Lecture 7:

Hidden Markov Models

Markus Hohle

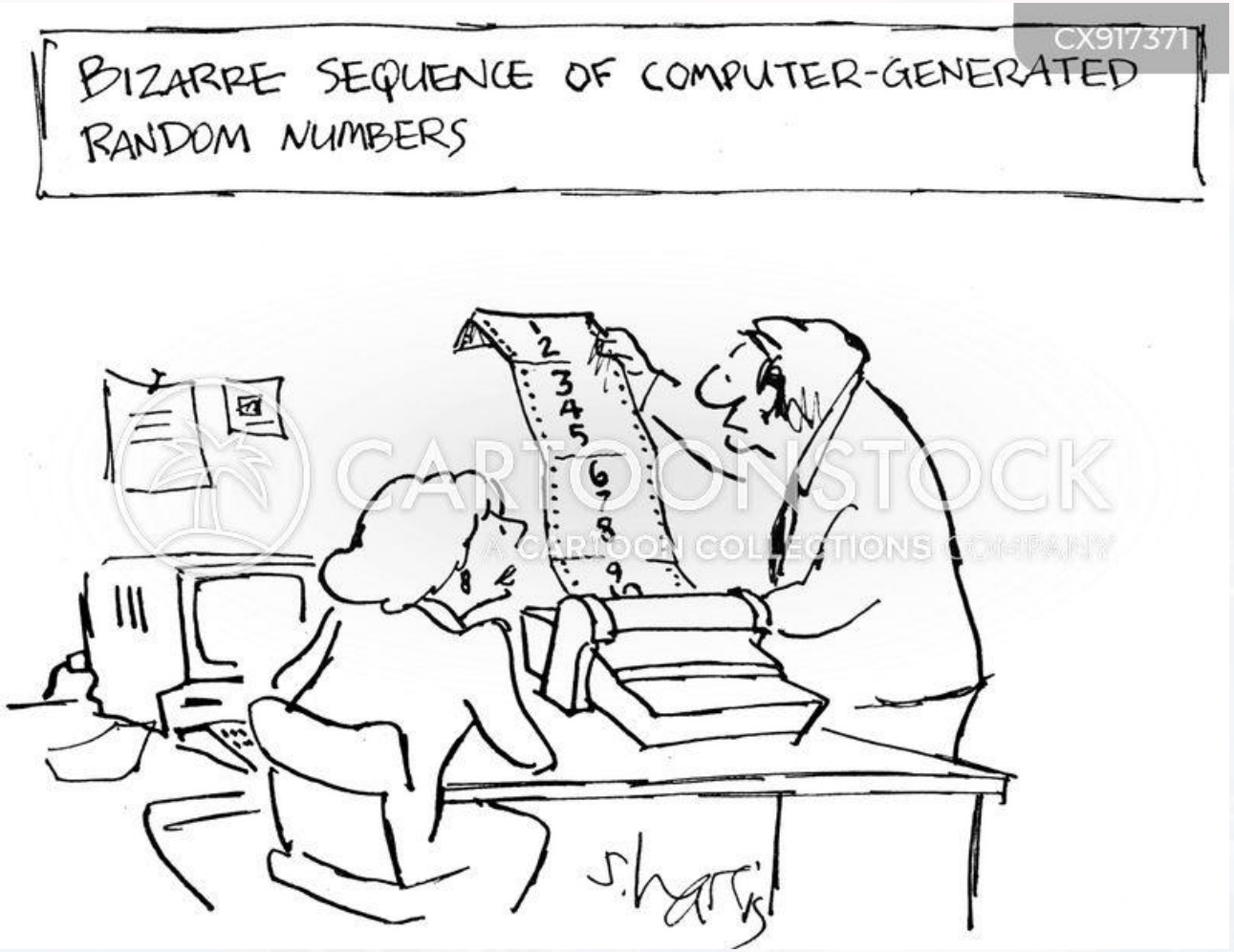
University California, Berkeley

Bayesian Data Analysis and Machine Learning for Physical Sciences

Course Map

Module 1	Maximum Entropy and Information, Bayes Theorem
Module 2	Naive Bayes, Bayesian Parameter Estimation, MAP
Module 3	MLE, Lin Regression
Module 4	Model selection I: Comparing Distributions
Module 5	Model Selection II: Bayesian Signal Detection
Module 6	Variational Bayes, Expectation Maximization
Module 7	**Hidden Markov Models**
Module 8	Stochastic Processes
Module 9	Machine Learning Overview, Supervised Methods
Module 10	Unsupervised Methods
Module 11	ANN: Perceptron, Backpropagation
Module 12	ANN: Basic Architecture, Regression vs Classification, Backpropagation again
Module 13	Convolution and Image Classification and Segmentation
Module 14	Graphs and GNNs
Module 15	RNNs and LSTMs
Module 16	Transformer and LLMs

Hidden Markov Models



Outline

Introduction

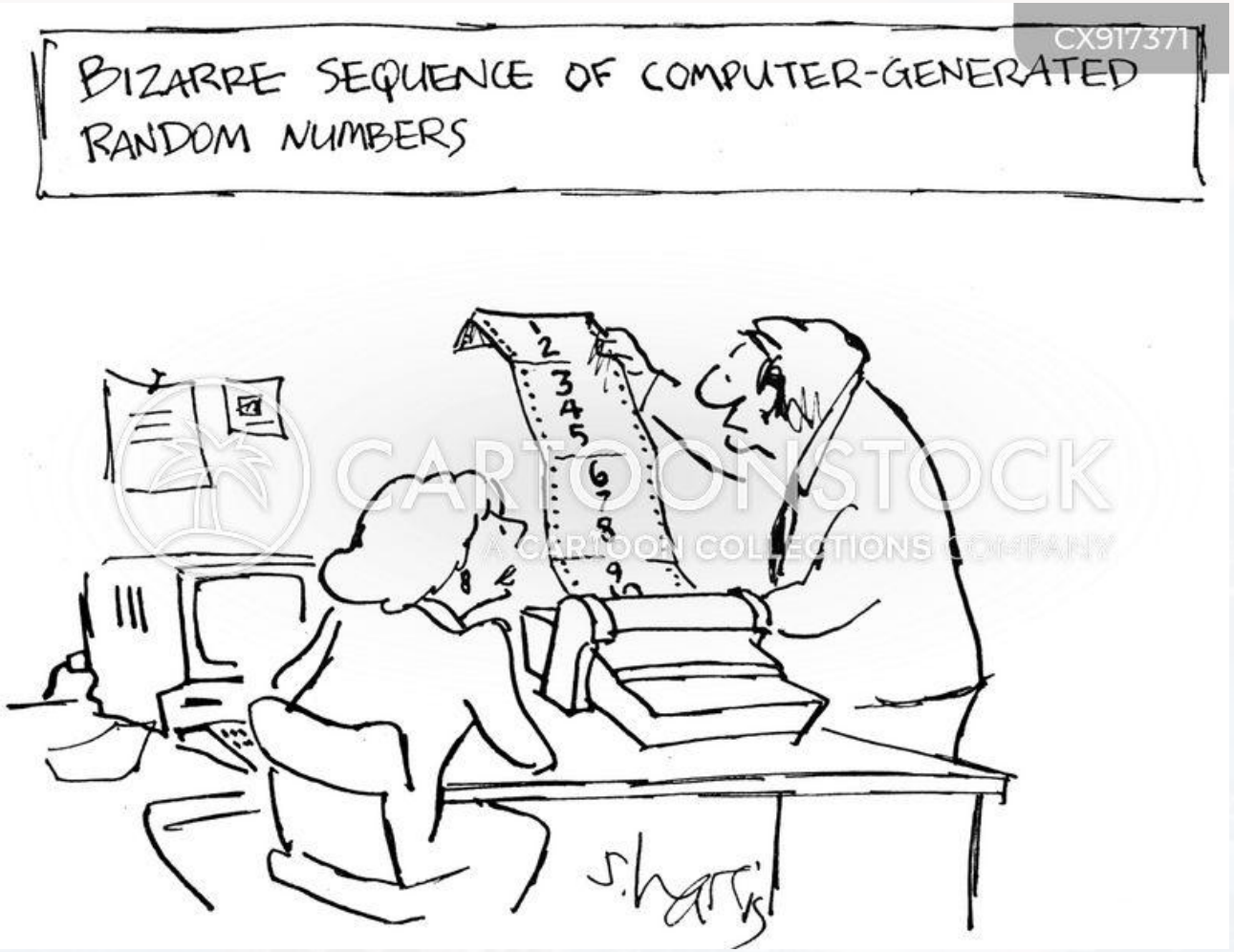
HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples

Hidden Markov Models



Outline

Introduction

HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples

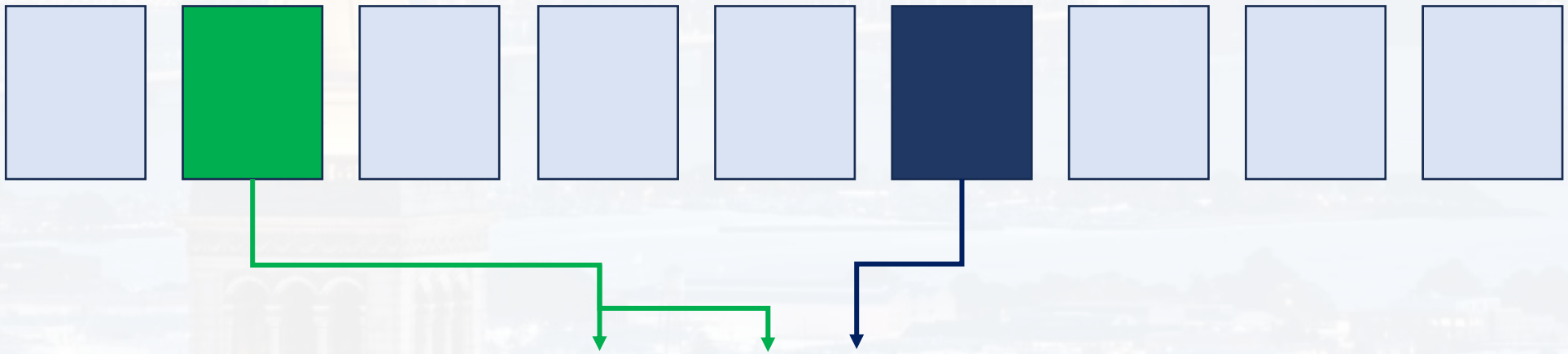
Hidden Markov Models:

applications:

- motif finding
- regulation/transcription processes
- diffusion, ligand binding
- drug development
- anomaly detection
- speech recognition
- saving process on hard drives
- economics

each state *n* draws O_t from a distribution $L_n(O_{t,n}|\theta)$ with parameters θ

set of different states/classes

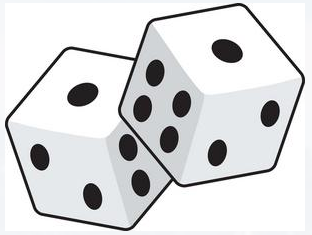


a set of *T* observations *O* $O = (O_1, O_2, O_3, \ldots O_t, \ldots O_T)$ drawing randomly from the states

Hidden Markov Models:

applications:

- motif finding
- regulation/transcription processes
- diffusion, ligand binding
- drug development
- anomaly detection
- speech recognition
- saving process on hard drives
- economics



sequence of observations:

3 3 5 5 2 3 3 4 5 5 2 5 4 1 1 3 6 3 4 2 5 2 4 5 6 6 4 1 1 2 6 2 5 2 6 3 2 2 2 2 1 2 3

Was the die biased towards 2?

goal: finding the sequence of **hidden** states (fair/biased die) for each time *t*

GGATTATATAACCGACGCGCATATGCGCGTTATCTGTCATATAGCATGCGCGCGCGCGGCAACAGTATA

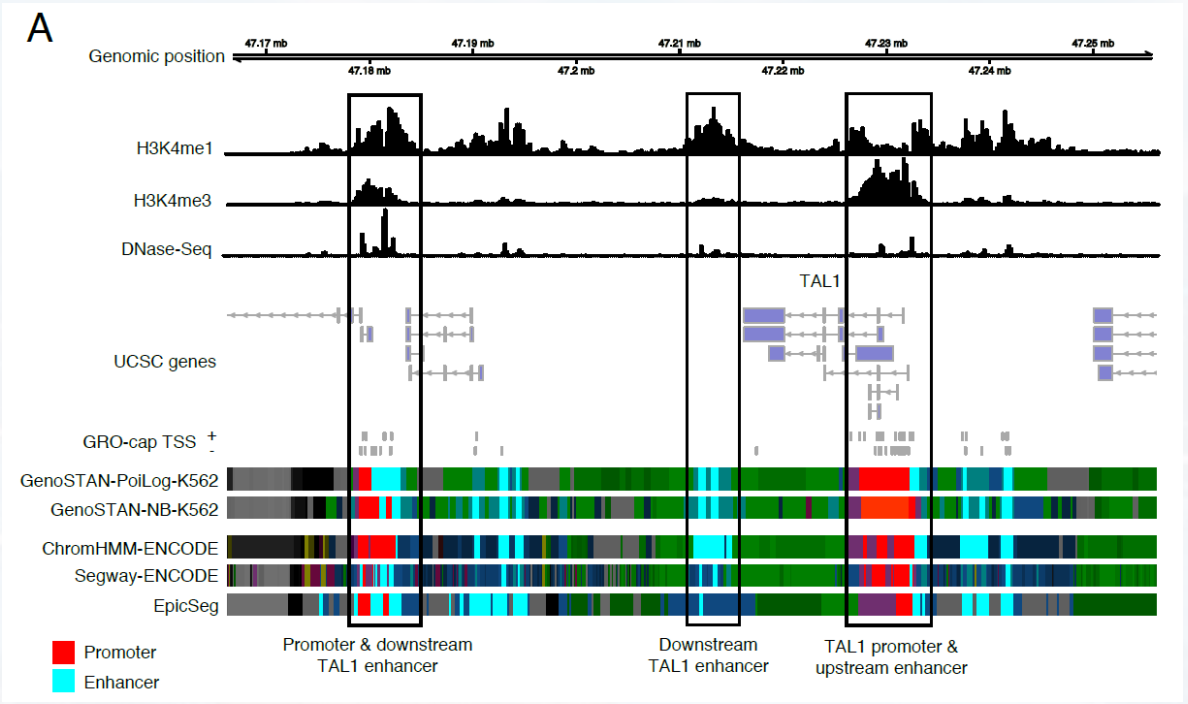
Is this a TATA box?

Is this a CpG island?

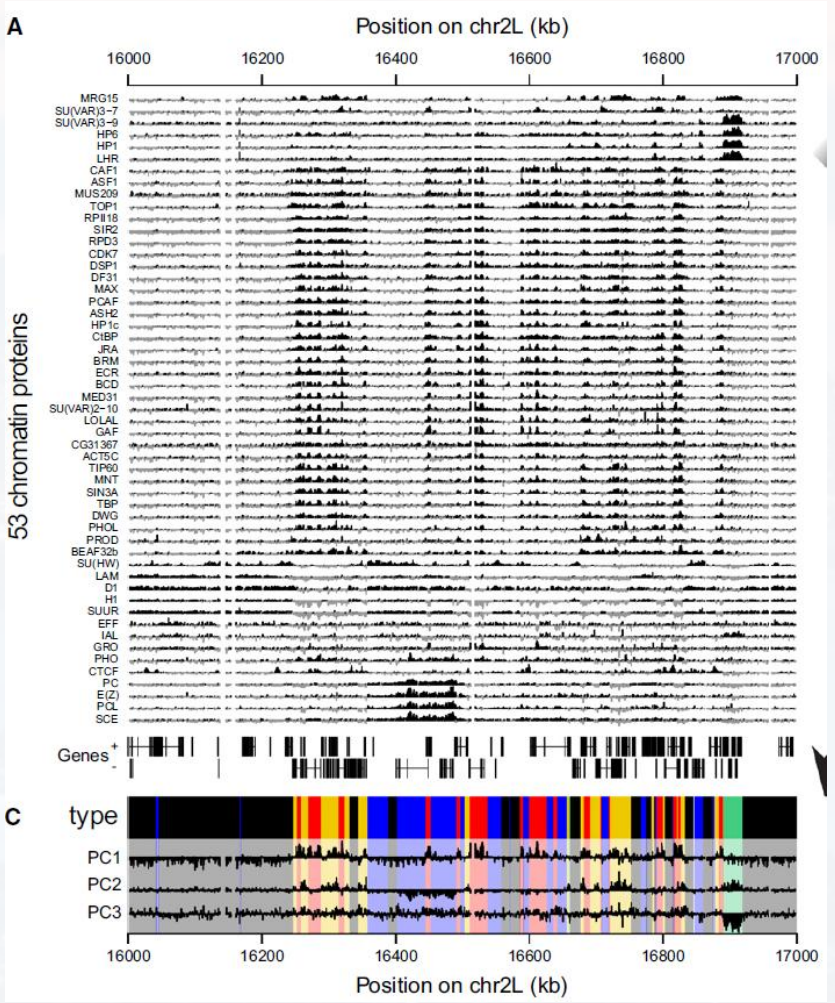
Hidden Markov Models:

applications:

- motif finding
- regulation/transcription processes
- drug development

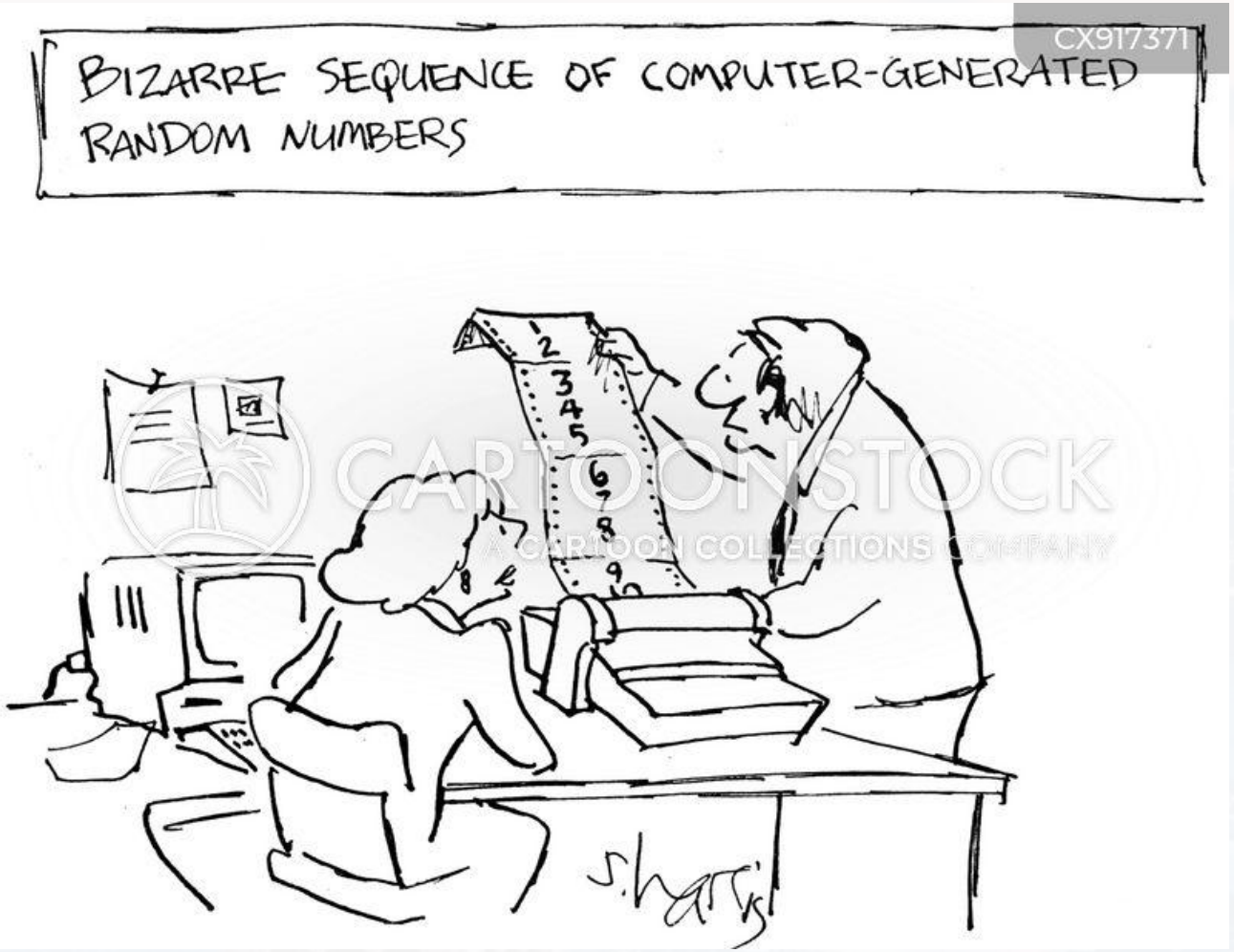


Zacher et al. 2016



Filion et al. 2010

Hidden Markov Models



Outline

Introduction

HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples

Wiki says: *"Markov chain or Markov process is a **stochastic process** describing a sequence of possible events in which the **probability of each event depends only on the state** attained in the **previous event**."*

- N number of states s_n
- time t
- q_t: actual state at time point t; note $q_t \; not \; identical \; s_n$

Markov Chain:

$$P(q_t = s_j | q_{t-1} = s_i, q_{t-2} = s_k, \dots, q_1 = s_l) \qquad 1 \le i, j, k, l \le N$$

1st order Markov Chain

$$= P(q_t = s_j | q_{t-1} = s_i) \qquad 1 \le t \le T$$

transition probability a_{ij} $\qquad A = \{a_{ij}\}$

$$\sum_{j=1}^{N} a_{ij} = 1$$

We assume that a_{ij} is not a function of t

N:	number of states s_n
t:	time
q_t:	actual state at time point t

Markov Chain:

$$P(q_t = s_j | q_{t-1} = s_i, q_{t-2} = s_k, \dots, q_1 = s_l)$$

$$1 \le i, j, k, l \le N$$

$$1 \le t \le T$$

1st order Markov Chain

$$= P(q_t = s_j | q_{t-1} = s_i)$$

$$A = \{a_{ij}\}$$

transition probability a_{ij}

$$\sum_{j=1}^{N} a_{ij} = 1$$

the Markov chain needs to start with some **initial state**:

$$\pi_i = P(q_1 = s_i)$$

$$\sum_{i=1}^{N} \pi_i = 1$$

N:	number of states s_n
t:	time
q_t:	actual state at time point t
π_i:	$P(q_1 = s_i)$
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

Markov Chain:

each state produces an **observable output** which is drawn from a set V (aka **the alphabet**) of M symbols:

$$V = \{V_1, \dots, V_m, \dots, V_M\} \qquad 1 \le m \le M$$

Where $B = \{b_j(m)\}$ is a set of **emission probabilities** $b_j(m)$

$$b_j(m) = P(V_m, t | q_t = s_j)$$

these emissions are the **observations**

$$O = (O_1, O_2, O_3, \dots O_t, \dots O_T)$$

the **model** $\lambda = (A, B, \Pi) = (\{a_{ij}\}, \{b_j(m)\}, \{\pi_i\})$

$$1 \le i, j, k, l \le N$$

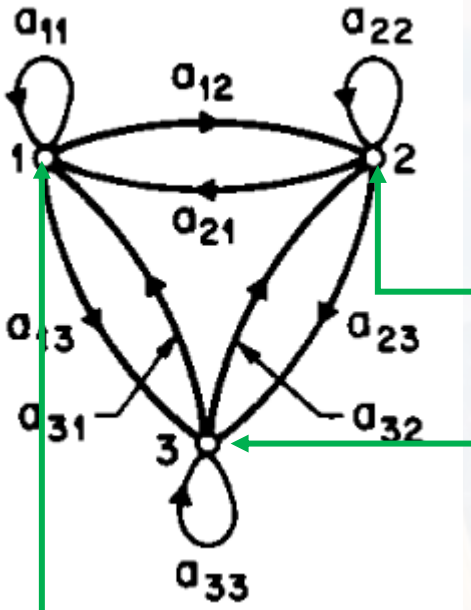
$$1 \le t \le T$$

$$A = \{a_{ij}\}$$

$$\sum_{j=1}^{N} a_{ij} = 1$$

$$\sum_{i=1}^{N} \pi_i = 1$$

Markov Chain:



three coins with different θ for tails H and $1-\theta$ for heads T

different states S_1, S_2 and S_3

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

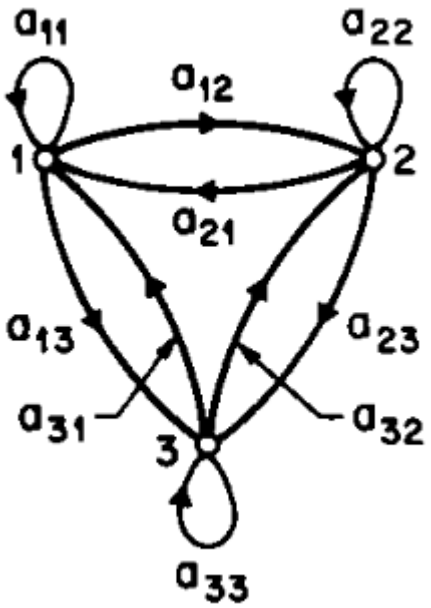
$$V = \{H, T\}$$

	S_1	S_2	S_3
$b_j(m=1)$	θ_1	θ_2	θ_3
$b_j(m=2)$	$1-\theta_1$	$1-\theta_2$	$1-\theta_3$

$$O = (H, H, T, T, H, T, H, H, T, T, H)$$

$$S = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)$$

Markov Chain:



$O = (H, H, T, T, H, T, H, H, T, T, H$

$S = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

problems to solve:

1) find model parameter λ that best explain the observations

$$\lambda_{best} = \underset{\lambda}{argmax} \{P(O|\lambda)\}$$

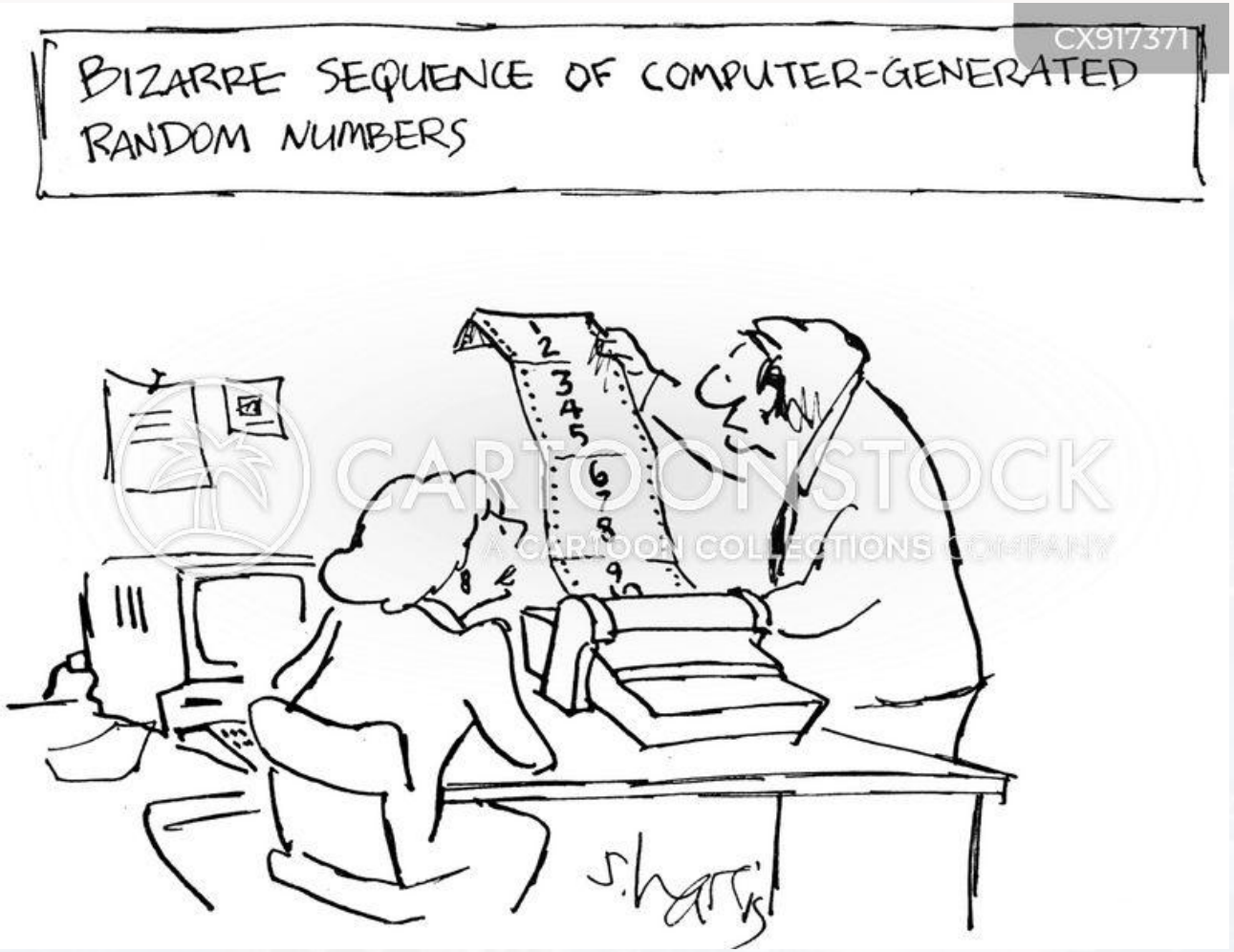
→ EM again!

2) once we have λ_{best}, find the sequence of states S that best explain the observations

$$S_{best} = \underset{S}{argmax} \{P(S|O, \lambda)\}$$

→ Viterbi algorithm

Hidden Markov Models



Outline

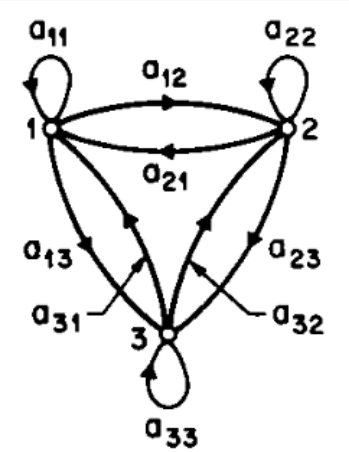
Introduction

HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples



$O = (H, H, T, T, H, T, H, H, T, T, H)$

$S = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)$

sequence Q of states

$Q = (q_1, \dots, q_t, \dots q_T)$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda)$$

for a specific sequence of states

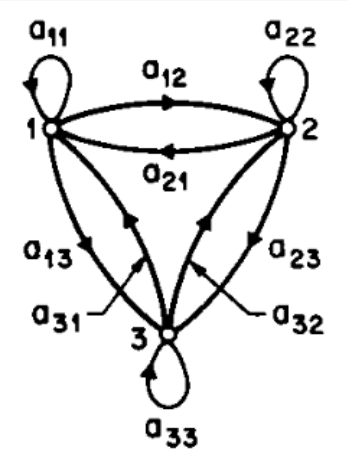
$$P(O|\lambda) = \sum_{\Omega} \boldsymbol{P(O|Q,\lambda)}\, \boldsymbol{P(Q|\lambda)}$$

Ω: over all combinations of $Q, |\Omega| = N^T$

$$\boldsymbol{P(O|Q,\lambda)} = \prod_{t=1}^{T} P(V_m, t|q_t = s_j) = \prod_{t=1}^{T} b_{q_t}(O_t)$$

$$\boldsymbol{P(Q|\lambda)} = \pi_{q_1} \prod_{t=1}^{T-1} P(q_{t+1} = s_j | q_t = s_i) = \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}$$

Hidden Markov Models:



$$O = (H, H, T, T, H, T, H, H, T, T, H)$$

$$S = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)$$

sequence Q of states

$$Q = (q_1, \dots, q_t, \dots q_T)$$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

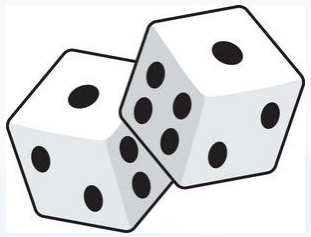
$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda)$$

for a specific sequence of states

$$P(O|\lambda) = \sum_{\Omega} \boldsymbol{P(O|Q,\lambda)\, P(Q|\lambda)}$$

Ω: over all combinations of $Q, |\Omega| = N^T$

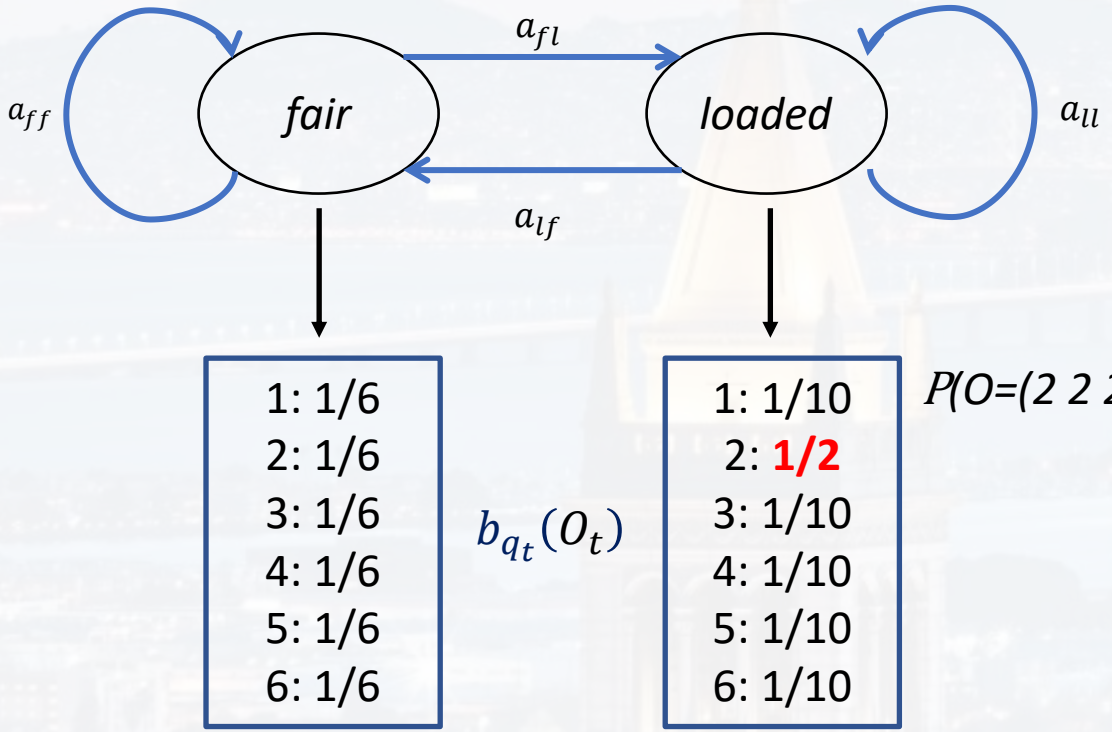
$$P(O|\lambda) = \sum_{\Omega} \left\{ \prod_{t=1}^{T} b_{q_t}(O_t)\, \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \right\}$$



$$P(O|\lambda) = \sum_{\Omega} \boldsymbol{P(O|Q,\lambda)}\, \boldsymbol{P(Q|\lambda)}$$

$$= \sum_{\Omega} \left\{ \prod_{t=1}^{T} b_{q_t}(O_t)\, \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \right\}$$

$\boldsymbol{N}$:	number of states s_n
$\boldsymbol{t}$:	time
$\boldsymbol{q_t}$:	actual state at t
$\boldsymbol{\pi_i}$:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$\boldsymbol{b_j(m)}$:	$P(V_m, t \mid q_t = s_j)$
$\boldsymbol{\lambda}$:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
$\boldsymbol{a_{ij}}$:	$P(q_t = s_j \mid q_{t-1} = s_i)$



Ω: over all combinations of Q, $\Omega = N^T$

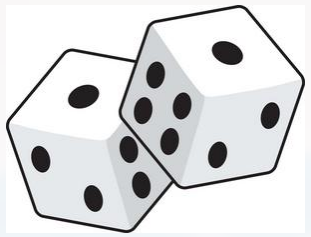
$$A = \begin{pmatrix} a_{ff} & a_{fl} \\ a_{lf} & a_{ll} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.1 \\ 0.25 & 0.75 \end{pmatrix}$$

$\Pi = \{\pi(f), \pi(l)\}$; say, $\Pi = \{0.8, 0.2\}$

$$P(O=(2\ 2\ 2), Q=(f\ f\ f)|\lambda) = \boldsymbol{P(O=(2,2,2)|Q=(fff),\lambda)}\, \boldsymbol{P(Q=(fff)|\lambda)}$$

$$= \boldsymbol{\pi_f}\ \ b_f(2)\ \ a_{ff}\ \ b_f(2)\ \ a_{ff}\ \ b_f(2)$$

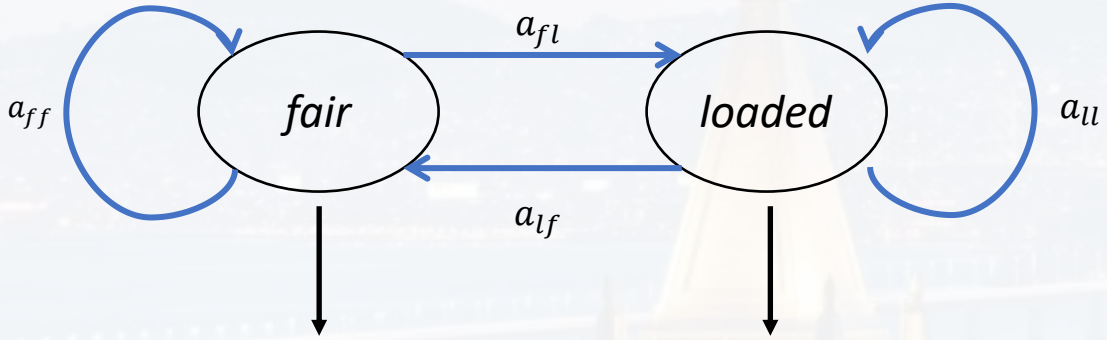
= 0.8 x 1/6 x 0.9 x 1/6 x 0.9 x 1/6 **= 0.003**



$$P(O|\lambda) = \sum_{\Omega} \boldsymbol{P(O|Q,\lambda)}\, \boldsymbol{P(Q|\lambda)}$$

$$= \sum_{\Omega} \left\{ \prod_{t=1}^{T} b_{q_t}(O_t)\, \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \right\}$$

$\boldsymbol{N}$:	number of states $\boldsymbol{s_n}$
$\boldsymbol{t}$:	time
$\boldsymbol{q_t}$:	actual state at $\boldsymbol{t}$
$\boldsymbol{\pi_i}$:	$\boldsymbol{P(q_1 = s_i)}$
$\boldsymbol{V = \{V_1, \dots, V_m, \dots, V_M\}}$:	alphabet
$\boldsymbol{b_j(m)}$:	$\boldsymbol{P(V_m, t|q_t = s_j)}$
$\boldsymbol{\lambda}$:	model
$\boldsymbol{O = (O_1, \dots O_t, \dots O_T)}$:	observations
$\boldsymbol{a_{ij}}$:	$\boldsymbol{P(q_t = s_j | q_{t-1} = s_i)}$



1: 1/6	1: 1/10
2: 1/6	2: **1/2**
3: 1/6	3: 1/10
4: 1/6	4: 1/10
5: 1/6	5: 1/10
6: 1/6	6: 1/10

$b_{q_t}(O_t)$

Ω: over all combinations of Q, $\Omega = N^T$

$$A = \begin{pmatrix} a_{ff} & a_{fl} \\ a_{lf} & a_{ll} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.1 \\ 0.25 & 0.75 \end{pmatrix}$$

$\Pi = \{\pi(f), \pi(l)\}$; say, $\Pi = \{0.8, 0.2\}$

P(O=(2 2 2), Q=(f f f)|λ) = 0.8 x 1/6 x 0.9 x 1/6 x 0.9 x 1/6 **= 0.003**

P(O=(2 2 2), Q=(lll)|λ) = *0.2 x 0.5 x 0.75 x 0.5 x 0.75 x 0.5* ***= 0.014***

$$\frac{P(O=(2\,2\,2), Q=(f\,f\,f)|\lambda)}{P(O=(2\,2\,2), Q=(lll)|\lambda)} = 4.7$$

How do we find the most likely sequence?

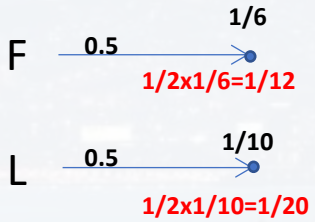
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\}$ $\pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path



observation	4	1	3	5	2	6	6	6
predicted state	?							

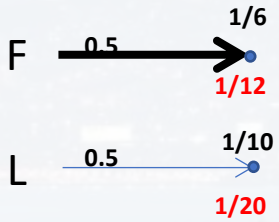
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path



observation	4	1	3	5	2	6	6	6
predicted state	?							

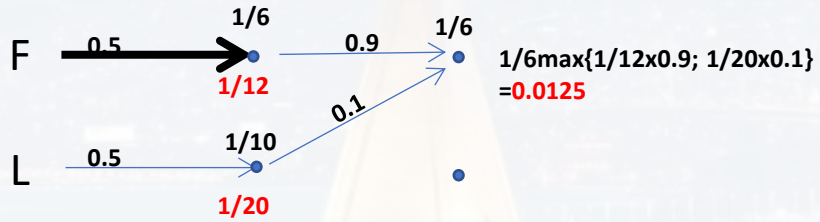
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path



observation	4	1	3	5	2	6	6	6
predicted state	?							

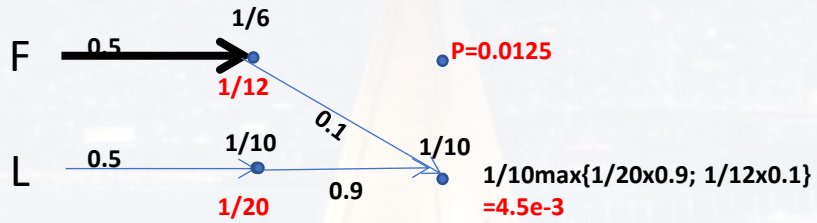
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\}$ $\pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path



observation	4	1	3	5	2	6	6	6
predicted state	?							

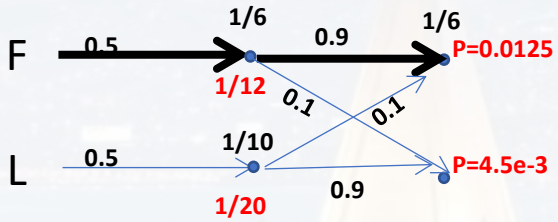
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path



observation	4	1	3	5	2	6	6	6
predicted state	?							

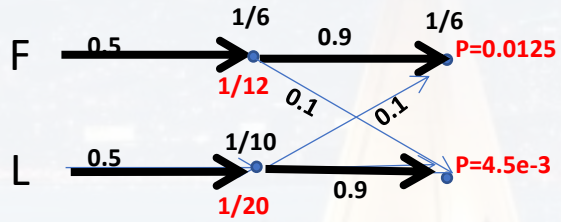
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path ***and*** store the path where each maximum came from



observation	4	1	3	5	2	6	6	6
predicted state	?							

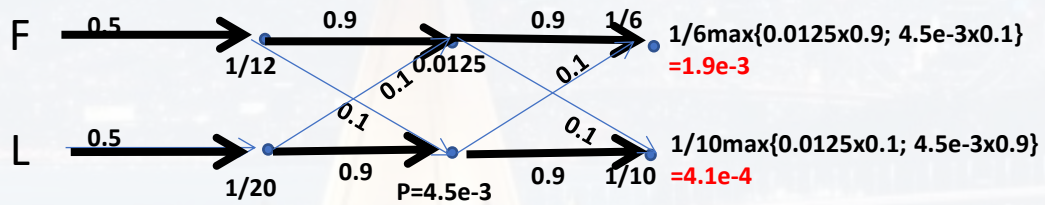
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path ***and*** store the path where each maximum came from



observation	4	1	3	5	2	6	6	6
predicted state	?							

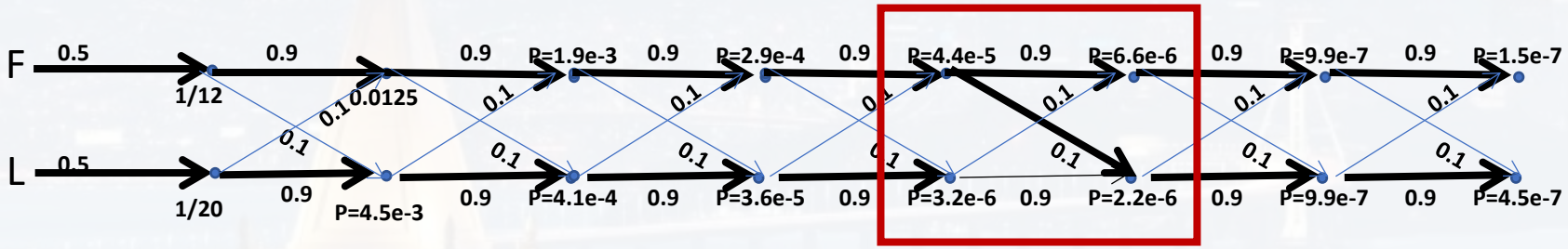
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path ***and*** store the path where each maximum came from



observation	4	1	3	5	2	6	6	6
predicted state	?							

2) backward, we **follow the path** where each maximum came from

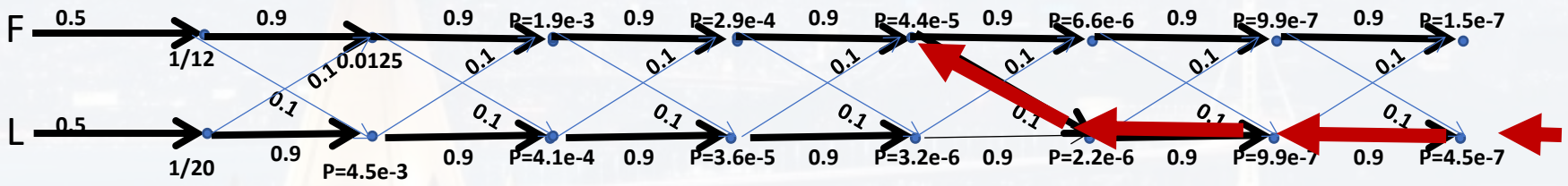
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\} \quad \pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path ***and*** store the path where each maximum came from



observation	4	1	3	5	2	6	6	6
predicted state	?				F	L	L	L

2) backward, we **follow the path** where each maximum came from

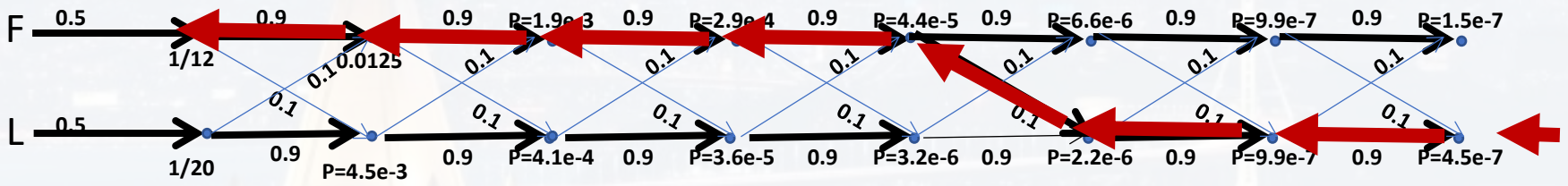
rolling dice with two states (fair = F, loaded = F)

$V = \{1,2,3,4,5,6\}$ $\pi = \{0.5,0.5\}$

$$A = \begin{pmatrix} a_{FF} = 0.9 & a_{LF} = 0.1 \\ a_{FL} = 0.1 & a_{LL} = 0.9 \end{pmatrix}$$

	$b_j(m=1)$	$b_j(m=2)$	$b_j(m=3)$	$b_j(m=2)$	$b_j(m=5)$	$b_j(m=6)$
$S_1 = F$	1/6	1/6	1/6	1/6	1/6	1/6
$S_2 = L$	1/10	1/10	1/10	1/10	1/10	1/2

idea: 1) forward, we follow the most likely path ***and*** store the path where each maximum came from



observation	4	1	3	5	2	6	6	6
predicted state	F	F	F	F	F	L	L	L

2) backward, we **follow the path** where each maximum came from

Hidden Markov Models:

$$P(O|\lambda) = \sum_{\Omega} \left\{ \prod_{t=1}^{T} b_{q_t}(O_t)\, \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \right\}$$

$(2T-1)N^T$	multiplications
$N^T - 1$	additions

Viterbi:

$N(N+1)(T-1)+N$	multiplications
$N(N-1)(T-1)$	additions

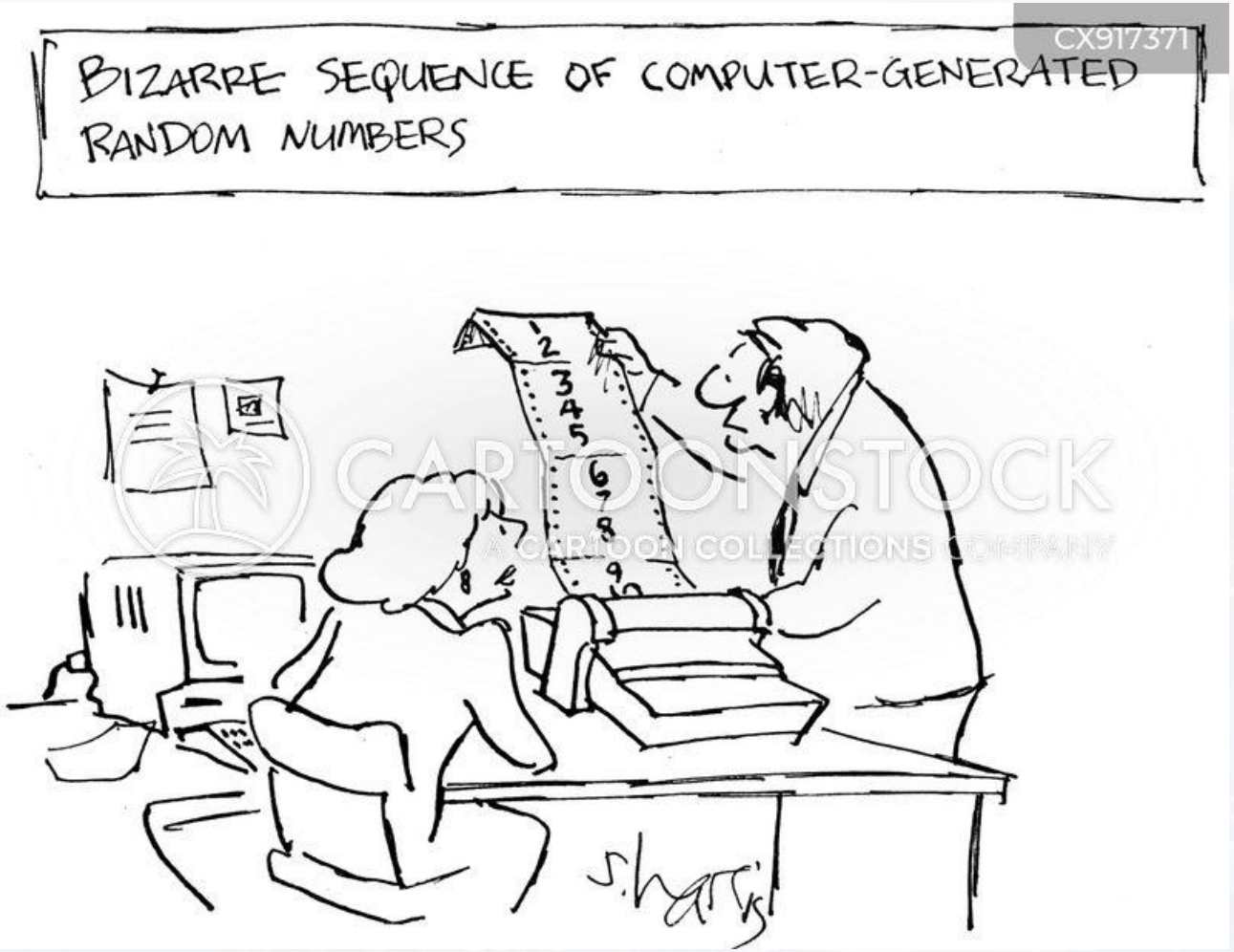
N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

Ω: over all combinations of $Q, |\Omega| = N^T$

note:

- **we solved problem 2)**
- equivalent to **Needleman-Wunsch** (sequence alignment)
- general: forward-backward-algorithm, many extensions/variations (“Lazy” Viterbi, Smith-Waterman, repeated matches, etc)
- solution is **global**

Hidden Markov Models



Outline

Introduction

HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples

Hidden Markov Models:

$$P(O|\lambda) = \sum_{\Omega} \left\{ \prod_{t=1}^{T} b_{q_t}(O_t)\, \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \right\}$$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

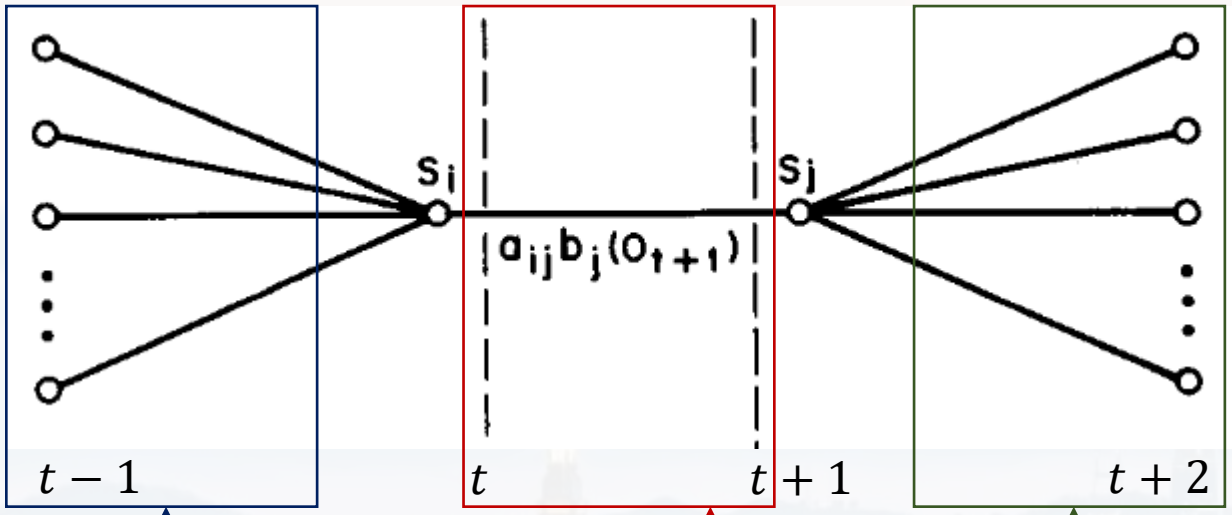
We could start with maximizing $P(O|\bar{\lambda})$ for the estimated model $\bar{\lambda}$ subject to the constrains

$$\sum_{i=1}^{N} \bar{\pi}_i = 1 \qquad \sum_{j=1}^{N} \bar{a}_{ij} = 1 \qquad \sum_{m=1}^{M} \bar{b}_i(m) = 1$$

→ variational calculus

main difference:

GMMs, K—means etc., the actual sequence did not matter
now: sequence (direction imposed by *t*) does matter!



N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

probability $\alpha_t(i)$ of the partial observation sequence *until* time t **and** state s_i at time t

$$\alpha_t(i) = P(O_1, \dots O_t, q_t = s_i|\lambda)$$

probability $\beta_t(i)$ of the partial observation sequence *from* time $t+1$ **given** state s_i at time t

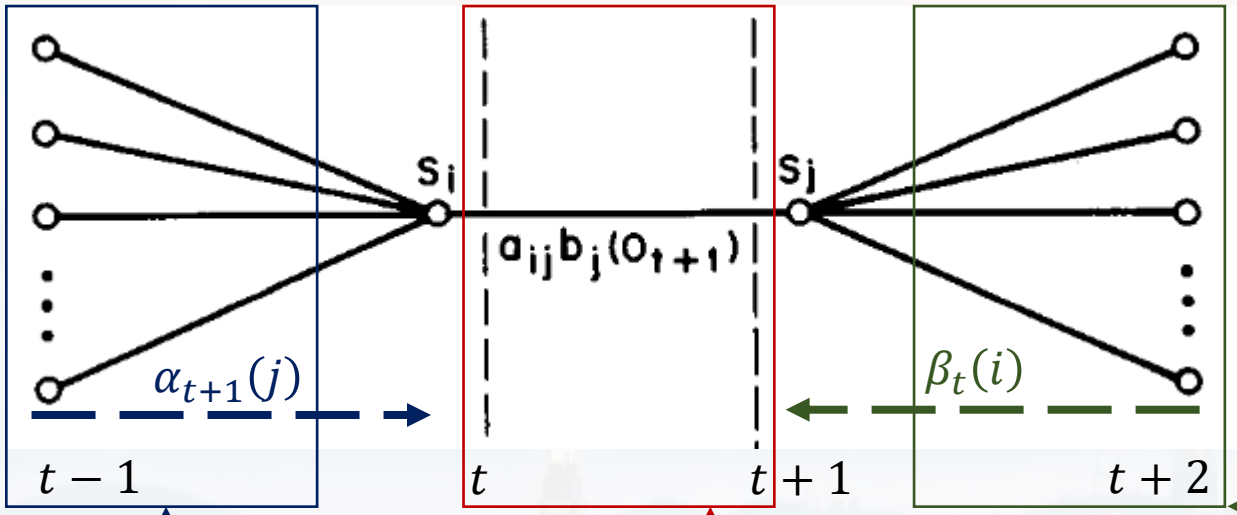
$$\beta_t(i) = P(O_{t+1}, \dots O_T|q_t = s_i, \lambda)$$

marginalization over the previous state sequences for $\alpha_t(i)$ and the subsequent sequences for $\beta_t(i)$

probability $\xi_t(i,j)$ that system was in state s_i at time t and switched to state s_j at $t+1$

$$\xi_t(i,j) = P(q_t = s_i; q_{t+1} = s_j|O, \lambda)$$

Hidden Markov Models:



N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

$$\xi_t(i,j) = P(q_t = s_i; q_{t+1} = s_j \mid O, \lambda)$$

$$\alpha_t(i) = P(O_1, \dots O_t, q_t = s_i \mid \lambda)$$

$$\alpha_1(i) = \pi_i b_i(O_1)$$

$$\alpha_2(j) = b_j(O_2) \sum_{i=1}^{N} \pi_i b_i(O_1)\, a_{ij}$$

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{N} \alpha_t(i)\, a_{ij}$$

note:

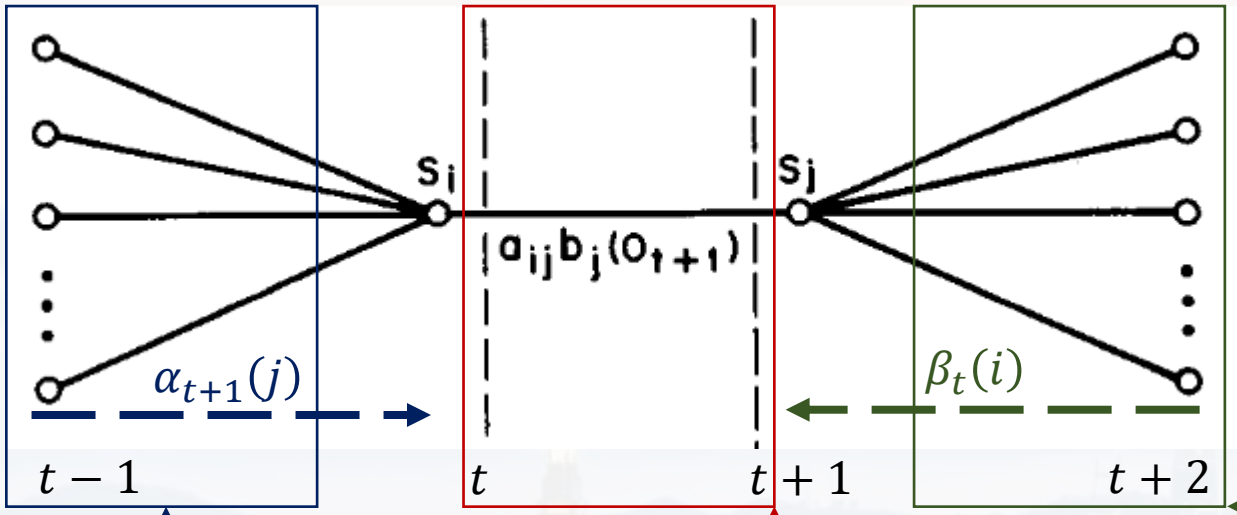
even tough $\beta_t(i)$ goes back in time we need a_{ij}, not a_{ji} because time still elapses $t \rightarrow t+1$

$$\beta_t(i) = P(O_{t+1}, \dots O_T \mid q_t = s_i, \lambda)$$

$$\beta_T(i) = init$$

$$\beta_{T-1}(i) = \sum_{j=1}^{N} \beta_T(j)\, b_j(O_T)\, a_{ij}$$

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



N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \dots, V_m, \dots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \dots O_t, \dots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

$$\alpha_t(i) = P(O_1, \dots O_t, q_t = s_i|\lambda)$$

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{N} \alpha_t(i)\, a_{ij}$$

$$\xi_t(i,j) = P(q_t = s_i; q_{t+1} = s_j | O, \lambda)$$

$$\beta_t(i) = P(O_{t+1}, \dots O_T | q_t = s_i, \lambda)$$

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

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}{P(O|\lambda)} = \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}$$

Hidden Markov Models:

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}{P(O|\lambda)}$$

$$= \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}$$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \ldots, V_m, \ldots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \ldots O_t, \ldots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

probability that the system is in state s_i at time t

$$\bar{\pi}_i = \gamma_1(i)$$

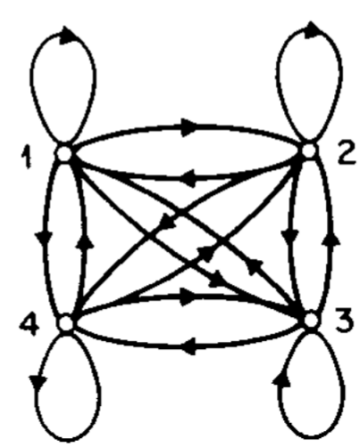
$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} \xi_t(i,j)} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{expected\ numer\ of\ times\ \boldsymbol{s_i} \rightarrow \boldsymbol{s_j}}{expected\ numer\ of\ times\ \boldsymbol{in}\ \boldsymbol{s_i}}$$

$$\bar{b}_j(m) = \frac{\sum_{t=1}^{T} \chi \sum_{i=1}^{N} \xi_t(i,j)}{\sum_{t=1}^{T}\sum_{i=1}^{N} \xi_t(i,j)} = \frac{\sum_{t=1}^{T} \chi\, \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{expected\ numer\ of\ times\ we\ observe\ V_m\ in\ \boldsymbol{s_j}}{expected\ numer\ of\ times\ \boldsymbol{in}\ \boldsymbol{s_j}}$$

$$\chi = \begin{cases} 1 & if\ O_t = V_m \\ 0 & else \end{cases}$$

the **model** $\lambda = (A, B, \Pi) = (\{a_{ij}\}, \{b_j(m)\}, \{\pi_i\})$

Hidden Markov Models:

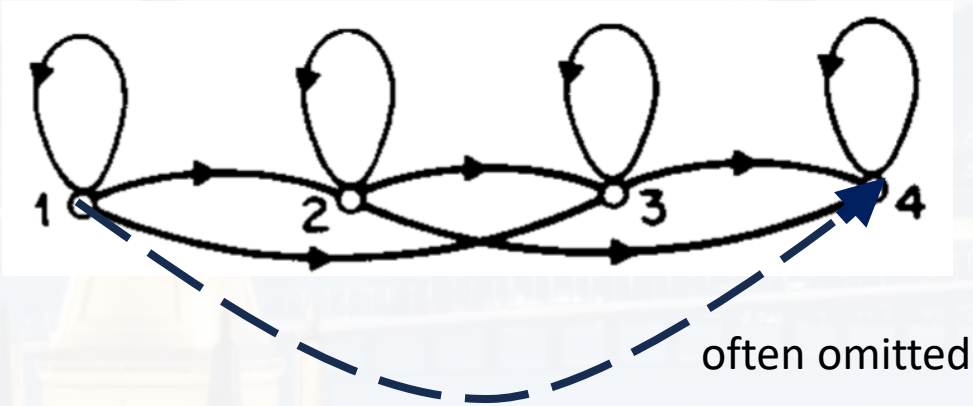


ergodic (aka fully connected)

$$A = \{a_{ij}\} \neq 0 \ \forall \ i, j$$

N:	number of states s_n
t:	time
q_t:	actual state at t
π_i:	$P(q_1 = s_i)$
$V = \{V_1, \ldots, V_m, \ldots, V_M\}$:	alphabet
$b_j(m)$:	$P(V_m, t \mid q_t = s_j)$
λ:	model
$O = (O_1, \ldots O_t, \ldots O_T)$:	observations
a_{ij}:	$P(q_t = s_j \mid q_{t-1} = s_i)$

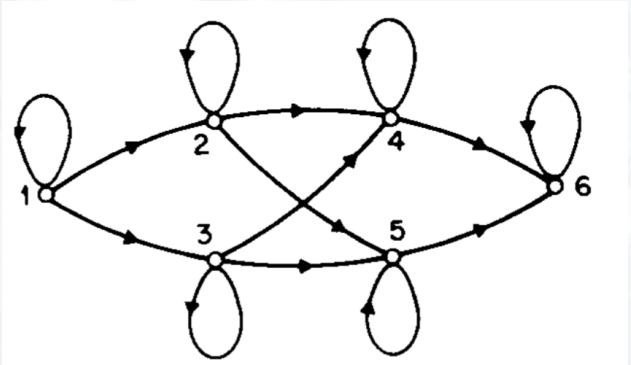
left/right:

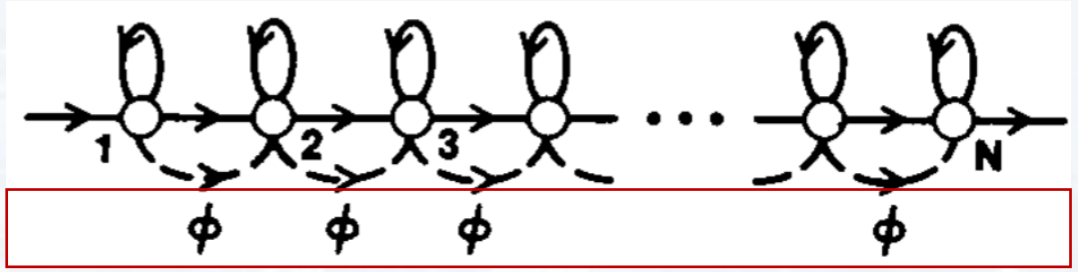


$$A = \begin{pmatrix} a_{ij} \neq 0 & \\ a_{ij} = 0 & \end{pmatrix}$$

$a_{ij} = 0$

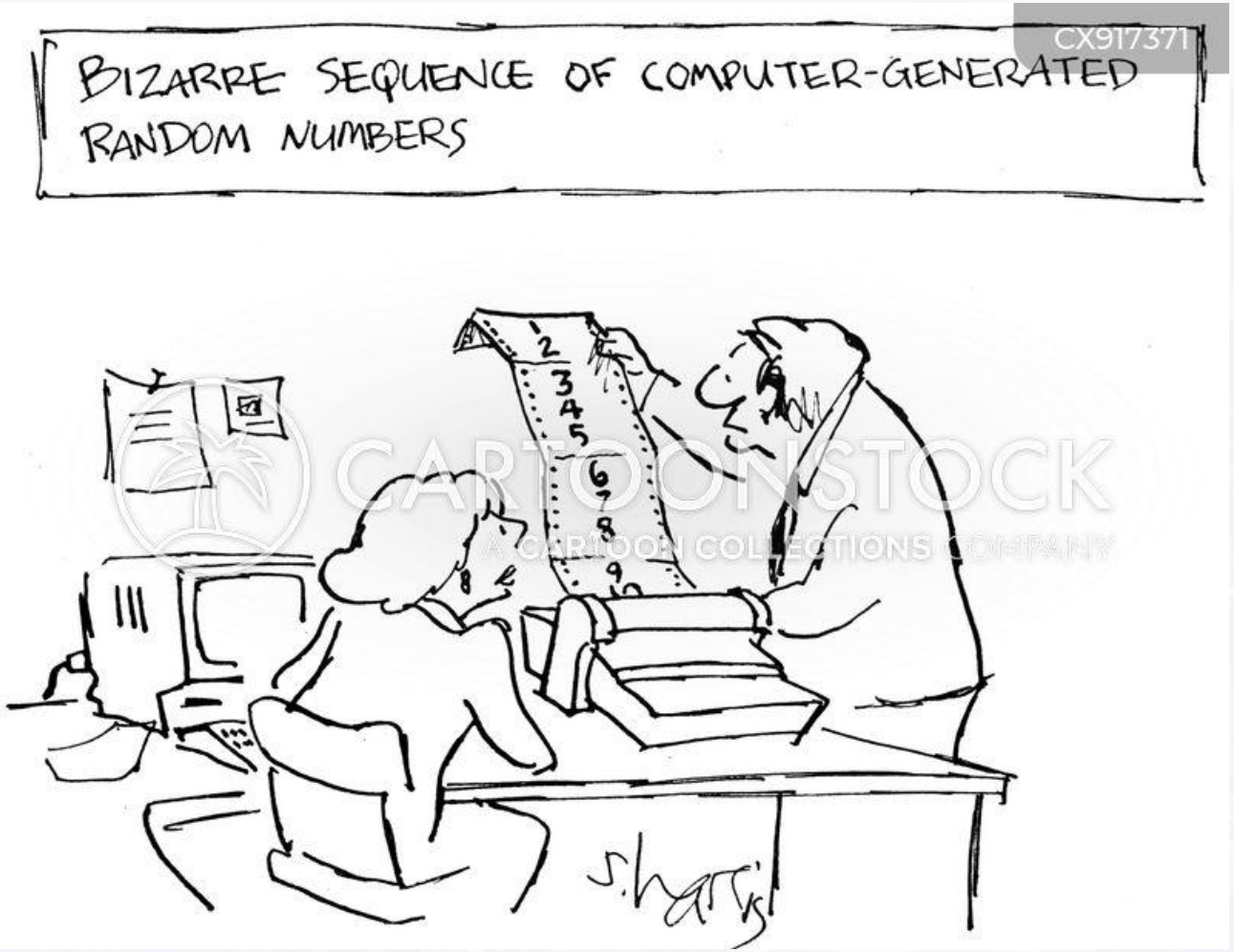
left/right:





silent states ϕ (null output)

Hidden Markov Models



Outline

Introduction

HMMs as Markov Chains

Viterbi-Algorithm

EM (again)

Examples

Hidden Markov Models:

Extracting intracellular diffusive states and transition rates from single-molecule tracking data

Fredrik Persson[1,3], Martin Lindén[2,3], Cecilia Unoson[1] & Johan Elf[1]

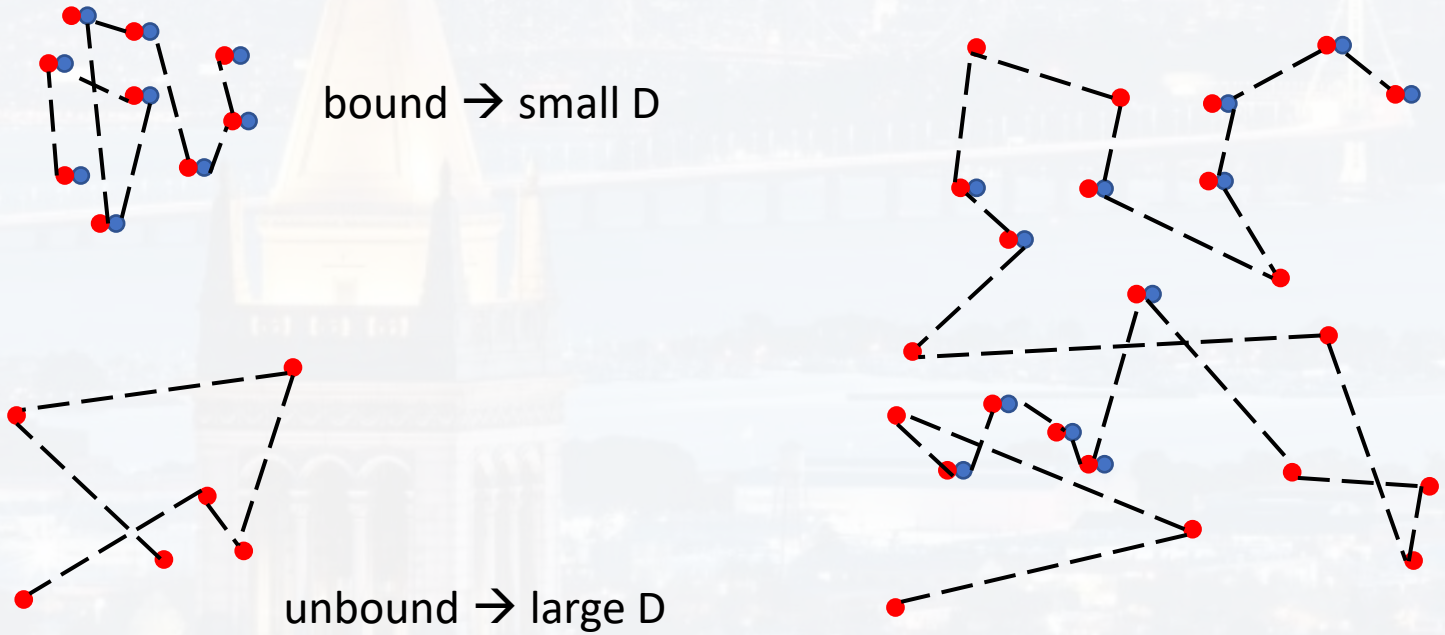
We provide an analytical tool based on a variational Bayesian treatment of hidden Markov models to combine the information from thousands of short single-molecule trajectories of intracellularly diffusing proteins. The method identifies the number of diffusive states and the state transition rates. Using this method we have created an objective interaction map for Hfq, a protein that mediates interactions between small regulatory RNAs and their mRNA targets.

(fewer than ~20 steps) instead of following a few long trajectories with multiple state transitions.

RESULTS

vbSPT for single-molecule tracking data

We provide an analytical tool, variational Bayes SPT (vbSPT), based on a variational Bayesian treatment of a hidden Markov model (HMM)[5,6] that describes diffusing particles making



Extracting intracellular diffusive states and transition rates from single-molecule tracking data

Fredrik Persson[1,3], Martin Lindén[2,3], Cecilia Unoson[1] & Johan Elf[1]

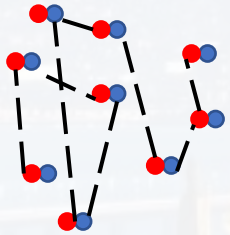
We provide an analytical tool based on a variational Bayesian treatment of hidden Markov models to combine the information from thousands of short single-molecule trajectories of intracellularly diffusing proteins. The method identifies the number of diffusive states and the state transition rates. Using this method we have created an objective interaction map for Hfq, a protein that mediates interactions between small regulatory RNAs and their mRNA targets.

(fewer than ~20 steps) instead of following a few long trajectories with multiple state transitions.

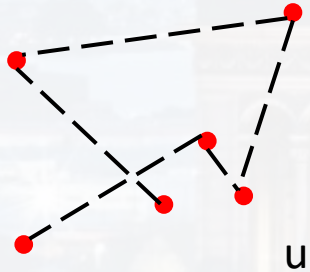
RESULTS

vbSPT for single-molecule tracking data

We provide an analytical tool, variational Bayes SPT (vbSPT), based on a variational Bayesian treatment of a hidden Markov model (HMM)[5,6] that describes diffusing particles making



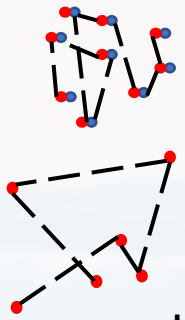
bound → small D



unbound → large D

bound/unbound?

Hidden Markov Models:

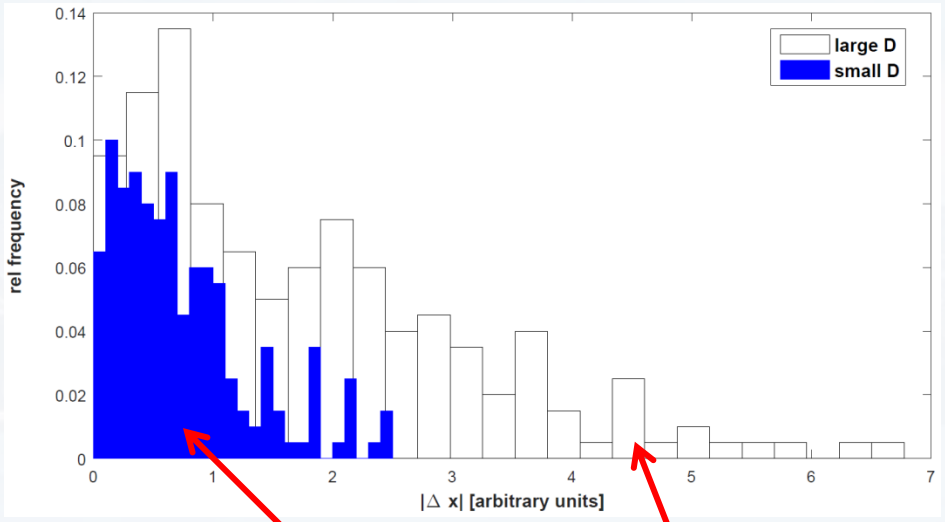


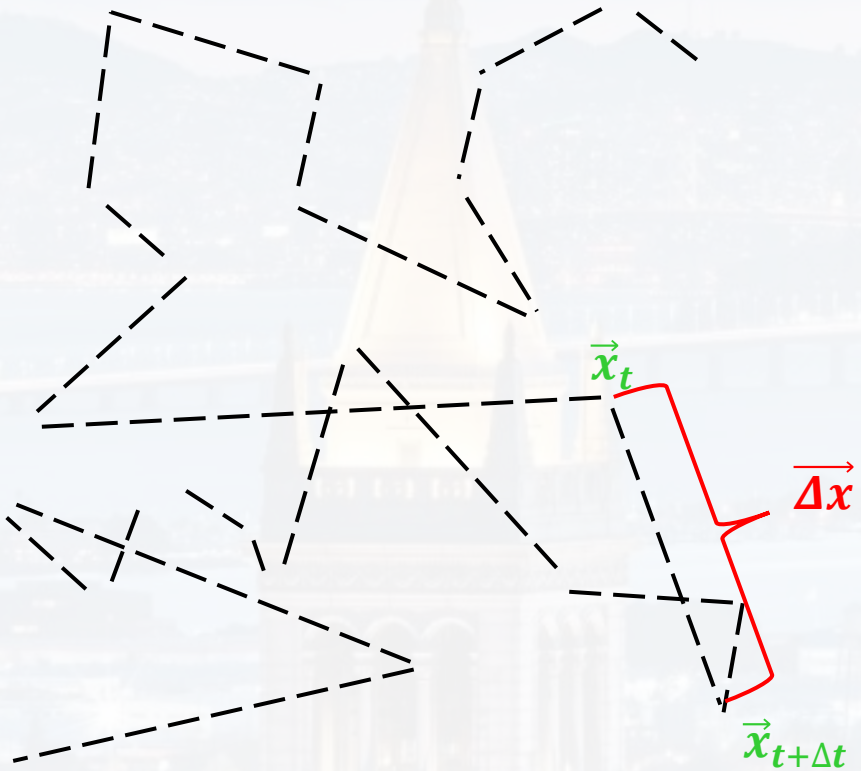
bound → small D

unbound → large D

two states: S={bound, unbound}

$S=\{D_1, D_2\}$

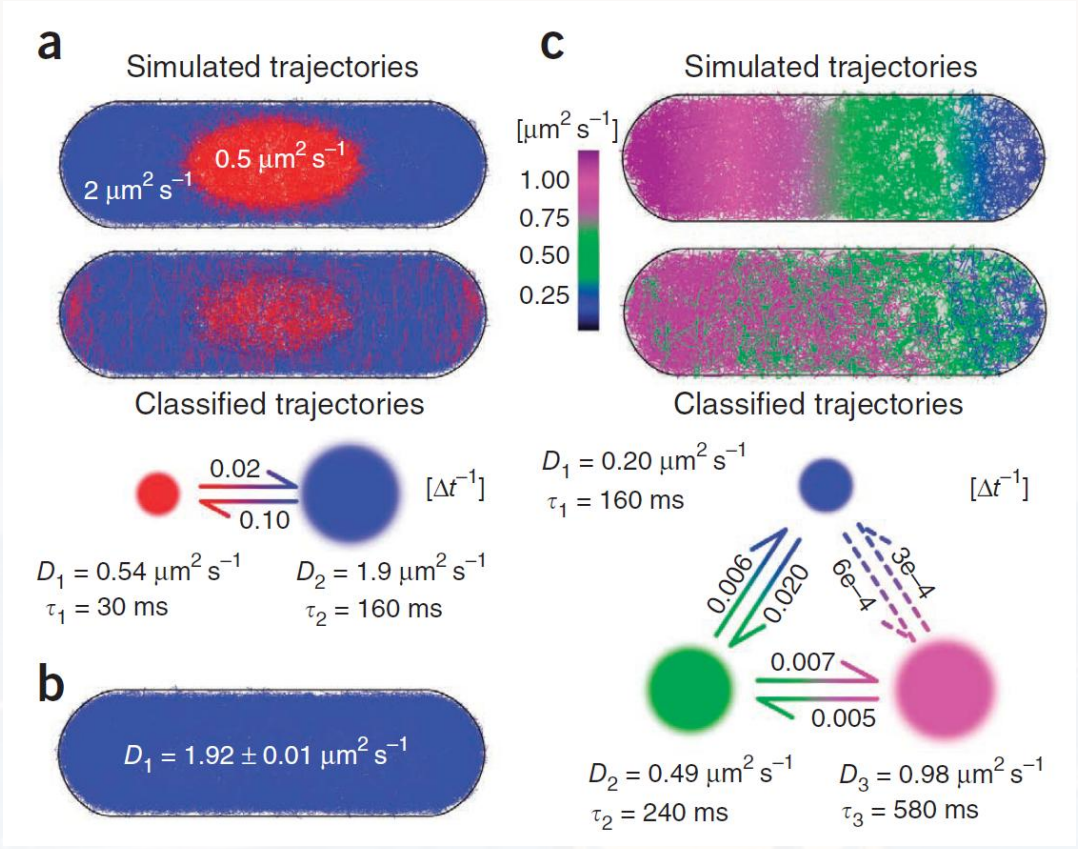




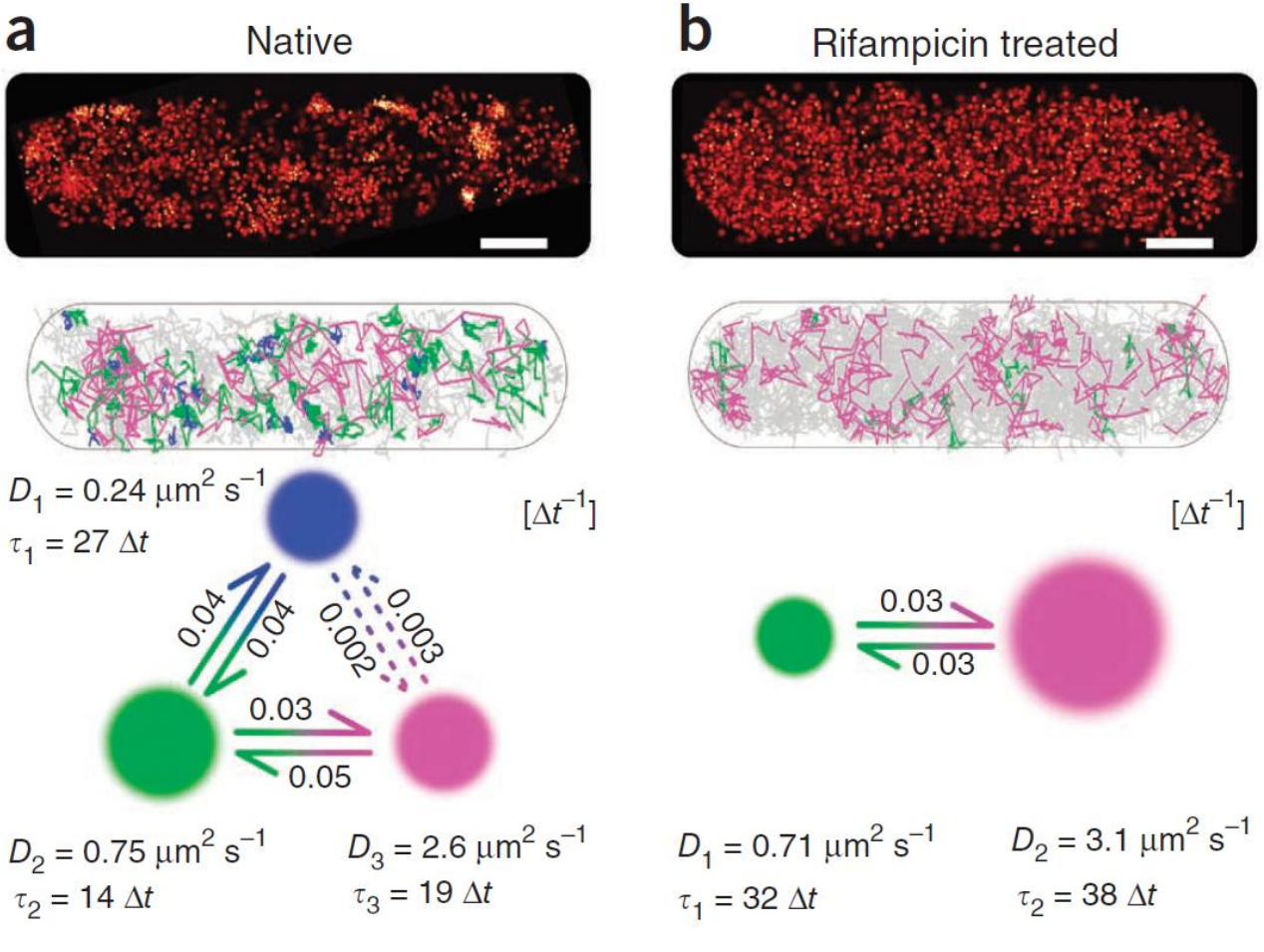
bound/unbound?

emission probabilities $P(\Delta x) \sim \exp\left(-\frac{\Delta x^2}{4\pi D_s \Delta t}\right)$

for letters $V = \{\Delta x\}$



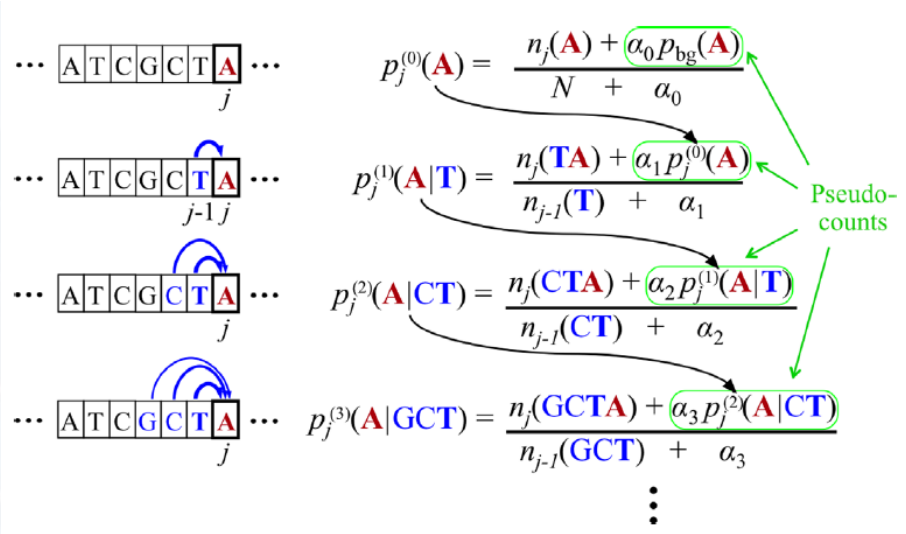
Persson et al., 2013, nature



Bayesian Markov models consistently outperform PWMs at predicting motifs in nucleotide sequences

Matthias Siebert[1,2] **and Johannes Söding**[1,*]

[1]Quantitative and Computational Biology, Max Planck Institute for Biophysical Chemistry, Am Fassberg 11, 37077 Göttingen, Germany and [2]Gene Center, Ludwig-Maximilians-Universität München, Feodor-Lynen-Strasse 25, 81377 Munich, Germany



$$p_j^{(0)}(\mathbf{A}) = \frac{n_j(\mathbf{A}) + \alpha_0 p_{\mathrm{bg}}(\mathbf{A})}{N + \alpha_0}$$

$$p_j^{(1)}(\mathbf{A}|\mathbf{T}) = \frac{n_j(\mathbf{TA}) + \alpha_1 p_j^{(0)}(\mathbf{A})}{n_{j-1}(\mathbf{T}) + \alpha_1}$$

$$p_j^{(2)}(\mathbf{A}|\mathbf{CT}) = \frac{n_j(\mathbf{CTA}) + \alpha_2 p_j^{(1)}(\mathbf{A}|\mathbf{T})}{n_{j-1}(\mathbf{CT}) + \alpha_2}$$

$$p_j^{(3)}(\mathbf{A}|\mathbf{GCT}) = \frac{n_j(\mathbf{GCTA}) + \alpha_3 p_j^{(2)}(\mathbf{A}|\mathbf{CT})}{n_{j-1}(\mathbf{GCT}) + \alpha_3}$$

n-th order HMM instead of PWM (**P**osition **W**eight **M**atrices)

best algorithm before (vision) transformers were available!

<u>motif finding:</u>

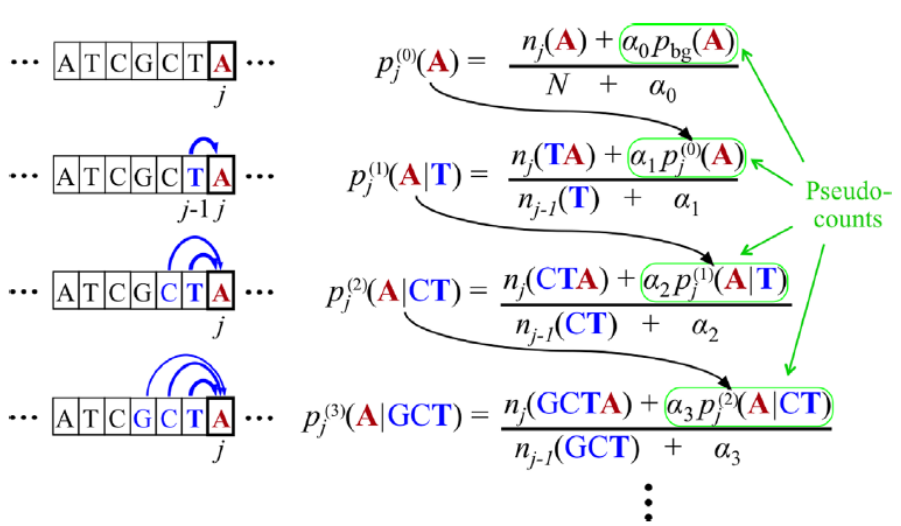
expression values → sequence ***O***
expression levels → alphabet ***V***

A, B, Π → has to be fitted → sequence of states ***(motifs)***

Bayesian Markov models consisten PWMs at predicting motifs in nucle

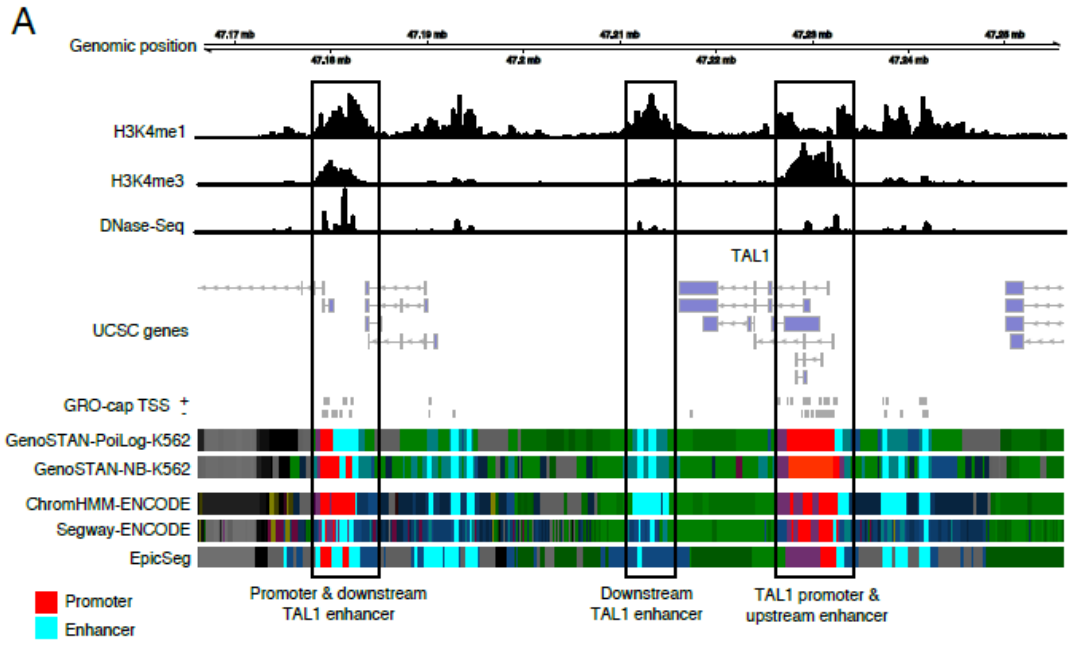
Matthias Siebert[1,2] **and Johannes Söding**[1,*]

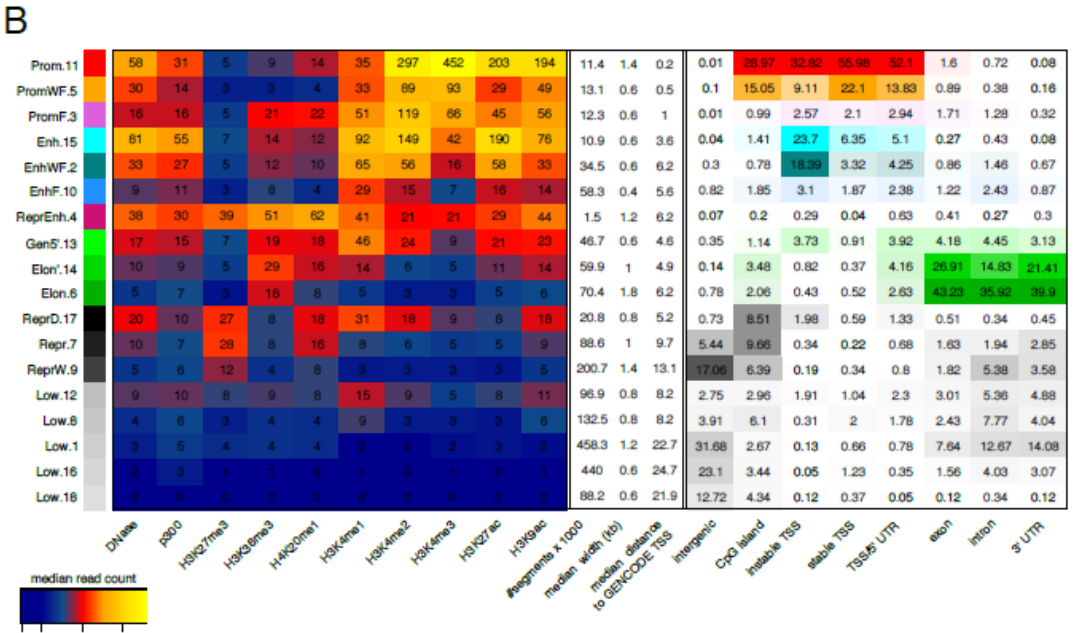
[1]Quantitative and Computational Biology, Max Planck Institute for Biophysic Göttingen, Germany and [2]Gene Center, Ludwig-Maximilians-Universität Mü Munich, Germany



n

P





motif finding:

expression values → sequence ***O***
expression levels → alphabet ***V***

A, B, Π → has to be fitted → sequence of state

Hidden Markov Models:


```
import numpy as np
import matplotlib.pyplot as plt
```

HMM tools are not available in `sklearn` anymore → `hmmlearn`

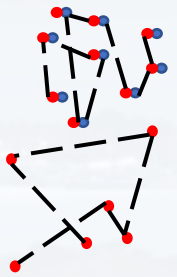
```
#pip install --upgrade --user hmmlearn
from hmmlearn import hmm
```

hmm.

- BaseHMM
- COVARIANCE_TYPES
- CategoricalHMM
- GMMHMM
- GaussianHMM
- MultinomialHMM
- PoissonHMM

die example

the diffusion example

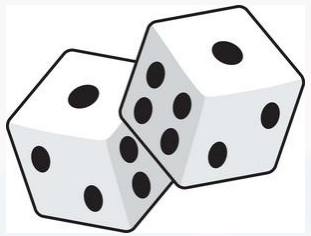


bound → small D

unbound → large D

$$P(\Delta x) \sim \exp\left(-\frac{\Delta x^2}{4\pi D_s \Delta t}\right)$$

Hidden Markov Models:



fair vs loaded

```
model                = hmm.CategoricalHMM(n_components = 2)

model.startprob_     = np.array([0.5, 0.5])

model.transmat_      = np.array([[0.90, 0.10],[0.05, 0.95]])

model.emissionprob_  = np.array(\
                       [[1/6, 1/6, 1/6, 1/6, 1/6, 1/6],\
                        [1/10, 1/10, 1/10, 1/10, 1/10, 1/2]])
```

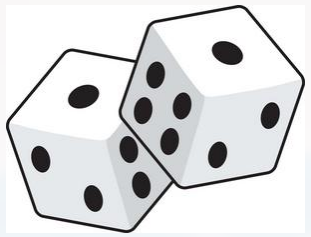
$\pi = \{\pi(f), \pi(l)\} = \{0.5, 0.5\}$

$$A = \begin{pmatrix} a_{ff} & a_{fl} \\ a_{lf} & a_{ll} \end{pmatrix} = \begin{pmatrix} 0.90 & 0.10 \\ 0.05 & 0.95 \end{pmatrix}$$

b_f = {1/6, 1/6, 1/6, 1/6, 1/6, 1/6}

b_l = {1/10, 1/10, 1/10, 1/10, 1/10, **1/2**}

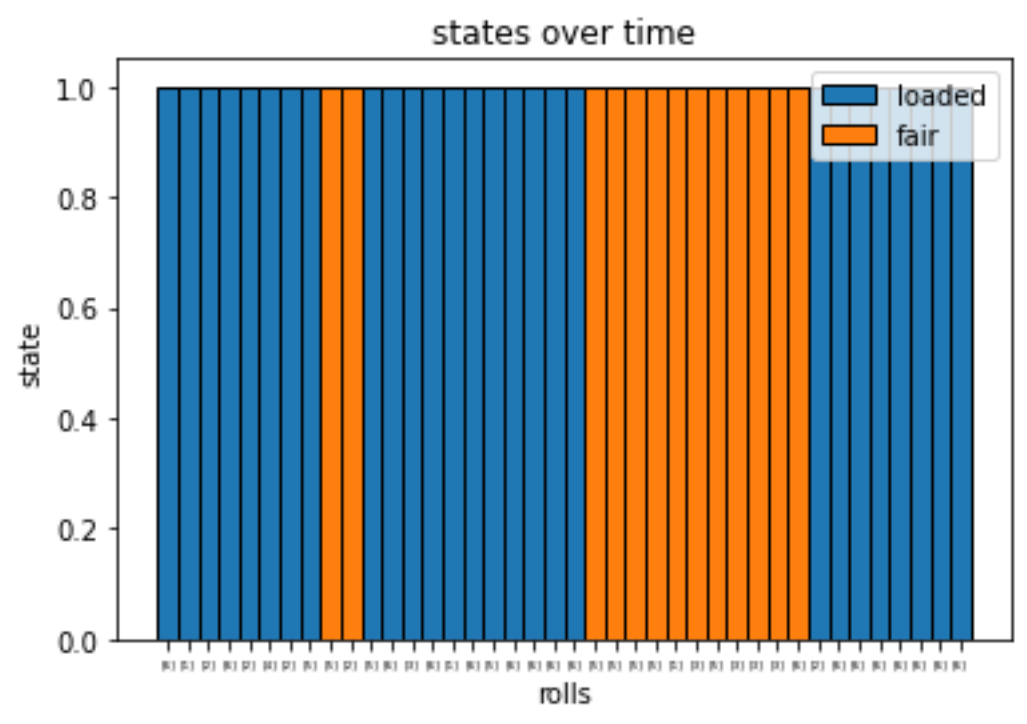
Hidden Markov Models:



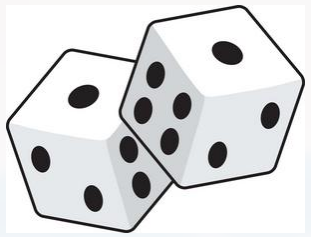
```
model                   = hmm.CategoricalHMM(n_components = 2)
model.startprob_        = np.array([0.5, 0.5])
model.transmat_         = np.array([[0.90, 0.10],[0.05, 0.95]])
model.emissionprob_     = np.array([[1/6,  1/6,  1/6,  1/6,  1/6,  1/6],\
                                    [1/10, 1/10, 1/10, 1/10, 1/10, 1/2]])
```

```
N = 40
[X, S] = model.sample(N)
```

generating the observations X
based on the model. S are the true states.

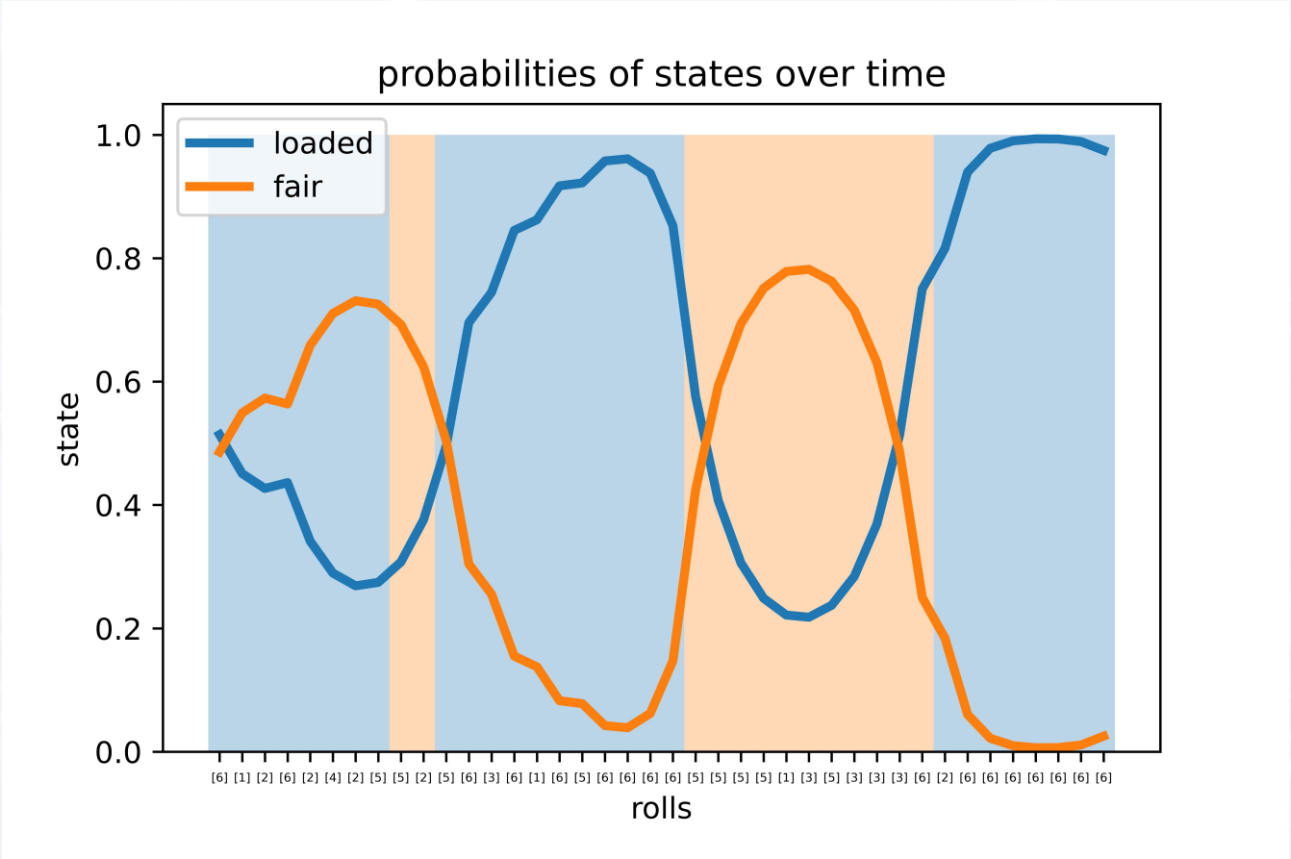


Hidden Markov Models:



```
[X, S] = model.sample(N)

Probs  = model.predict_proba(X)
```



Hidden Markov Models

Thank you very much for your attention!