\boldsymbol{I} & \frac{1}{\Delta t^2}\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_t \\ \boldsymbol{x}_{t+1} \\ \boldsymbol{x}_{t+2} \end{bmatrix}$$

Here, the number of derivatives will be set up to acceleration ($C = 3$), but the same approach can be applied to a different number of derivatives.

A GMM/HMM/HSMM with centers $\{\boldsymbol{\mu}_i\}_{i=1}^K$ and covariance matrices $\{\boldsymbol{\Sigma}_i\}_{i=1}^K$ is first fit to the dataset $[\boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2, \ldots, \boldsymbol{\zeta}_T]$.

HMM with dynamic features

$$\boldsymbol{\zeta}_t = \begin{bmatrix} \boldsymbol{x}_t \\ \dot{\boldsymbol{x}}_t \\ \ddot{\boldsymbol{x}}_t \end{bmatrix}$$

$\boldsymbol{\zeta}$ and $\boldsymbol{x}$ are defined as large vectors concatenating $\boldsymbol{\zeta}_t$ and $\boldsymbol{x}_t$ for all time steps

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\zeta}_1 \\ \boldsymbol{\zeta}_2 \\ \vdots \\ \boldsymbol{\zeta}_T \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_T \end{bmatrix}$$

Similarly to the matrix operator defined in the previous slide for a single time step, a large sparse matrix $\boldsymbol{\Phi}$ can be defined so that

$$\boldsymbol{\zeta} = \boldsymbol{\Phi}\boldsymbol{x}$$

HMM with dynamic features

D dimensions
C derivatives
T time steps

$$
\begin{bmatrix} \vdots \\ \boldsymbol{x}_t \\ \dot{\boldsymbol{x}}_t \\ \ddot{\boldsymbol{x}}_t \\ \boldsymbol{x}_{t+1} \\ \dot{\boldsymbol{x}}_{t+1} \\ \ddot{\boldsymbol{x}}_{t+1} \\ \vdots \end{bmatrix}
=
\begin{bmatrix}
\ddots & \vdots & \vdots & \vdots & \iddots \\
\cdots & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \cdots \\
\cdots & -\frac{1}{\Delta t}\boldsymbol{I} & \frac{1}{\Delta t}\boldsymbol{I} & \boldsymbol{0} & \cdots \\
\cdots & \frac{1}{\Delta t^2}\boldsymbol{I} & -\frac{2}{\Delta t^2}\boldsymbol{I} & \frac{1}{\Delta t^2}\boldsymbol{I} & \cdots \\
 & \cdots & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\
 & \cdots & -\frac{1}{\Delta t}\boldsymbol{I} & \frac{1}{\Delta t}\boldsymbol{I} & \boldsymbol{0} \\
 & \cdots & \frac{1}{\Delta t^2}\boldsymbol{I} & -\frac{2}{\Delta t^2}\boldsymbol{I} & \frac{1}{\Delta t^2}\boldsymbol{I}
\end{bmatrix}
\begin{bmatrix} \vdots \\ \boldsymbol{x}_t \\ \boldsymbol{x}_{t+1} \\ \boldsymbol{x}_{t+2} \\ \boldsymbol{x}_{t+3} \\ \vdots \end{bmatrix}
$$

$\boldsymbol{\zeta} \in \mathbb{R}^{DCT}$ $\quad$ $\boldsymbol{\Phi} \in \mathbb{R}^{DCT \times DT}$ $\quad$ $\boldsymbol{x} \in \mathbb{R}^{DT}$

(C=3 here)

Large sparse matrix

HMM with dynamic features

For a sequence of states $\boldsymbol{s} = \{s_1, s_2, \ldots, s_T\}$ of T time steps, with discrete states $s_t \in \{1, \ldots, K\}$, the likelihood of a movement $\boldsymbol{\zeta} = \boldsymbol{\Phi x}$ is given by

$$\mathcal{P}(\boldsymbol{\zeta}|\boldsymbol{s}) = \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{\zeta}_t \,|\, \boldsymbol{\mu}_{s_t}, \boldsymbol{\Sigma}_{s_t})$$

where $\boldsymbol{\mu}_{s_t}$ and $\boldsymbol{\Sigma}_{s_t}$ are the center and covariance of state s_t at time step t.

This product can be rewritten as

$$\mathcal{P}(\boldsymbol{\Phi x}|\boldsymbol{s}) = \mathcal{N}(\boldsymbol{\Phi x} \,|\, \boldsymbol{\mu}_{\boldsymbol{s}}, \boldsymbol{\Sigma}_{\boldsymbol{s}})$$

$$\text{with} \quad \boldsymbol{\mu}_{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{\mu}_{s_1} \\ \boldsymbol{\mu}_{s_2} \\ \vdots \\ \boldsymbol{\mu}_{s_T} \end{bmatrix} \quad \text{and} \quad \boldsymbol{\Sigma}_{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{\Sigma}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{s_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Sigma}_{s_T} \end{bmatrix}$$

HMM with dynamic features

For example, for a sequence of states $\boldsymbol{s} = \{1, 1, 2, 2, 3, 3, 3, 4\}$ with $K = 4$ and $T = 8$, we have

$$\boldsymbol{\mu}_{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \\ \boldsymbol{\mu}_2 \\ \boldsymbol{\mu}_3 \\ \boldsymbol{\mu}_3 \\ \boldsymbol{\mu}_3 \\ \boldsymbol{\mu}_4 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\Sigma}_{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_4 \end{bmatrix}$$

$$\boldsymbol{\mu}_{\boldsymbol{s}} \in \mathbb{R}^{DCT} \qquad \boldsymbol{\Sigma}_{\boldsymbol{s}} \in \mathbb{R}^{DCT \times DCT}$$

HMM with dynamic features

By using the relation $\boldsymbol{\zeta} = \boldsymbol{\Phi}\boldsymbol{x}$, we want to retrieve a trajectory

$$\hat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}} \; \log \mathcal{P}(\boldsymbol{\Phi}\boldsymbol{x} \mid \boldsymbol{s})$$

$$\mathcal{N}(\boldsymbol{\Phi}\boldsymbol{x} \mid \boldsymbol{\mu}_{\boldsymbol{s}}, \boldsymbol{\Sigma}_{\boldsymbol{s}}) = (2\pi)^{-\frac{DCT}{2}} |\boldsymbol{\Sigma}_{\boldsymbol{s}}|^{-\frac{1}{2}} \cdot \exp\Big(-\frac{1}{2}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{s}})^{\top} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} (\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{s}})\Big)$$

Equating to zero the derivative of

$$\log \mathcal{P}(\boldsymbol{\Phi}\boldsymbol{x} \mid \boldsymbol{s}) = -\frac{1}{2}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{s}})^{\top} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} (\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{s}}) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{s}}| - \frac{DCT}{2}\log(2\pi)$$

$$\frac{\partial}{\partial \boldsymbol{x}} \boldsymbol{x}^{\top} \boldsymbol{A} \boldsymbol{x} = (\boldsymbol{A} + \boldsymbol{A}^{\top}) \boldsymbol{x}$$

with respect to $\boldsymbol{x}$ yields

$$\boldsymbol{\Phi}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} (\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{\mu}_{\boldsymbol{s}}) = \boldsymbol{0}$$

$$\iff \quad \hat{\boldsymbol{x}} = \left(\boldsymbol{\Phi}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1} \boldsymbol{\mu}_{\boldsymbol{s}}$$

Weighted Least Squares!

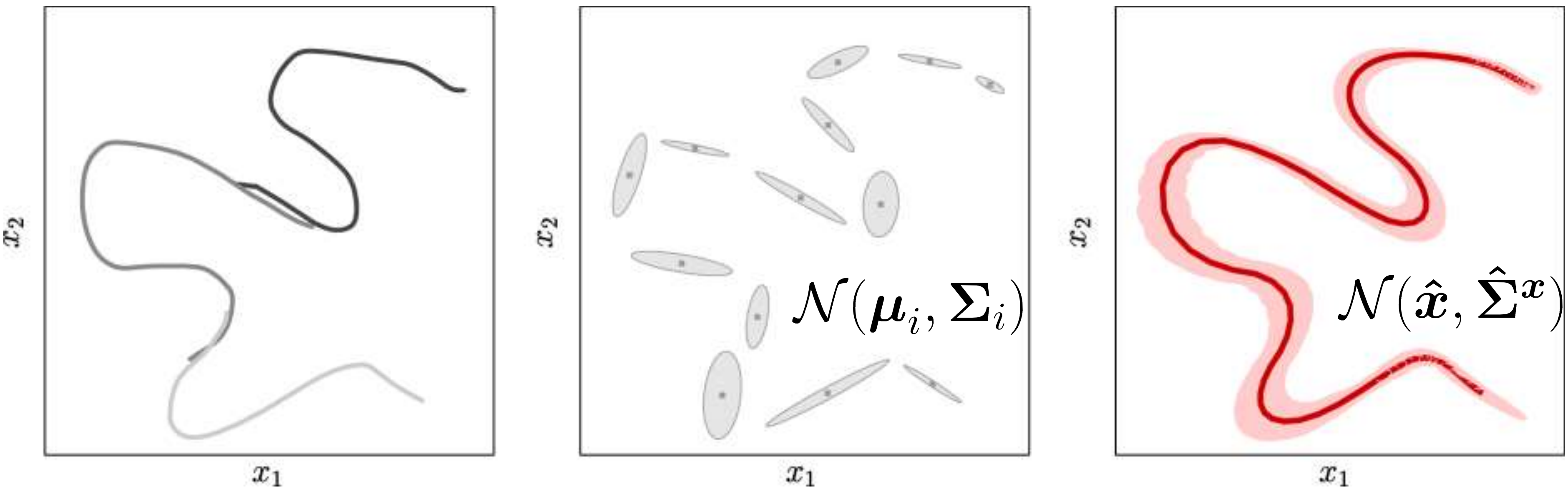
HMM with dynamic features

The covariance error of this estimate is given by

$$\hat{\boldsymbol{\Sigma}}^{\boldsymbol{x}} = \sigma\left(\boldsymbol{\Phi}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1}\boldsymbol{\Phi}\right)^{-1}$$

where σ is a scaling factor.

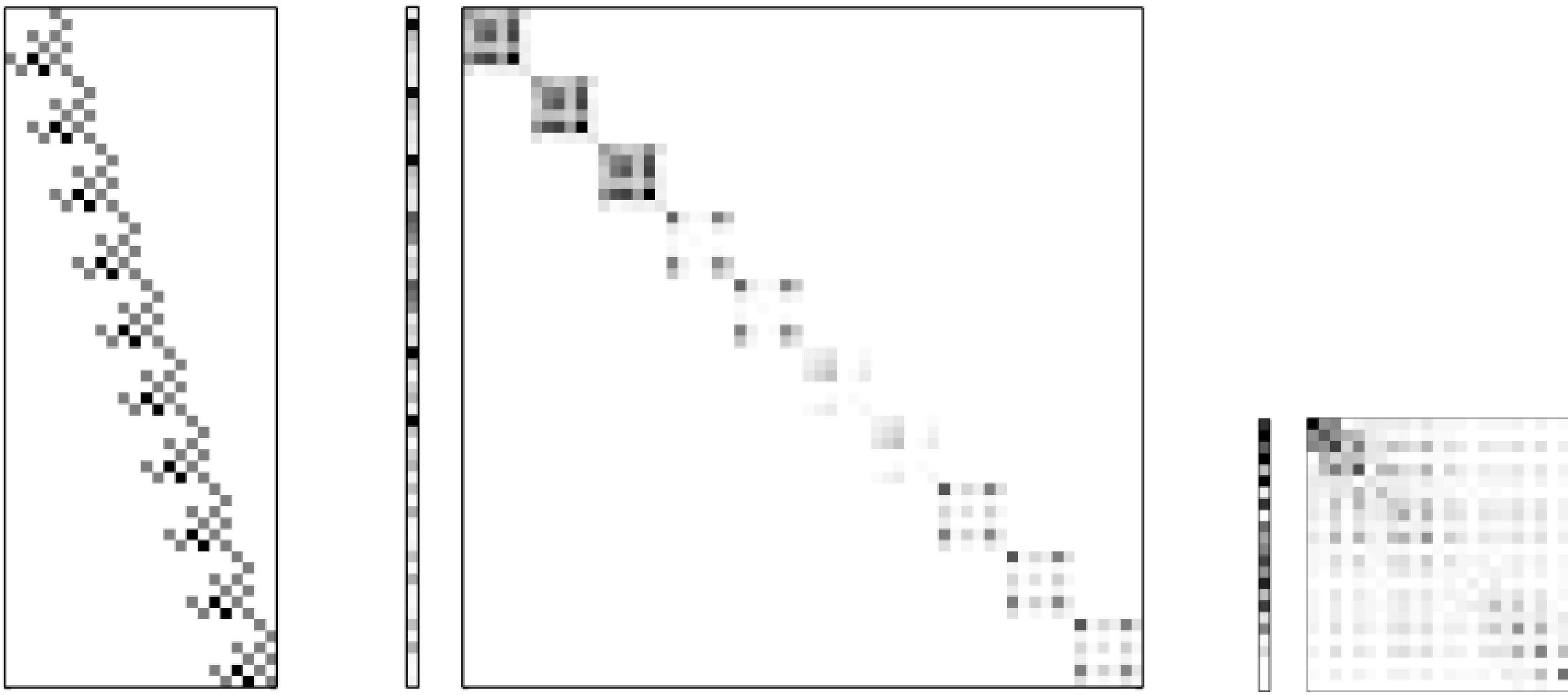
The resulting Gaussian $\mathcal{N}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\Sigma}}^{\boldsymbol{x}})$ forms a trajectory distribution, where $\hat{\boldsymbol{x}} \in \mathbb{R}^{DT}$ is an average trajectory stored in a vector form.



HMM with dynamic features - Summary

$$\hat{\boldsymbol{x}} = \left(\boldsymbol{\Phi}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1}\boldsymbol{\mu}_{\boldsymbol{s}}$$

$$\hat{\boldsymbol{\Sigma}}^{\boldsymbol{x}} = \sigma\left(\boldsymbol{\Phi}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{s}}^{-1}\boldsymbol{\Phi}\right)^{-1}$$



$\boldsymbol{\Phi}$ $\qquad$ $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{s}}, \boldsymbol{\Sigma}_{\boldsymbol{s}})$ $\qquad$ $\mathcal{N}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\Sigma}}^{\boldsymbol{x}})$

References

Hidden Markov model (HMM)

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE, 77:2:257–285, February 1989

Hidden semi-Markov model (HSMM)

S.-Z. Yu. Hidden semi-Markov models. Artificial Intelligence, 174:215–243, 2010

S. E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. Computer Speech & Language, 1(1):29–45, 1986

HMM with dynamic features (Trajectory HMM)

S. Furui. Speaker-independent isolated word recognition using dynamic features of speech spectrum. IEEE Trans. on Acoustics, Speech, and Signal Processing, 34(1):52–59, 1986

H. Zen, K. Tokuda, and T. Kitamura. Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences. Computer Speech and Language, 21(1):153–173, 2007

Appendix

Markov models - Transition matrix

The elements of a n-step transition matrix $\boldsymbol{A}(n)$ are defined as $a_{i,j}(n) = \mathcal{P}(s_{t+n} = j | s_t = i)$, representing the probability to get from state i to j in exactly n steps.

We then have $\boldsymbol{A}(1) = \boldsymbol{A}$ and $a_{i,j}(m+n) = \sum_{k=1}^{K} a_{i,k}(m) a_{k,j}(n)$.

In other words, the probability of getting from i to j in $m+n$ steps is the probability of getting from state i to k in m steps, and then from state k to j in n steps, summed over all k.

We can write this as a matrix multiplication

$$\boldsymbol{A}(m+n) = \boldsymbol{A}(m)\boldsymbol{A}(n)$$

We then have

$$\boldsymbol{A}(n) = \boldsymbol{A}\,\boldsymbol{A}(n-1) = \boldsymbol{A}\,\boldsymbol{A}\,\boldsymbol{A}(n-2) = \ldots = \boldsymbol{A}^n$$

Thus, we can simulate n steps of a Markov chain by raising the transition matrix at the power of n.

MLE of transition matrix in Markov models

A Markov model is described by $\boldsymbol{\Theta}^{\text{MM}} = \{\{a_{i,j}\}_{j=1}^K, \Pi_i\}_{i=1}^K$, where the transition probabilities $a_{i,j}$ are stored in a matrix $\boldsymbol{A}$.

The probability of a sequence $\boldsymbol{\xi}_{1:T}$ of length T is given by

$$\begin{aligned}\mathcal{P}(\boldsymbol{\xi}_{1:T}|\boldsymbol{\Theta}^{\text{MM}}) &= \boldsymbol{\Pi}(\xi_1)\ \boldsymbol{A}(\xi_1,\xi_2)\ \boldsymbol{A}(\xi_2,\xi_3)\ \ldots\ \boldsymbol{A}(\xi_{T-1},\xi_T)\\ &= \prod_{i=1}^K (\Pi_i)^{\mathbb{I}(\xi_1=i)} \prod_{t=2}^T \prod_{i=1}^K \prod_{j=1}^K (a_{i,j})^{\mathbb{I}(\xi_{t-1}=i,\xi_t=j)}\end{aligned}$$

1 if true, 0 if false (e.g. $\Pi_1^0 \cdot \Pi_2^0 \cdot \Pi_3^1 = 1 \cdot 1 \cdot \Pi_3$)

The log-likelihood of a set of M sequences of length T_m is given by

$\log(a^b) = b\log(a)$

$$\sum_{m=1}^M \log \mathcal{P}(\boldsymbol{\xi}_{m,1:T_m}|\boldsymbol{\Theta}^{\text{MM}}) = \sum_{i=1}^K N_i \log \Pi_i + \sum_{i=1}^K \sum_{j=1}^K N_{i,j} \log a_{i,j}$$

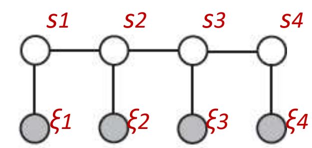
with $N_i = \sum_{m=1}^M \mathbb{I}(\xi_{m,1}=i)$, $\quad N_{i,j} = \sum_{m=1}^M \sum_{t=2}^T \mathbb{I}(\xi_{m,t-1}=i, \xi_{m,t}=j)$

HMM: Smoothed edge marginals

$\zeta_{t,i,j}^{\text{HMM}} = \mathcal{P}(s_t{=}i,\, s_{t+1}{=}j \,|\, \boldsymbol{\xi}_{1:T})$

This result can be retrieved by rewriting the numerator with Bayes rules and the conditional independence properties of the model

$$\mathcal{P}(s_t, s_{t+1}, \boldsymbol{\xi}_{1:T}) = \mathcal{P}(\boldsymbol{\xi}_{1:T} \,|\, s_t, s_{t+1})\; \mathcal{P}(s_t, s_{t+1})$$



$$= \mathcal{P}(\boldsymbol{\xi}_{1:t}|s_t, s_{t+1})\mathcal{P}(\boldsymbol{\xi}_{t+1}|s_t, s_{t+1})\mathcal{P}(\boldsymbol{\xi}_{t+2:T}|s_t, s_{t+1})\mathcal{P}(s_{t+1}|s_t)\mathcal{P}(s_t)$$

Conditional independence property

$$= \mathcal{P}(\boldsymbol{\xi}_{1:t} \,|\, s_t)\; \mathcal{P}(\boldsymbol{\xi}_{t+1} \,|\, s_{t+1})\; \mathcal{P}(\boldsymbol{\xi}_{t+2:T} \,|\, s_{t+1})\; \mathcal{P}(s_{t+1} \,|\, s_t)\; \mathcal{P}(s_t)$$

$$= \underbrace{\mathcal{P}(s_t, \boldsymbol{\xi}_{1:t})}_{\alpha_{t,i}^{\text{HMM}}}\; \underbrace{\mathcal{P}(\boldsymbol{\xi}_{t+1} \,|\, s_{t+1})}_{\mathcal{N}\left(\boldsymbol{\xi}_{t+1}|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}\; \underbrace{\mathcal{P}(\boldsymbol{\xi}_{t+2:T} \,|\, s_{t+1})}_{\beta_{t+1,j}^{\text{HMM}}}\; \underbrace{\mathcal{P}(s_{t+1} \,|\, s_t)}_{a_{i,j}}$$

$$\mathcal{P}(s_{t+1}{=}j|s_t{=}i) = a_{i,j}$$

$$\mathcal{P}(\boldsymbol{\xi}_{t+1}|s_{t+1}{=}j) = \mathcal{N}(\boldsymbol{\xi}_{t+1}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

$$\beta_{t+1,j}^{\text{HMM}} = \mathcal{P}(\boldsymbol{\xi}_{t+2:T} \,|\, s_{t+1}{=}j)$$

$$\alpha_{t,i}^{\text{HMM}} = \mathcal{P}(s_t{=}i, \boldsymbol{\xi}_{1:t}) = \mathcal{P}(\boldsymbol{\xi}_{1:t} \,|\, s_t{=}i)\; \mathcal{P}(s_t{=}i)$$

HSMM: Initialization of forward variable

For $t < d^{\max}$, the initialization is given by

$$\alpha_{1,i}^{\text{HSMM}} = \Pi_i \, \mathcal{N}_{1,i}^{\mathcal{D}} \, \mathcal{N}_{1,i}$$

$$\alpha_{2,i}^{\text{HSMM}} = \Pi_i \, \mathcal{N}_{2,i}^{\mathcal{D}} \prod_{s=1}^{2} \mathcal{N}_{s,i} + \sum_{j=1}^{K} \alpha_{1,j}^{\text{HSMM}} \, a_{j,i} \, \mathcal{N}_{1,i}^{\mathcal{D}} \, \mathcal{N}_{2,i}$$

$$\alpha_{3,i}^{\text{HSMM}} = \Pi_i \, \mathcal{N}_{3,i}^{\mathcal{D}} \prod_{s=1}^{3} \mathcal{N}_{s,i} + \sum_{j=1}^{K} \sum_{d=1}^{2} \alpha_{3-d,j}^{\text{HSMM}} \, a_{j,i} \, \mathcal{N}_{d,i}^{\mathcal{D}} \prod_{s=4-d}^{3} \mathcal{N}_{s,i} \quad \text{etc.}$$

which corresponds to an update rule for $t < d^{\max}$ written as

$$\alpha_{t,i}^{\text{HSMM}} = \Pi_i \, \mathcal{N}_{t,i}^{\mathcal{D}} \prod_{s=1}^{t} \mathcal{N}_{s,i} + \sum_{j=1}^{K} \sum_{d=1}^{t-1} \alpha_{t-d,j}^{\text{HSMM}} \, a_{j,i} \, \mathcal{N}_{d,i}^{\mathcal{D}} \prod_{s=t-d+1}^{t} \mathcal{N}_{s,i}$$