SB1.2/SM2 Computational Statistics
Hidden Markov Models

François Caron

Hilary Term 2019

SB1.2/SM2 Computational Statistics - HMM

- References:
  - D. Barber. Bayesian Reasoning and Machine Learning, Cambridge University Press, 2012.
  - K.P. Murphy. Machine Learning. A probabilistic perspective. The MIT Press, 2012
- More advanced references
  - R. van Handel. Hidden Markov models. Lecture notes, University of Princeton, 2008.
  - O. Cappé, E. Moulines, T. Ryden. Inference in Hidden Markov Models. Springer, 2007.
- The course requires the following notions:
  - Discrete Markov chains [Part A Probability]
  - Bayesian methods: prior, posterior, maximum a posteriori [Part A Statistics]

Outline

Motivating example

- Sequence of observations $y_{1:T} = (y_1, y_2, \ldots, y_T)$, $T \geq 1$
- Some natural order of the data
- Index t in $(y_t)_{t=1,\ldots,T}$ may refer to time, index of a site on a chromosome or a piece of DNA or the position of a word in a sentence
- Objective: For each t, infer some non-observed/hidden quantity of interest $x_t \in \mathcal{X}$ where $\mathcal{X}$ is a finite set

Motivating example

- Part-of-Speech Tagging (POST): task of labelling a word in a text corpus as a particular part of speech, such as noun, verb, adjective or adverb

PRON	VB	ADV	ADJ	PREP	DET	ADJ	NOUN
Nothing	*is*	*so*	*painful*	*to*	*the*	*human*	*mind*

PREP	DET	ADJ	COORD	ADJ	NOUN
as	*a*	*great*	*and*	*sudden*	*change.*

Figure: Example of Part-of-Speech tagging. Observations $(y_1, \ldots, y_T)$ are the T words in a document, where y_t refers to the t's word in the document. One is interested in inferring the tags $(x_1, \ldots, x_T)$ where x_t is the tag associated to the t's word y_t in the sentence.

Motivating example

- POST may be challenging, as some words such as *change* or *mind* may correspond to different parts of speech (noun/verb) depending on the context.
- One possibility is to treat the unknown tags x_t as fixed parameters.
- But often a lot of prior information on the hidden sequence is available
- In the POST example, some POS tags have a higher frequency of appearance than others. We also know that a sentence has some structure: a pronoun is often followed by a verb, an adjective by a noun, etc. and we may want to probabilistically encode this information in order to get better estimates.

Motivating example

- Bayesian framework is attractive here
- Hidden tags of interest $X_1, \ldots, X_T$ are random variables
- Joint probability mass function over the hidden and observed variables:

$$\begin{aligned} p(x_{1:T}, y_{1:T}) &:= \mathbb{P}(X_{1:T} = x_{1:T}, Y_{1:t} = y_{1:t}) \\ &= \underbrace{\mathbb{P}(Y_{1:T} = y_{1:T} | X_{1:T} = x_{1:T})}_{\text{Likelihood}} \underbrace{\mathbb{P}(X_{1:T} = x_{1:T})}_{\text{Prior}}. \end{aligned}$$

- This joint probability mass function defines our statistical model and can capture complex dependencies between the hidden states and the observations.

Motivating example

- Given some observation sequence $(y_1, \ldots, y_T)$ the information about the hidden parameter of interest is encapsulated in the posterior probability mass function

$$\begin{aligned} p(x_{1:T}|y_{1:T}) &:= \mathbb{P}(X_{1:T} = x_{1:T}|Y_{1:T} = y_{1:T}) \\ &= \frac{\mathbb{P}(Y_{1:T} = y_{1:T}|X_{1:T} = x_{1:T})\mathbb{P}(X_{1:T} = x_{1:T})}{\mathbb{P}(Y_{1:T} = y_{1:T})} \end{aligned}$$

Motivating example

- Posterior mode or maximum a posteriori (MAP) estimate
- Solve the combinatorial optimization problem

$$\hat{x}_{1:T} = \underset{x_{1:T} \in \mathcal{X}^T}{\arg\max}\ p(x_{1:T}|y_{1:T}).$$

- Combinatorial search space has $|\mathcal{X}|^T$ elements and grows exponentially fast with T.
- Calculating exactly the MAP estimate becomes impossible even for reasonably small values of T.
- For example, for a document with $T = 100$ words and $|\mathcal{X}| = 20$ tags, exhaustive search requires to evaluate the $20^{100} \simeq 10^{130}$ possible sequences.

Motivating example

- Simpler model $p(x_{1:T}, y_{1:T})$?
- Obvious simplification is to assume independence between the pairs (X_t, Y_t), (X_τ, Y_τ) for any $t \neq \tau$, thus ignoring the sequential structure.
- Posterior factorizes over t
- MAP estimation reduces to solving independently

$$\hat{x}_t = \arg\max_{x_t \in \mathcal{X}} \mathbb{P}(Y_t = y_t | X_t = x_t)\mathbb{P}(X_t = x_t), \quad t = 1, \ldots, T,$$

- Linear complexity $T|\mathcal{X}|$ in both T and $|\mathcal{X}|$
- We can now compute estimates, but does the model capture enough information to perform the task?
- No: By considering each word independently, the estimated tag will be the same for multiple occurences of the same word, which is clearly inappropriate for words like **mind** or **change** which may be assigned different tags.

Motivating example

- Considering a full model $p(x_{1:T}, y_{1:T})$ may give a realistic probabilistic representation of the data and hidden variables, but is practically useless as the estimate cannot be calculated
- Using a much simpler statistical model which assumes independence across time allows to compute the estimate, but is too simplistic to address the task.
- Necessary trade-off between the complexity of the model and the computational cost of obtaining summaries of the parameters of interest for that model
- Hidden Markov Models is a class of models for sequential data that offers a very attractive trade-off between the model's ability to capture dependencies and the tractability of the estimation algorithms.

Motivating example

- Keep in mind that HMMs, like other statistical models, are in general not meant to reproduce the true data generating process.
- They are an interpretation and approximation of the real world, targeted to the problem at hand.
- As George Box famously wrote in his 1987 book

  > *Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful.[...]*
  > *Essentially, all models are wrong, but some are useful.*

- For many problems involving sequential data, Hidden Markov Models are indeed very useful, if not realistic, statistical models!

Outline

Outline

Recap: Discrete Markov chain

- Let $X_{0:T} = (X_0, X_1, \ldots, X_T) \in \mathcal{X}^{T+1}$ be a sequence of random variables
- Finite set $\mathcal{X}$ is called the state-space.
- The process is called a Markov chain if for any $t \geq 0$ and any $x_0, \ldots, x_{t+1} \in \mathcal{X}$,

$$\mathbb{P}(X_{t+1} = x_{t+1} | X_t = x_t, \ldots, X_0 = x_0) = \mathbb{P}(X_{t+1} = x_{t+1} | X_t = x_t)$$

- The Markov chain is said to be homogeneous if $\mathbb{P}(X_{t+1} = j | X_t = i)$ does not depend on t
- In this case we write

$$A_{i,j} := \mathbb{P}(X_{t+1} = j | X_t = i) \quad i, j \in \mathcal{X}$$

- For simplicity of exposure, we will only consider homogeneous Markov chains
- For $x_0 \in \mathcal{X}$, let $\mu_{x_0} = \mathbb{P}(X_0 = x_0)$ be the pmf of the initial state X_0.

Recap: Discrete Markov chain

- The joint pmf of $X_{1:T}$ is parametrized by $(A_{i,j})_{i,j\in\mathcal{X}}$ and $(\mu_i)_{i\in\mathcal{X}}$

$$\begin{aligned} p(x_{0:T}) &:= \mathbb{P}(X_0 = x_0, \dots, X_T = x_t) \\ &= \mathbb{P}(X_0 = x_0) \prod_{t=1}^{T} \mathbb{P}(X_t = x_t | X_{t-1} = x_{t-1}) \\ &= \mu_{x_0} \prod_{t=1}^{T} A_{x_{t-1}, x_t} \end{aligned}$$

Outline

Hidden Markov Models

- Let $X_{0:T} = (X_0, X_1, \dots, X_T) \in \mathcal{X}^{T+1}$ be a homogeneous Markov chain with transition matrix (A_{ij}).
- Consider another sequence of random variables $Y_{1:T} = (Y_1, \dots, Y_T)$ taking values in some set $\mathcal{Y}$ called the observation space.
- The random variables Y_t may be continuous or discrete.
- For simplicity of exposure, we only consider discrete random variables Y_t
- The random variables $(Y_1, \dots, Y_T)$ are **independent** conditional on the sequence $(X_0, \dots, X_T)$.
- For discrete random variables Y_t

$$\begin{aligned}\mathbb{P}(Y_1 = y_1, \dots, Y_T = y_T | X_0 = x_0, \dots, X_T = x_T) \\ = \prod_{t=1}^{T} \mathbb{P}(Y_t = y_t | X_t = x_t)\end{aligned}$$

Hidden Markov Models

- If the conditional probability $\mathbb{P}(Y_t = y_t | X_t = x_t)$ does not depend on t, then the HMM is said to be homogeneous.
- We write, for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$

$$g_x(y) := \mathbb{P}(Y_t = y | X_t = x)$$

where $g_x(y)$ is called the emission probability mass function.

- Using the Markov and conditional independence properties of the HMM, the joint probability of the hidden states and observations is

$$\mathbb{P}(X_{0:T} = x_{0:T}, Y_{0:T} = y_{0:T}) = \mu_{x_0} \prod_{t=1}^{T} g_{x_t}(y_t) A_{x_{t-1}, x_t}$$

Hidden Markov Models

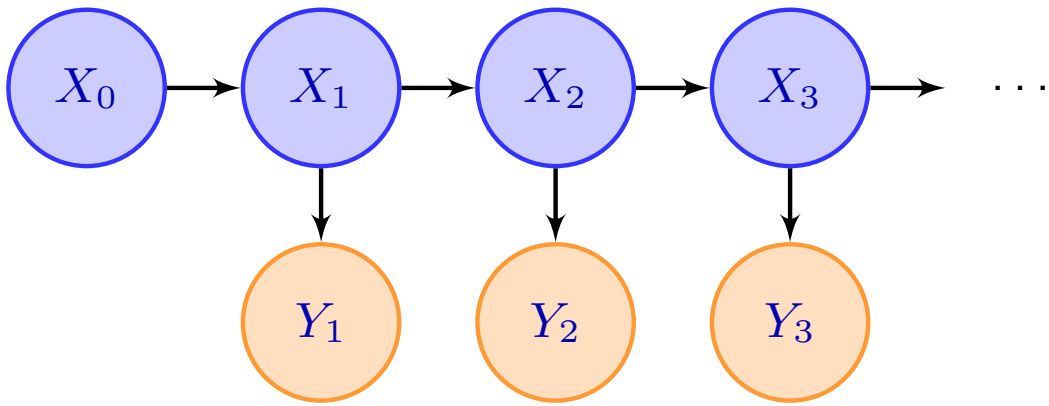


Figure: Graphical representation of a hidden Markov model. Hidden states are represented with blue circles, and observations with orange circles.

Outline

Hidden Markov Models: Examples

Part-of-Speech tagging

PRON	VB	ADV	ADJ	PREP	DET	ADJ	NOUN
Nothing	*is*	*so*	*painful*	*to*	*the*	*human*	*mind*

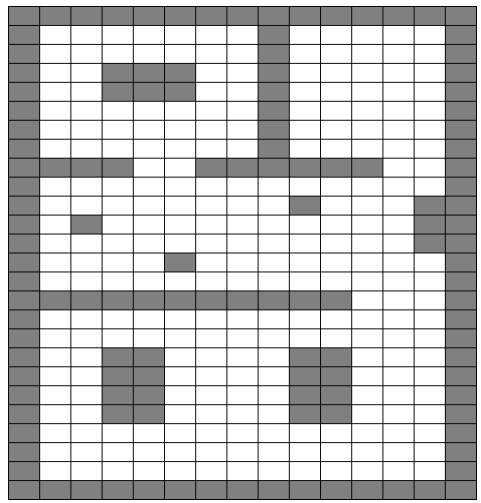
PREP	DET	ADJ	COORD	ADJ	NOUN
as	*a*	*great*	*and*	*sudden*	*change.*

- $\mathcal{X}$ is a set of tags or POS
- $\mathcal{Y}$ is a set of words (vocabulary)
- Y_t is the t's word in the document
- X_t is the unknown POS tag

Hidden Markov Models: Examples

Robot localization

- Robot with an internal grid-based map of his environment and some sensors which enable it to detect obstacles/amers.
- The state space $\mathcal{X}$ is set of possible positions of the robot on a grid
- Observation space is $\mathcal{Y} = \{0, 1\}$ (detection/non-detection)
- X_t is the (unknown) position of the robot at time t on the grid
- Y_t is the observed detection/non-detection of an obstacle
- Objective: calculate over time the probability $\mathbb{P}(X_t = x_t | Y_{1:T} = y_{1:T})$ that the robot is in a given cell $x_t \in \mathcal{X}$ at time t given the measurements up to time t.



Hidden Markov Models: Examples

Gene finding

- Genetic material of an organism is encoded in DNA
- Sequence of base pairs made of four chemical bases: adenine (A), guanine (G), cytosine (C) and thymine (T).
- Genetic sequence is made of coding and non-coding sub-sequences.
- Coding sub-sequences encode proteins and the task of separating coding and non-coding sequences of DNA is known as gene finding and is an important problem in computational biology.

Coding Non-coding

ATTGAC CCATCGTGC CATAGTCGC TGA

Hidden Markov Models: Examples

Gene finding

- State space $\mathcal{X} = \{0, 1\}$ where 0:coding, 1:non-coding
- Observations Y_t are the type of the base pair, encoded with the four-letters observation space $\mathcal{Y} = \{A, C, G, T\}$.
- $X_t \in \{0, 1\}$ in the hidden state (coding/non-coding).
- Based on the observed DNA sequence $y_{1:T}$, we aim at inferring the coding sub-sequences.

Coding Non-coding

ATTGAC CCATCGTGC CATAGTCGC TGA

Outline

Hidden Markov Models

Some notations

We will use the following notations

$$\begin{aligned} p(x_{t+1}|x_t) &= \mathbb{P}(X_{t+1} = x_{t+1}|X_t = x_t) \\ p(y_t|x_t) &= \mathbb{P}(Y_t = y_t|X_t = x_t) \\ p(x_t|y_{1:t}) &= \mathbb{P}(X_t = x_t|Y_1 = y_1, \dots, Y_t = y_t) \\ p(y_{1:t}) &= \mathbb{P}(Y_1 = y_1, \dots, Y_t = y_t) \end{aligned}$$

etc., where the subscripts indicate which random variables we are referring to.

Hidden Markov Models: Inference

- Assume that we have a sequence of observations $(y_1, \dots, y_T)$.
- The classical inference problems are the following:
  - Filtering
    $$p(x_t|y_{1:t})$$
  - Prediction
    $$p(x_t|y_{1:s}), \quad s < t$$
  - Smoothing
    $$p(x_t|y_{1:s}), \quad s > t$$
  - Likelihood
    $$p(y_{1:T})$$
  - Most likely state path
    $$\arg\max_{x_{0:T}} p(x_{0:T}|y_{1:T})$$

Hidden Markov Models: Inference

Filtering and forward recursion

- We are interested in the conditional probability mass function

$$p(x_t|y_{1:t})$$

  of the hidden state X_t given the data observed up to time t.
- Note that, by Bayes rule, $p(x_t|y_{1:t})$ can be obtained by normalizing $p(x_t, y_{1:t})$

$$p(x_t|y_{1:t}) = \frac{p(x_t, y_{1:t})}{\sum_{x'_t \in \mathcal{X}} p(x'_t, y_{1:t})}$$

- Recursion for $p(x_t, y_{1:t})$

Hidden Markov Models: Inference

Filtering and forward recursion

$$
\begin{aligned}
p(x_t, y_{1:t}) &= \sum_{x_{t-1} \in \mathcal{X}} p(x_t, x_{t-1}, y_t, y_{1:t-1}) \\
&= \sum_{x_{t-1} \in \mathcal{X}} p(y_t | x_t, x_{t-1}, y_{1:t-1}) p(x_t | x_{t-1}, y_{1:t-1}) p(x_{t-1}, y_{1:t-1}) \\
&= p(y_t | x_t) \sum_{x_{t-1} \in \mathcal{X}} p(x_t | x_{t-1}) p(x_{t-1}, y_{1:t-1})
\end{aligned}
$$

Define $\alpha_t(x_t) = p(x_t, y_{1:t})$. The equation above defines the α-recursion. For $t = 1, \ldots, T$, $x_t \in \mathcal{X}$

$$
\alpha_t(x_t) = p(y_t | x_t) \sum_{x_{t-1} \in \mathcal{X}} p(x_t | x_{t-1}) \alpha_{t-1}(x_{t-1})
$$

with $\alpha_0(x_0) = p(x_0)$.

Hidden Markov Models: Inference

Filtering and forward recursion

- Consider $\mathcal{X} = \{1, \dots, K\}$
- Forward α-recursion:
- For $i = 1, \dots, K$, set $\alpha_0(i) = \mu_i$
- For $t = 1, \dots, T$
    - For $j = 1, \dots, K$, set

$$\alpha_t(j) = g_j(y_t) \sum_{i=1}^{K} A_{i,j} \alpha_{t-1}(i)$$

Hidden Markov Models: Inference

Filtering and forward recursion

The filtering pmf is obtained by normalizing $\alpha_t(x_t)$ as

$$p(x_t|y_{1:t}) = \frac{p(x_t, y_{1:t})}{p(y_{1:t})} = \frac{\alpha_t(x_t)}{\sum_{x \in \mathcal{X}} \alpha_t(x)}$$

The likelihood term $p(y_{1:T})$ can be computed from the α recursion

$$p(y_{1:T}) = \sum_{x \in \mathcal{X}} \alpha_T(x)$$

Hidden Markov Models: Inference

Filtering and forward recursion

- The computation cost of the whole forward recursion is $O(T|\mathcal{X}|^2)$
- Note that the proposed recursion may suffer from numerical underflow/overflow, as α_t may become very small/large for large t
- A solution is to normalize α_t; or similarly, to propagate the alternative predict-update recursion

$$p(x_t|y_{1:t-1}) = \sum_{x_{t-1}\in\mathcal{X}} p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1}) \qquad \text{Predict}$$

$$p(x_t|y_{1:t}) = \frac{g_{x_t}(y_t)p(x_t|y_{1:t-1})}{\sum_{x'_t\in\mathcal{X}} g_{x'_t}(y_t)p(x'_t|y_{1:t-1})} \qquad \text{Update}$$

Hidden Markov Models: Inference

Forward-backward smoothing

- We are now interested in the conditional probability mass function

$$p(x_t|y_{1:T})$$

of the state X_t given all the data from time 1 to T.

- First note that

$$\begin{aligned} p(x_t|y_{1:T}) &= \frac{p(x_t, y_{1:T})}{p(y_{1:T})} \\ &= \frac{p(x_t, y_{1:t})p(y_{t+1:T}|x_t)}{p(y_{1:T})} \end{aligned}$$

hence $p(x_t|y_{1:T})$ can be obtained by normalizing $p(x_t, y_{1:t})p(y_{t+1:T}|x_t)$.

- The first term is $\alpha_t(x_t)$ which can be obtained by a forward recursion. We now show how to obtain the second term $\beta_t(x_t) = p(y_{t+1:T}|x_t)$ by a backward recursion

Hidden Markov Models: Inference

Forward-backward smoothing

$$
\begin{aligned}
p(y_{t:T}|x_{t-1}) &= \sum_{x_t \in \mathcal{X}} p(y_{t:T}, x_t|x_{t-1}) \\
&= \sum_{x_t \in \mathcal{X}} p(y_t|y_{t+1:T}, x_t, x_{t-1}) p(y_{t+1:T}, x_t|x_{t-1}) \\
&= \sum_{x_t \in \mathcal{X}} p(y_t|x_t) p(y_{t+1:T}|x_t, x_{t-1}) p(x_t|x_{t-1}) \\
&= \sum_{x_t \in \mathcal{X}} p(y_t|x_t) p(y_{t+1:T}|x_t) p(x_t|x_{t-1})
\end{aligned}
$$

Hence β_t follows the following backward recursion for $t = T, \ldots, 2$

$$
\beta_{t-1}(x_{t-1}) = \sum_{x_t \in \mathcal{X}} p(y_t|x_t) p(x_t|x_{t-1}) \beta_t(x_t)
$$

with $\beta_T(x_T) = 1$.

Hidden Markov Models: Inference

Forward-backward smoothing

- The forward and backward recursions can be run independently
- The smoothing posterior is finally obtained by normalization

$$p(x_t|y_{1:T}) = \frac{p(x_t, y_{1:T})}{p(y_{1:T})} = \frac{\alpha_t(x_t)\beta_t(x_t)}{\sum_{x\in\mathcal{X}} \alpha_t(x)\beta_t(x)}$$

- The overall computational cost of the forward-backward algorithm is $O(T|\mathcal{X}|^2)$

Hidden Markov Models: Inference

Illustration

- Consider a frog on a ladder with $K = 6$ levels, and let X_t be the level at which the frog is at time t.
- $(X_t)_{t=0,\dots}$ is a Markov chain with the following transition probabilities

$$A_{i,i+1} = \frac{1-p}{2} \quad \text{for } i = 1, \dots, T-1$$
$$A_{i,i} = p \quad \text{for } i = 1, \dots, T$$
$$A_{i,i-1} = \frac{1-p}{2} \quad \text{for } i = 2, \dots, T$$

  and $A_{1,2} = 1 - p$ and $A_{K,1} = \frac{1-p}{2}$, $p = 0.4$.
- At time $t = 0$, all states are equally likely

Hidden Markov Models: Inference

Illustration

- The frog's position is not observed, but a frog's detector is installed at the lowest level of the ladder, which sends a signal $Y_t \in \{1, 2\}$ at each time t where 2 indicates detection and 1 non-detection.
- Probability of detection

$$B_{k,2} := \mathbb{P}(Y_t = 2 | X_t = k) = \begin{cases} 0.9 & \text{if } k = 1 \\ 0.5 & \text{if } k = 2 \\ 0.1 & \text{if } k = 3 \\ 0 & \text{Otherwise} \end{cases}$$

- We observe the following sequence $y_{1:14} = (1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 2, 2, 1, 2)$ and want to infer the filtering and smoothing pmfs of the frog's position at each time t

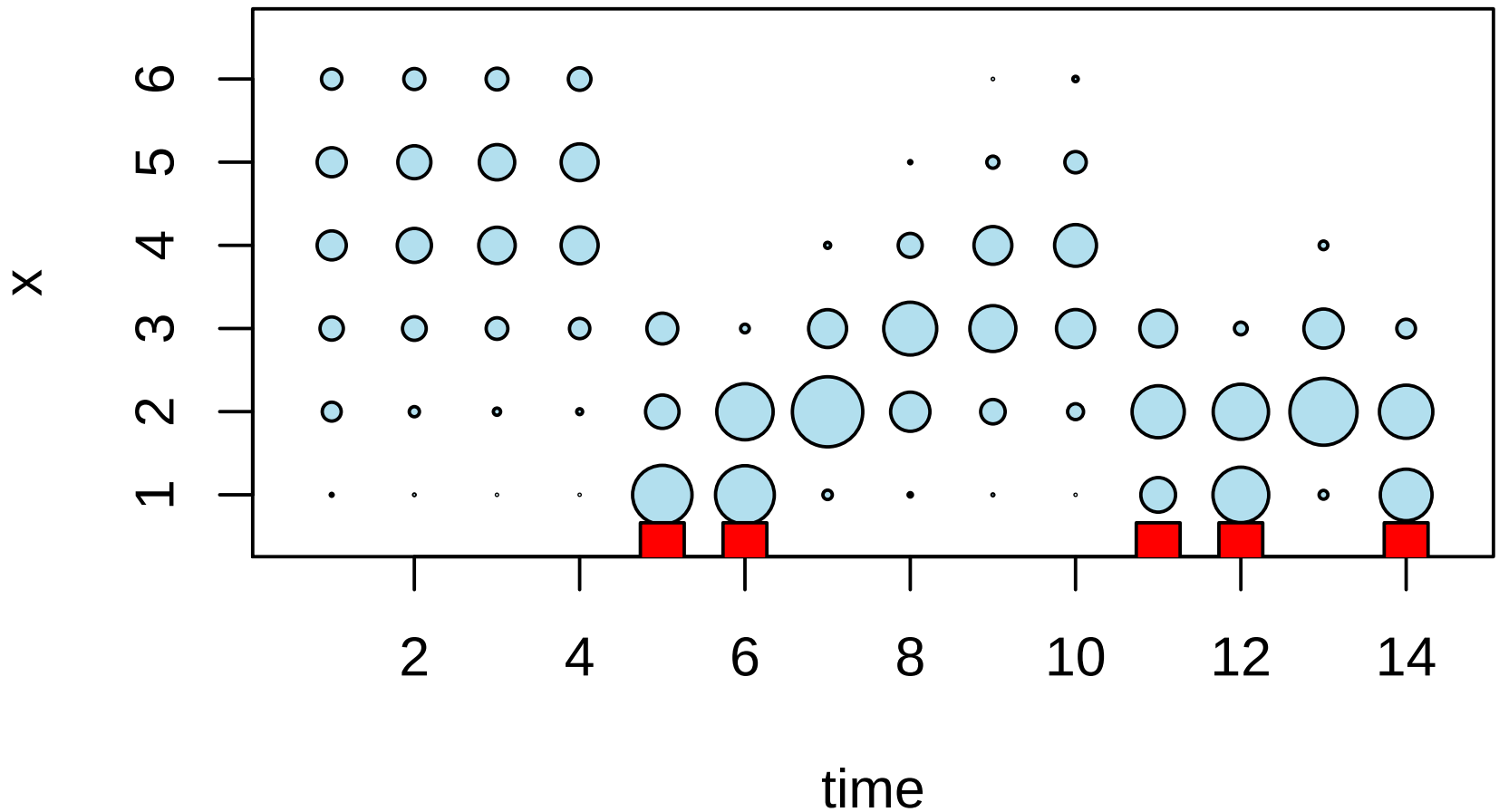
Hidden Markov Models: Inference

Illustration: Filtering

```
alpha_recursion = function(y, mu, A, B)
{
  K = length(mu)
  T = length(y)
  alpha = matrix(0, nrow=T,ncol=K)
  for (j in 1:K) alpha[1,j] = B[j,y[1]] *sum(A[,j]* mu)
  for (t in 2:T) for (j in 1:K) alpha[t,j] = B[j,y[t]] *sum(A[,j]* alpha[t-1,])
  return(alpha)
}
```

Hidden Markov Models: Inference

Illustration: Filtering



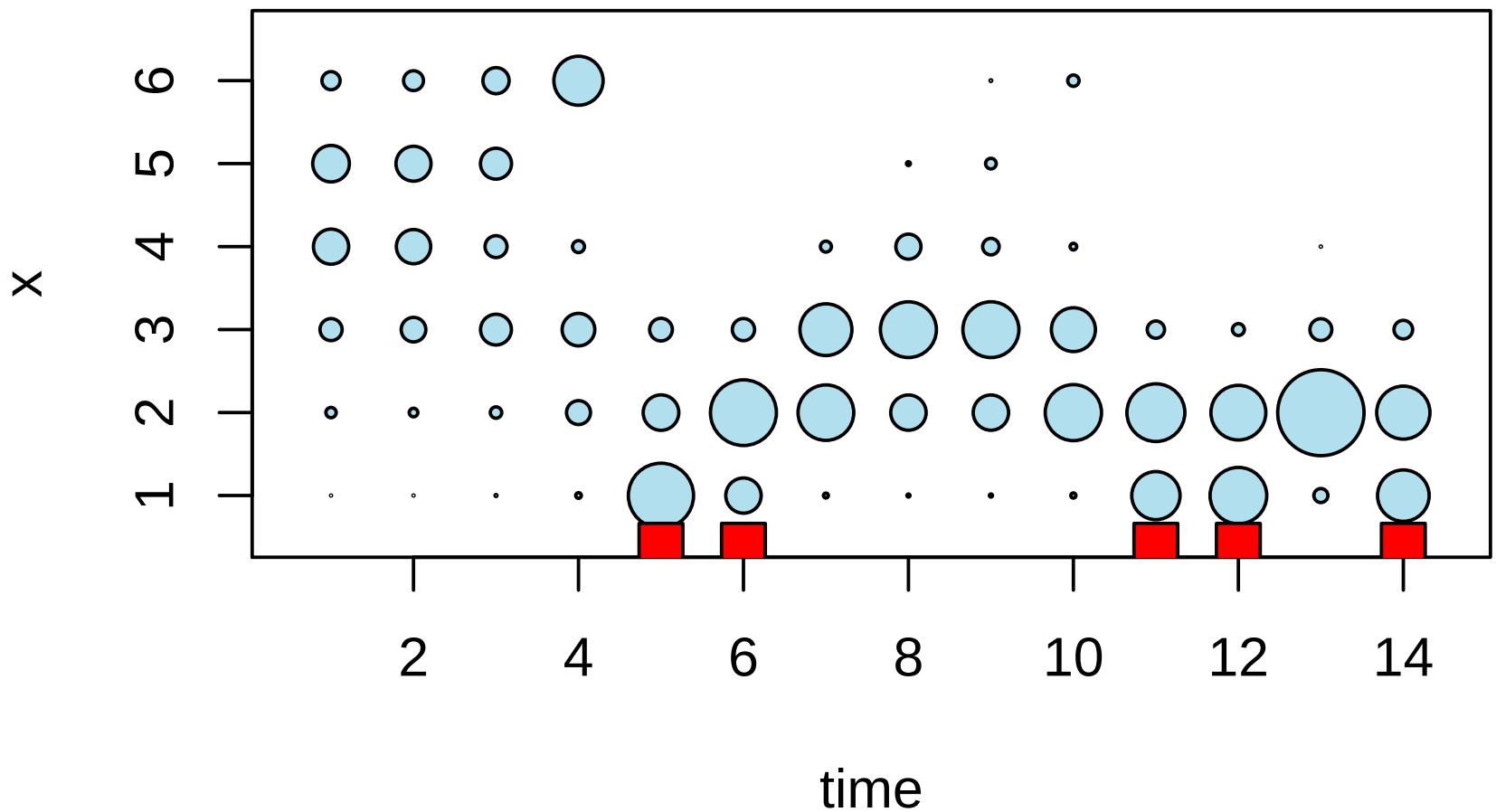
Hidden Markov Models: Inference

Illustration: Smoothing

```
beta_recursion = function(y, mu, A, B)
{
  K = length(mu)
  T = length(y)
  beta = matrix(0, nrow=T,ncol=K)
  for (j in 1:K) beta[T,j] = 1
  for (t in T:2) for (i in 1:K) beta[t-1,i] =  sum(B[,y[t]]*A[i,]* beta[t,])
  return(beta)
}
```

Hidden Markov Models: Inference

Illustration: Smoothing



Hidden Markov Models: Inference

MAP estimation

- We are interested in the Maximum a posterior estimate

$$\hat{x}_{0:T} = \arg\max_{x_{0:T}} p(x_{0:T}|y_{1:T})$$

or equivalently, for fixed $y_{1:T}$

$$\hat{x}_{0:T} = \arg\max_{x_{0:T}} p(x_{0:T}, y_{1:T})$$

- Note that direct optimization would quickly become unfeasible as the number of different state paths is $|\mathcal{X}|^{T+1}$.
- The MAP estimate can be calculated efficiently using the Viterbi algorithm, which uses a backward-forward (or forward-backward) recursion and is a special case of the so-called max-product algorithm.

Hidden Markov Models: Inference

MAP estimation

- First note that

$$p(x_{0:T}, y_{1:T}) = p(x_0) \prod_{t=1}^{T} p(x_t|x_{t-1})p(y_t|x_t)$$

and

$$\begin{aligned}
&\max_{x_{0:T}} p(x_0) \prod_{t=1}^{T} p(x_t|x_{t-1})p(y_t|x_t) \\
&= \max_{x_{0:T-1}} \left\{ \left[p(x_0) \prod_{t=1}^{T-1} p(x_t|x_{t-1})p(y_t|x_t) \right] \max_{x_T} p(x_T|x_{T-1})p(y_T|x_T) \right\} \\
&= \max_{x_{0:T-1}} \left\{ \left[p(x_0) \prod_{t=1}^{T-1} p(x_t|x_{t-1})p(y_t|x_t) \right] m_{T-1}(x_{T-1}) \right\}
\end{aligned}$$

where $m_{T-1}(x_{T-1}) = \max_{x_T} p(x_T|x_{T-1})p(y_T|x_T)$ is the message from the end of the chain to the penultimate timestep.

Hidden Markov Models: Inference

MAP estimation

- We can continue in this manner
- For $t = T-1, \dots, 1$, let

$$m_{t-1}(x_{t-1}) = \max_{x_{t:T}} \left\{ \prod_{k=t}^{T} p(x_k|x_{k-1}) p(y_k|x_k) \right\}$$

  and $m_T(x_T) = 1$.
- $m_{t-1}(x_{t-1})$ satisfies the following backward recursion

$$m_{t-1}(x_{t-1}) = \max_{x_t} p(y_t|x_t) p(x_t|x_{t-1}) m_t(x_t)$$

Hidden Markov Models: Inference

MAP estimation

- Note that

$$p(x_0)m_0(x_0) = \max_{x_{1:T}} p(x_{0:T}, y_{1:T})$$

Hence

$$\begin{aligned}\hat{x}_0 &= \arg\max_{x_0} \left(\max_{x_{1:T}} p(x_0, x_{1:T}, y_{1:T}) \right) \\ &= \arg\max_{x_0} m_0(x_0)p(x_0)\end{aligned}$$

- Similarly,

$$\begin{aligned}\hat{x}_t &= \arg\max_{x_t} \left(\max_{x_{t+1:T}} p(\hat{x}_{0:t-1}, x_t, x_{t+1:T}, y_{1:T}) \right) \\ &= \arg\max_{x_t} \left(\max_{x_{t+1:T}} p(\hat{x}_{t-1}, x_t, x_{t+1:T}, y_{t:T}) \right) \\ &= \arg\max_{x_t} \left(m_t(x_t)p(y_t|x_t)p(x_t|\hat{x}_{t-1}) \right).\end{aligned}$$

Hidden Markov Models: Inference

MAP estimation

Viterbi algorithm with $\mathcal{X} = \{1, \dots, K\}$

- For $i = 1, \dots, K$, set $m_T(i) = 1$.
- For $t = T, \dots, 1$
  - For $i = 1, \dots, K$, let

$$m_{t-1}(i) = \max_{j=1,\dots,K} g_j(y_t) A_{i,j} m_t(j)$$

- Set $\hat{x}_0 = \arg\max_{i=1\dots,K} \; m_0(i)\mu_i$
- For $t = 1, \dots, T$
  - Set

$$\hat{x}_t = \arg\max_{i=1\dots,K} \; m_t(i) g_i(y_t) A_{\hat{x}_{t-1},i}.$$

Hidden Markov Models: Inference

MAP estimation

- The computational complexity of the Viterbi algorithm is $O(T|\mathcal{X}|^2)$, the same as the forward-backward recursion
- For numerical stability, logarithms are computed in practice

Hidden Markov Models: Inference

MAP estimation: Illustration

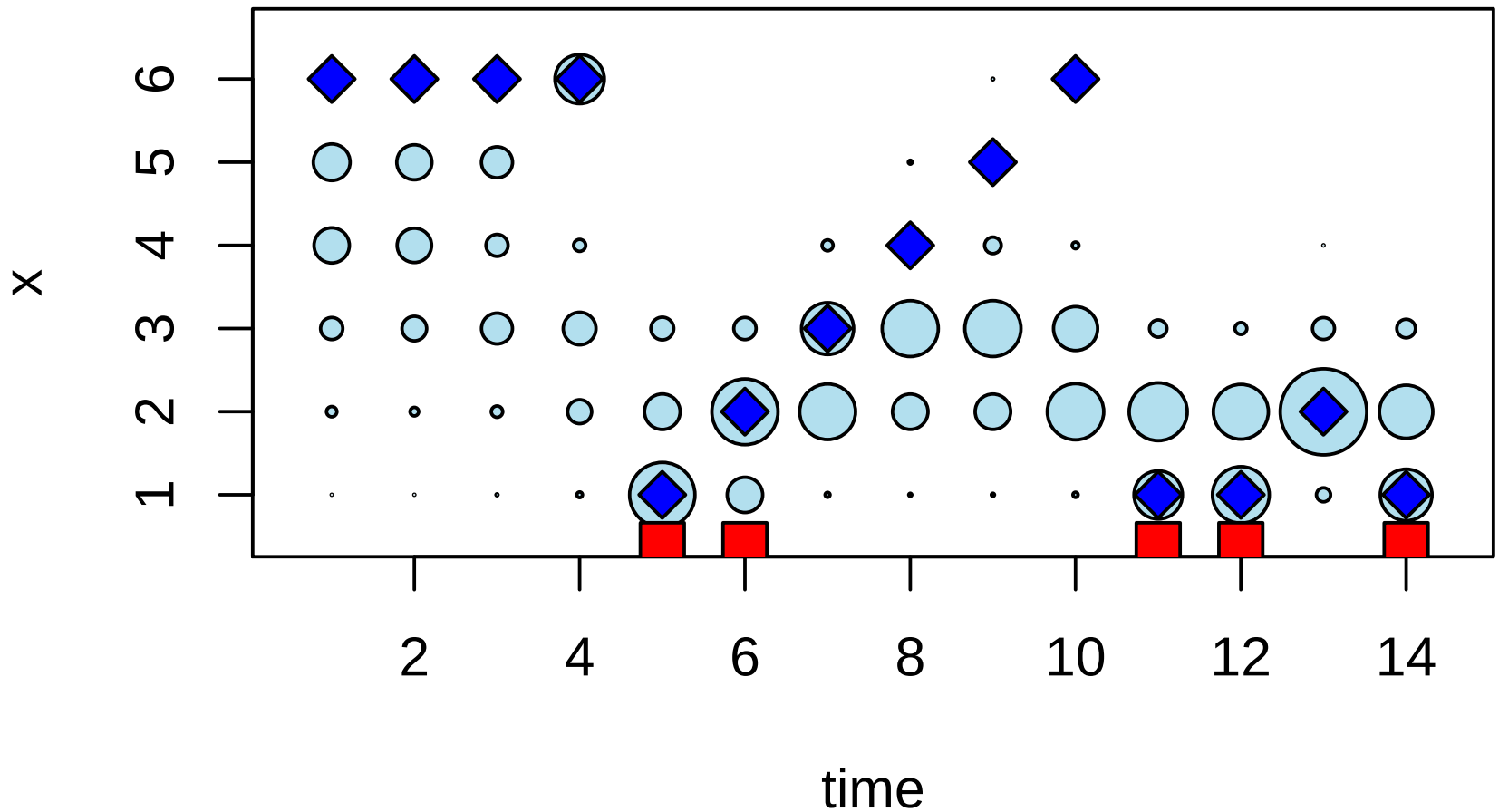
```
viterbi = function(y, mu, A, B)
{
  K = length(mu)
  T = length(y)
  m = matrix(0, nrow=T,ncol=K)
  m0 = matrix(0, nrow=1,ncol=K)
  x.map = rep(0, T)

  # Backward
  for (i in 1:K) m[T,i] = 1
  for (t in T:2) for (i in 1:K) m[t-1,i] =  max(B[,y[t]]*A[i,]* m[t,])
  for (i in 1:K) m0[i] =  max(B[,y[1]]*A[i,]* m[1,])

  #Forward
  x0.map = which.max(m0 * mu)
  x.map[1] = which.max(m[1,]*B[,y[1]]*A[x0.map,])
  for (t in 2:T) x.map[t] = which.max(m[t,]*B[,y[t]]*A[x.map[t-1],])
  return(x.map)
}
```

Hidden Markov Models: Inference

MAP estimation: Illustration



Outline

Hidden Markov Models: Learning

- So far we have assumed that the parameters A, μ and g of the HMMs were known
- This is not the case in general
- We can differentiate two cases
  - The fully observed case: we have a dataset where the hidden states $(x_0, x_1, \ldots, x_T)$ are known
  - The unsupervised case: all we have is the data $(y_1, \ldots, y_T)$ and the hidden variables are not observed
- For simplicity, we only consider estimation of the transition matrix A.

Hidden Markov Models: Learning

Fully observed data

- If the hidden states $(x_0, x_1, \ldots, x_T)$ are known, the parameter A can be fitted using maximum likelihood
- Let $n_{i,j} = \sum_{t=1}^{T} I(x_t = j, x_{t-1} = i)$ be the number of transitions between state i and state j.
- The MLE of $A_{i,j}$ is

$$\widehat{A}_{i,j} = \frac{n_{i,j}}{\sum_{\ell \in \mathcal{X}} n_{i,\ell}}$$

Hidden Markov Models: Learning

Unsupervised case

- If the hidden states are not observed, finding the MLE is much more challenging as we want to optimize

$$\widehat{A} = \arg\max_{A} \ \log p(y_{1:T}; A)$$

- It is possible to derive an iterative algorithm, known as the Baum-Welch algorithm to find the MLE
- The Baum-Welch algorithm is a special case of the Expectation-Maximization algorithm, applied to HMMs

Hidden Markov Models: Learning

Unsupervised case

- EM algorithm
- At iteration k
  - E step

$$Q(A; A^{(k-1)}) = \mathbb{E}[\log p(X_{0:T}, y_{1:T}; A) | y_{1:T}, A^{(k-1)}]$$

  - M step

$$A^{(k)} = \arg\max_{A} Q(A; A^{(k-1)})$$

- Each iteration increases the value of the log-likelihood

Hidden Markov Models: Learning

Unsupervised case

- The prior pmf can be expressed as

$$p(x_{0:T}; A) = \mu_{x_0} \prod_{i,j \in \mathcal{X}} A_{i,j}^{n_{i,j}}$$

- The log joint pmf can be expressed as

$$\log p(x_{0:T}, y_{1:T}; A) = \log \mu_{x_0} + \sum_{i,j \in \mathcal{X}} n_{i,j} \log A_{i,j} + \sum_{t=1}^{T} \log g_{y_t}(x_t)$$

Hidden Markov Models: Learning

Unsupervised case

- The Q function of the EM is thus expressed as

$$\begin{aligned} Q(A; A^*) &= \mathbb{E}[\log p(X_{0:T}, y_{1:T}; A) | y_{1:T}, A^*] \\ &= \sum_{i,j \in \mathcal{X}} \mathbb{E}[N_{i,j} | y_{1:T}, A^*] \log A_{i,j} + C \end{aligned}$$

where C is a constant independent of A and

$$N_{i,j} = \sum_{t=1}^{T} I(X_t = j, X_{t-1} = i).$$

The expected counts can be expressed as

$$\mathbb{E}[N_{i,j} | y_{1:T}, A^*] = \sum_{t=1}^{T} \mathbb{P}(X_t = j, X_{t-1} = i | y_{1:T}; A^*)$$

Hidden Markov Models: Learning

Unsupervised case

- The terms $p(x_t, x_{t-1}|y_{1:T}; A^*)$ can be obtained from the forward-backward recursion, as [check!]

$$p(x_t, x_{t-1}|y_{1:T}; A^*) \propto \alpha_{t-1}(x_{t-1})p(y_t|x_t)p(x_t|x_{t-1})\beta_t(x_t)$$

- The M step gives

$$A_{i,j}^{(k)} = \arg\max_{A_{i,j}} \mathbb{E}[N_{i,j}|y_{1:T}, A^{(k-1)}] \log A_{i,j}$$

$$= \frac{\mathbb{E}[N_{i,j}|y_{1:T}, A^{(k-1)}]}{\sum_\ell \mathbb{E}[N_{i,\ell}|y_{1:T}, A^{(k-1)}]}$$

Outline

Outline

Recap: multivariate Gaussian random variables

- The probability density function of a multivariate Gaussian random variable $X \in \mathbb{R}^{d_x}$ with mean μ and covariance matrix Σ is given as

$$\mathcal{N}(x; \mu, \Sigma) := \frac{1}{(2\pi)^{d_x/2}\sqrt{|\Sigma|}} \exp\left\{-\frac{1}{2}(x-\mu)^\mathsf{T}\Sigma^{-1}(x-\mu)\right\}.$$

- Notations: For X, Y continuous random variables, we write $p(x)$, $p(x, y)$ and $p(x|y)$ for the pdf of X, (X, Y) and $X|Y = y$

Recap: multivariate Gaussian random variables

- Let (X, Y), $X \in \mathbb{R}^{d_x}$ and $Y \in \mathbb{R}^{d_y}$, be a jointly Gaussian vector with mean and covariance matrix

$$\mu = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}, \Sigma = \begin{pmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{pmatrix}.$$

Then the marginals are given by

$$X \sim \mathcal{N}(\mu_x, \Sigma_{xx})$$
$$Y \sim \mathcal{N}(\mu_y, \Sigma_{yy})$$

and the conditional

$$X|Y = y \sim \mathcal{N}(\mu_{x|y}, \Sigma_{x|y})$$

where

$$\Sigma_{x|y} = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}$$
$$\mu_{x|y} = \mu_x + \Sigma_{xy}\Sigma_{yy}^{-1}(y - \mu_y)$$

Recap: multivariate Gaussian random variables

- Consider Gaussian random variables $X \in \mathbb{R}^{d_x}$ and $Y \in \mathbb{R}^{d_y}$ with

$$X \sim \mathcal{N}(\mu_x, \Sigma_{xx})$$
$$Y|X = x \sim \mathcal{N}(Ax + b, \Sigma_{y|x})$$

where $\mu_x \in \mathbb{R}^{d_x}$, Σ_{xx} is a $d_x \times d_x$ covariance matrix, A is a $d_y \times d_x$ matrix, b is a d_y vector and $\Sigma_{y|x}$ is a $d_y \times d_y$ covariance matrix. Then

$$(X|Y = y) \sim \mathcal{N}\left(\mu_{x|y}, \Sigma_{x|y}\right)$$
$$Y \sim \mathcal{N}\left(\mu_y, \Sigma_{yy}\right)$$

where

$$\mu_y = A\mu_x + b$$
$$\Sigma_{yy} = \Sigma_{y|x} + A\Sigma_{xx}A^{\mathsf{T}}$$
$$\Sigma_{x|y} = \left(\Sigma_{xx}^{-1} + A^{\mathsf{T}}\Sigma_{y|x}^{-1}A\right)^{-1}$$
$$\mu_{x|y} = \Sigma_{x|y}\left(\Sigma_{xx}^{-1}\mu_x + A^{\mathsf{T}}\Sigma_{y|x}^{-1}(y - b)\right)$$

Outline

Linear Gaussian state-space models

- $(X_0, \ldots, X_T)$, $X_t \in \mathbb{R}^{d_x}$ be the hidden states
- $(Y_1, \ldots, Y_T)$, $Y_t \in \mathbb{R}^{d_y}$ are observations
- Linear Gaussian state-space model, for $t = 1, \ldots, T$

$$X_t = F_t X_{t-1} + G_t V_t \qquad \text{State model}$$
$$Y_t = H_t X_t + W_t \qquad \text{Observation model}$$

where the random variables $(X_0, V_1, V_2, \ldots, V_T, W_1, W_2, \ldots, W_T)$ are independent with $X_0 \sim \mathcal{N}(\mu_0, \Sigma_0)$ and for $t = 1, \ldots, T$,

$$V_t \sim \mathcal{N}(0, Q_t), \quad W_t \sim \mathcal{N}(0, R_t)$$

with

- V_t is the state noise at time t
- W_t is the observation noise at time t
- F_t is the $d_x \times d_x$ state transition matrix
- G_t is the $d_x \times d_v$ noise transfer matrix
- H_t is the $d_y \times d_x$ observation matrix

Linear Gaussian state-space models

- Under the above assumptions, the sequence $(X_0, X_1, Y_1, \ldots, X_T, Y_T)$ is a (continuous-state) hidden Markov model
- If $G_t Q_t G_t^\mathsf{T}$ has full rank, the joint pdf $p(x_{0:T}, y_{1:T})$ factorizes as

$$p(x_{0:T}, y_{1:T}) = p(x_0) \prod_{t=1}^{T} p(y_t|x_t) p(x_t|x_{t-1})$$

where

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t; F_t x_{t-1}, G_t Q_t G_t^\mathsf{T})$$
$$p(y_t|x_t) = \mathcal{N}(y_t; H_t x_t, R_t)$$

Linear Gaussian state-space models

Example: Object tracking

- Let $X_t = (P_t^x, P_t^y, P_t^z, V_t^x, V_t^y, V_t^z)^\mathsf{T}$ denote the position and velocity of an object at time index $t = 0, 1, \ldots$.
- The position and velocity are not directly observed, but a GPS delivers noisy observations of the position

$$Y_t = (P_t^x, P_t^y, P_t^z)^\mathsf{T} + W_t$$

  where $W_t \sim \mathcal{N}(0, R)$.
- White noise acceleration model

$$P_t^x = P_{t-1}^x + \delta V_{t-1}^x + \frac{\delta^2}{2} A_{t-1}^x$$
$$V_t^x = V_{t-1}^x + \delta A_{t-1}^x$$

  where $\delta = 1$ here, A_{t-1}^x is the unknown acceleration at time t, assumed to be Gaussian with zero mean and variance Q^x, and similarly for the other coordinates.

Linear Gaussian state-space models

Example: Object tracking

- Dynamic linear Gaussian model with

$$F = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, G = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

  and $V_t = (A^x_{t-1}, A^y_{t-1}, A^z_{t-1})^\mathsf{T}$.

- Objective is to calculate $p(x_t|y_{1:t})$ at each time t.

Linear Gaussian state-space models

Example: Linear regression with time-varying coefficients

- Let (z_t, Y_t), $t = 1, \ldots, T$ where $z_t \in \mathbb{R}^p$ are covariates and $Y_t \in \mathbb{R}$ are response variables
- Linear relation between the response and the covariate
- Regression coefficients are assumed to evolve over time

$$\beta_t = \beta_{t-1} + V_t$$
$$Y_t = z_t \beta_t + W_t$$

where $\beta_t \in \mathbb{R}^p$ is the regressor at time t, and V_t is a vector of independent Gaussian random variables with variance σ_V^2.

Outline

Inference in Linear Gaussian state-space models

Kalman Filter

- Filtering pdf $p(x_t|y_{1:t})$ of the hidden state X_t given observations $y_{1:t}$ up to time t.
- Let

$$\begin{aligned}
\mu_{t|t-1} &:= \mathbb{E}[X_t|Y_{1:t-1} = y_{1:t-1}] \\
\Sigma_{t|t-1} &:= \mathbb{E}[(X_t - \mu_{t|t-1})(X_t - \mu_{t|t-1})^{\mathsf{T}}|Y_{1:t-1} = y_{1:t-1}] \\
\mu_{t|t} &:= \mathbb{E}[X_t|Y_{1:t} = y_{1:t}] \\
\Sigma_{t|t} &:= \mathbb{E}[(X_t - \mu_{t|t})(X_t - \mu_{t|t})^{\mathsf{T}}|Y_{1:t} = y_{1:t}]
\end{aligned}$$

Inference in Linear Gaussian state-space models

Kalman Filter

- Let $p(x_t|y_{1:t})$ and $p(x_t|y_{1:t-1})$ be the filtering and one-step predictive pdfs at time t. Then

$$p(x_t|y_{1:t-1}) = \mathcal{N}(x_t; \mu_{t|t-1}, \Sigma_{t|t-1})$$
$$p(x_t|y_{1:t}) = \mathcal{N}(x_t; \mu_{t|t}, \Sigma_{t|t})$$

- $(\mu_{t|t-1}, \Sigma_{t|t-1})$ and $(\mu_{t|t}, \Sigma_{t|t})$ follow a two-step recursion

Inference in Linear Gaussian state-space models

Kalman Filter

- Prediction Step

$$\mu_{t|t-1} = F_t \mu_{t-1|t-1}$$
$$\Sigma_{t|t-1} = F_t \Sigma_{t-1|t-1} F_t^\mathsf{T} + G_t Q_t G_t^\mathsf{T}$$

Inference in Linear Gaussian state-space models

Kalman Filter

- Update/correction step

$$\mu_{t|t} = \mu_{t|t-1} + K_t \nu_t$$
$$\Sigma_{t|t} = (I - K_t H_t)\Sigma_{t|t-1}$$

where ν_t is the residual or innovation

$$\nu_t := y_t - \hat{y}_{t|t-1}$$
$$\hat{y}_{t|t-1} := \mathbb{E}[Y_t | Y_{1:t-1} = y_{1:t-1}] = H_t \mu_{t|t-1}$$

and K_t is the **Kalman gain**

$$K_t = \Sigma_{t|t-1} H_t^\mathsf{T} S_t^{-1}$$

with

$$S_t := \mathbb{E}[(Y_t - \hat{y}_{t|t-1})(Y_t - \hat{y}_{t|t-1})^\mathsf{T} | Y_{1:t-1} = y_{1:t-1}]$$
$$= H_t \Sigma_{t|t-1} H_t^\mathsf{T} + R_t$$

Inference in Linear Gaussian state-space models

Kalman Smoother

- We are now interested in the smoothing pdfs $p(x_t|y_{1:T})$ of the hidden state X_t given all the observations $y_{1:T}$.
- Let

$$\mu_{t|T} := \mathbb{E}[X_t|Y_{1:T} = y_{1:T}]$$
$$\Sigma_{t|T} := \mathbb{E}[(X_t - \mu_{t|T})(X_t - \mu_{t|T})^\mathsf{T}|Y_{1:T} = y_{1:T}]$$

- We can obtain the smoothing pdfs by
  1. Run the forward recursion of the Kalman filter, in order to obtain $(\mu_{t|t}, \Sigma_{t|t})$ for $t = 1, \ldots, T$, and
  2. Run a backward recursion.

Inference in Linear Gaussian state-space models

Kalman Smoother

- Let $p(x_t|y_{1:T})$ be the smoothing pdf at time t. Then

$$p(x_t|y_{1:T}) = \mathcal{N}(x_t; \mu_{t|T}, \Sigma_{t|T})$$

where $(\mu_{t|T}, \Sigma_{t|T})$ follow the recursion

$$\begin{aligned}\mu_{t|T} &= \mu_{t|t} + J_t(\mu_{t+1|T} - \mu_{t+1|t}) \\ \Sigma_{t|T} &= \Sigma_{t|t} + J_t(\Sigma_{t+1|T} - \Sigma_{t+1|t})J_t^{\mathsf{T}}\end{aligned}$$

where J_t is the **backward Kalman gain**

$$J_t = \Sigma_{t|t}F_{t+1}^{\mathsf{T}}\Sigma_{t+1|t}^{-1}.$$

Inference in Linear Gaussian state-space models

Example

- Consider the following simple scalar example of a random walk observed in noise

$$X_t = X_{t-1} + V_t$$
$$Y_t = X_t + W_t$$

where $X_0 \sim \mathcal{N}(0, 1)$, $V_t \sim \mathcal{N}(0, Q)$, $W_t \sim \mathcal{N}(0, R)$ where $Q = 0.02$ and $R = 0.2$.

Inference in Linear Gaussian state-space models

Example

- Prediction

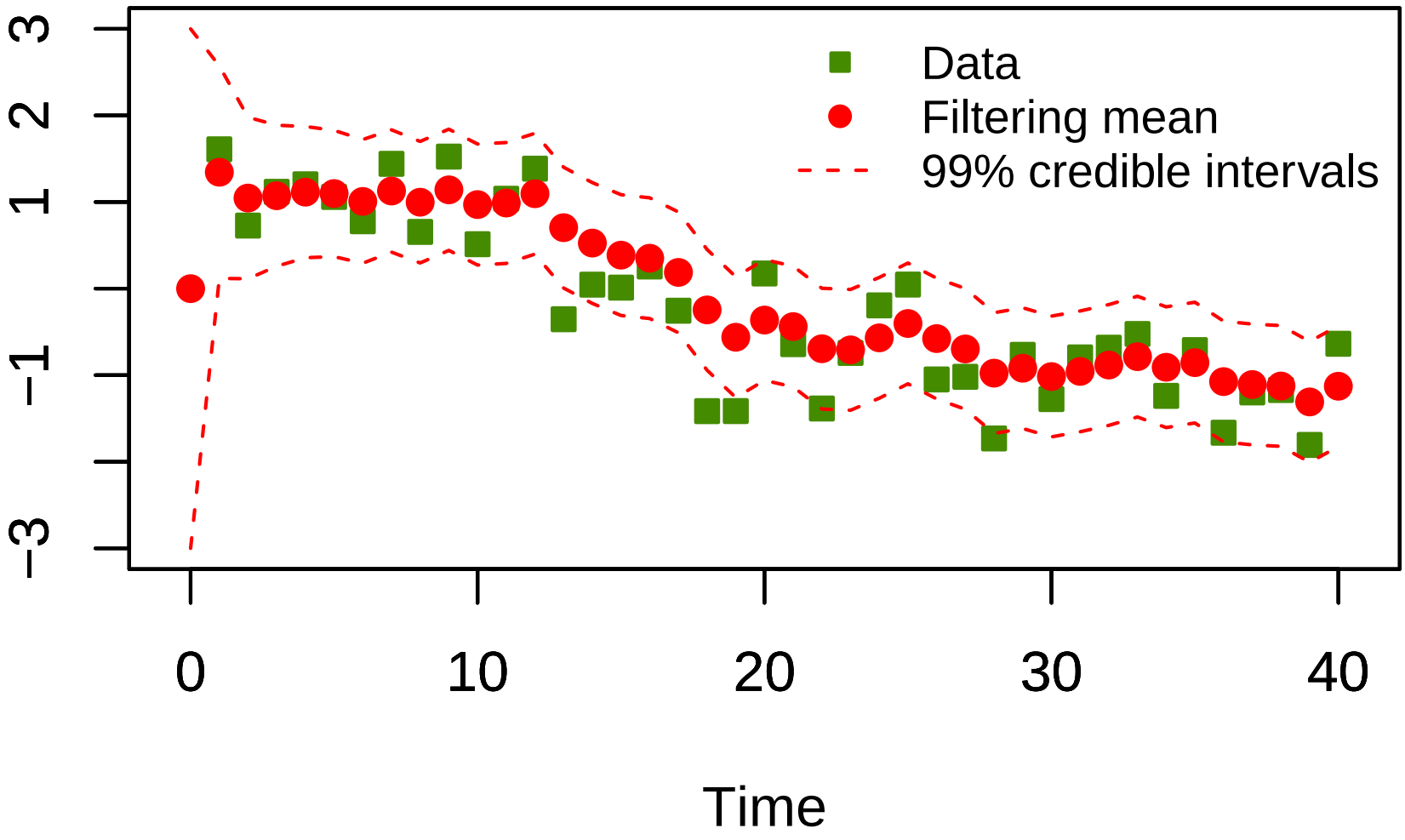
$$\mu_{t|t-1} = \mu_{t-1|t-1}$$
$$\Sigma_{t|t-1} = \Sigma_{t-1|t-1} + Q$$

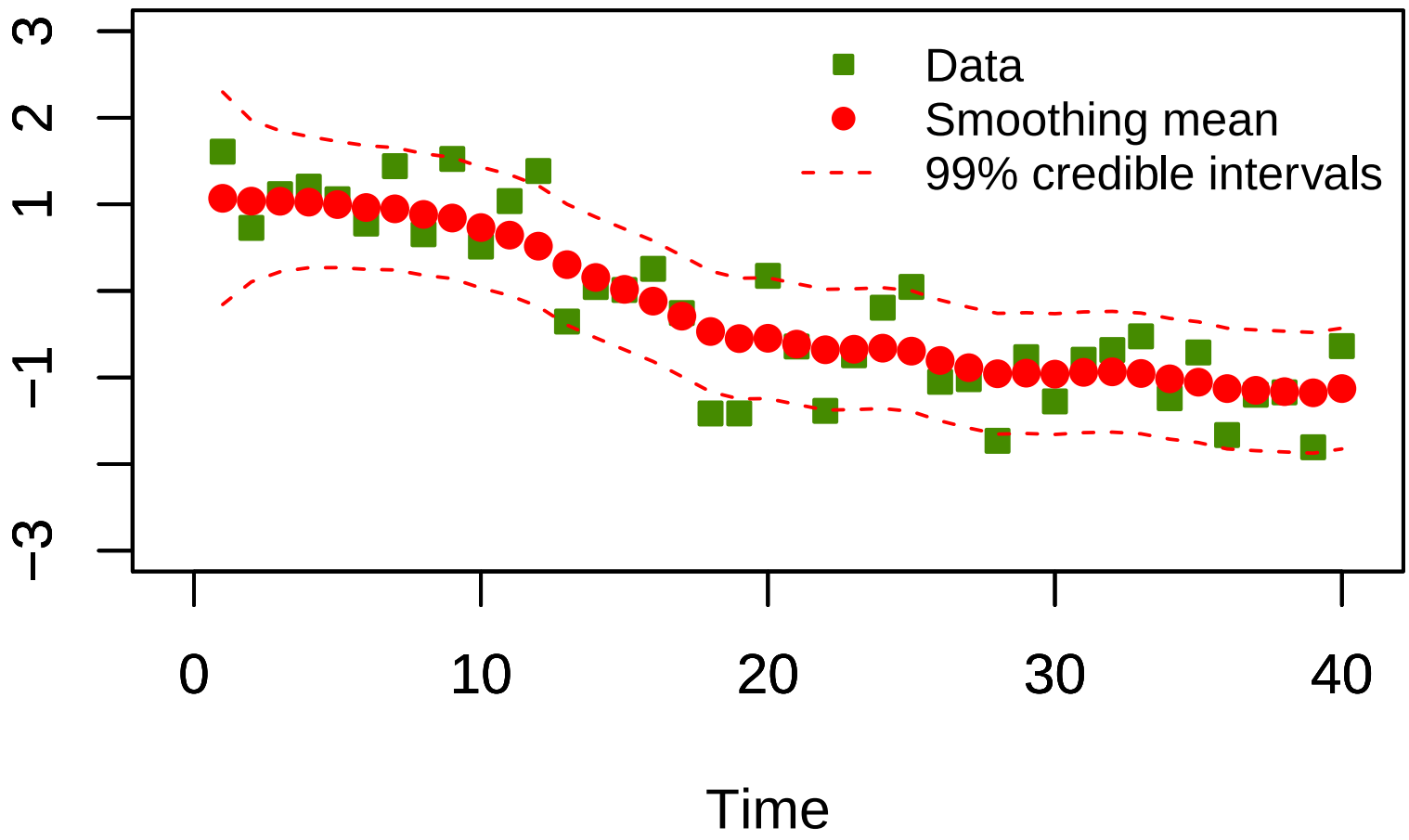
- Update

$$K_t = \frac{\Sigma_{t|t-1}}{\Sigma_{t|t-1} + R}$$

and

$$\Sigma_{t|t} = \frac{R\Sigma_{t|t-1}}{\Sigma_{t|t-1} + R}$$
$$\mu_{t|t} = \frac{R}{\Sigma_{t|t-1} + R}\mu_{t|t-1} + \frac{\Sigma_{t|t-1}}{\Sigma_{t|t-1} + R}y_t$$


Data
Filtering mean
99% credible intervals
3
2
1
−1
−3
0
10
20
30
40
Time


Data
Smoothing mean
99% credible intervals
3
2
1
−1
−3
0
10
20
30
40
Time