Connecting Neural and Probabilistic Models of Sequence Generation: An Example from Birdsong Studies

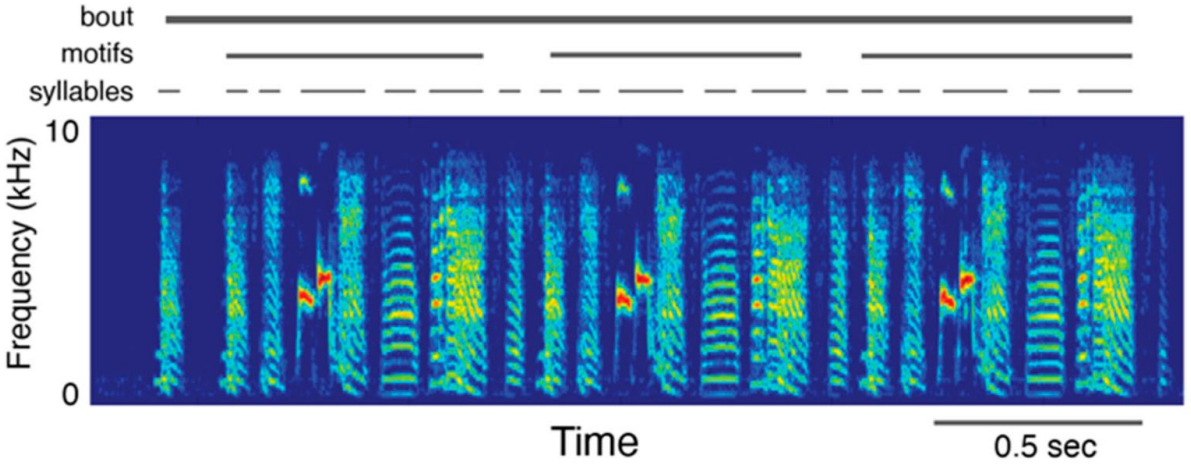
Sumithra Surendralal
Symbiosis School for Liberal Arts, Pune

Brains, Dynamics, and Computation 2025

Timing is a critical element of many behaviours

Eadweard Muybridge
Animal Locomotion, Plate 626

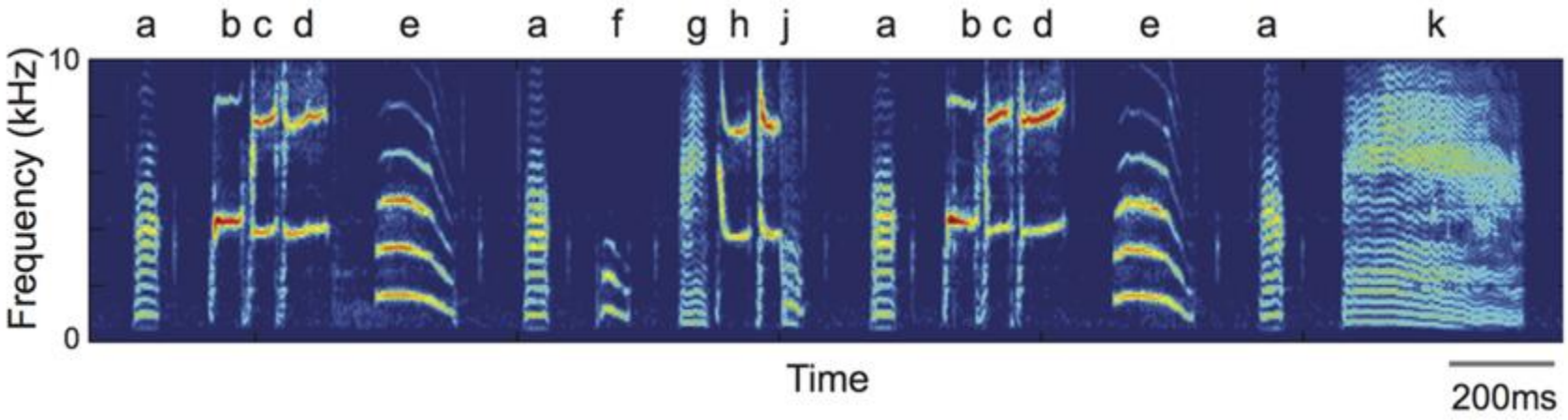
Birdsong is an example of behaviour that relies on precise timing



Zebra finch

CREDIT
Long Lab/NYU School of Medicine

Birdsong is an example of behaviour that relies on precise timing



Bengalese finch

How are temporally precise sequences generated neurally?

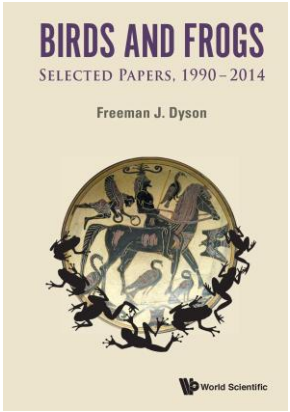
Tutorial 1: A neural model of birdsong sequence production

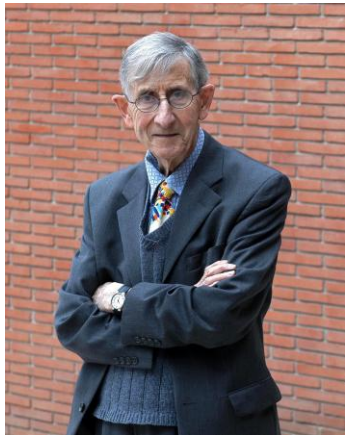
What is the syntax underlying these sequences?

Tutorial 2: Hidden Markov Models for sequence modelling

How do models for the two inform each other?

Tutorial 3: Evaluating probabilistic models of birdsong sequence production





Big picture

https://images.app.goo.gl/PFXWEw8GUR89ce4m7

Familiarity with details

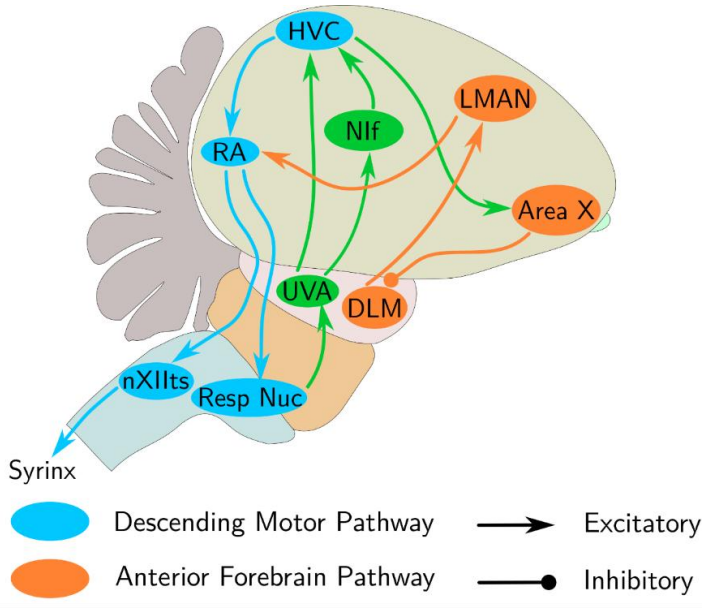
The perfect balance of both

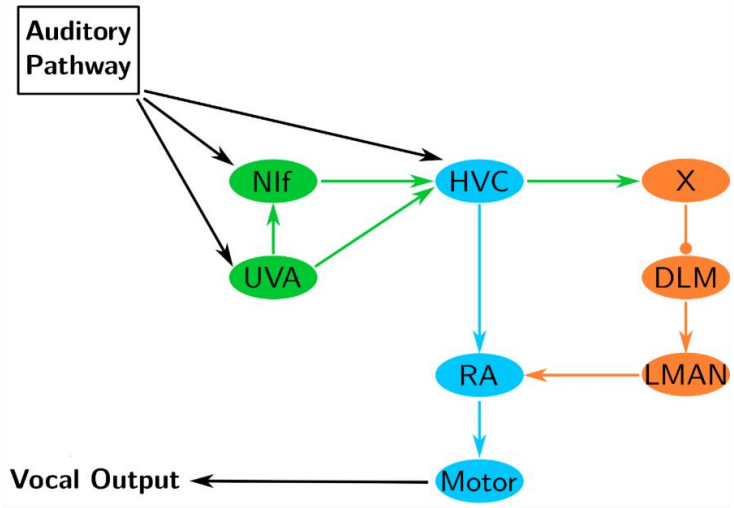
https://images.app.goo.gl/5P8SyaCE18SvZKJ6A

Tutorial 1

A neural model of birdsong sequence production

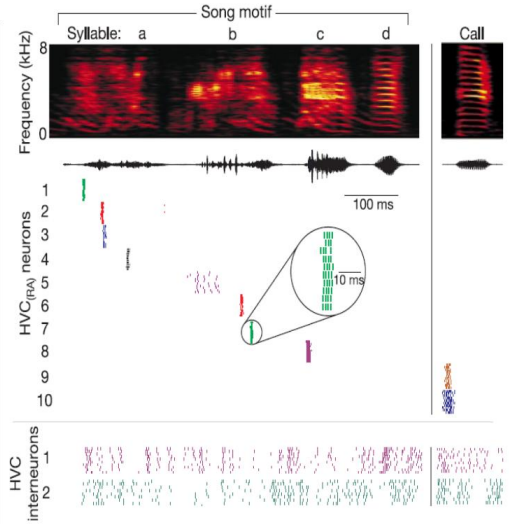
Songbirds have specialized neural circuitry for the learning and production of song



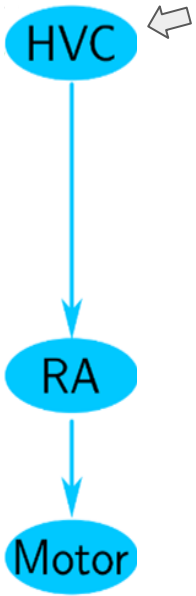


Nottebohm, Stokes, & Leonard, 1976

Experimental observation – Sparse, precisely timed spiking in HVC during singing

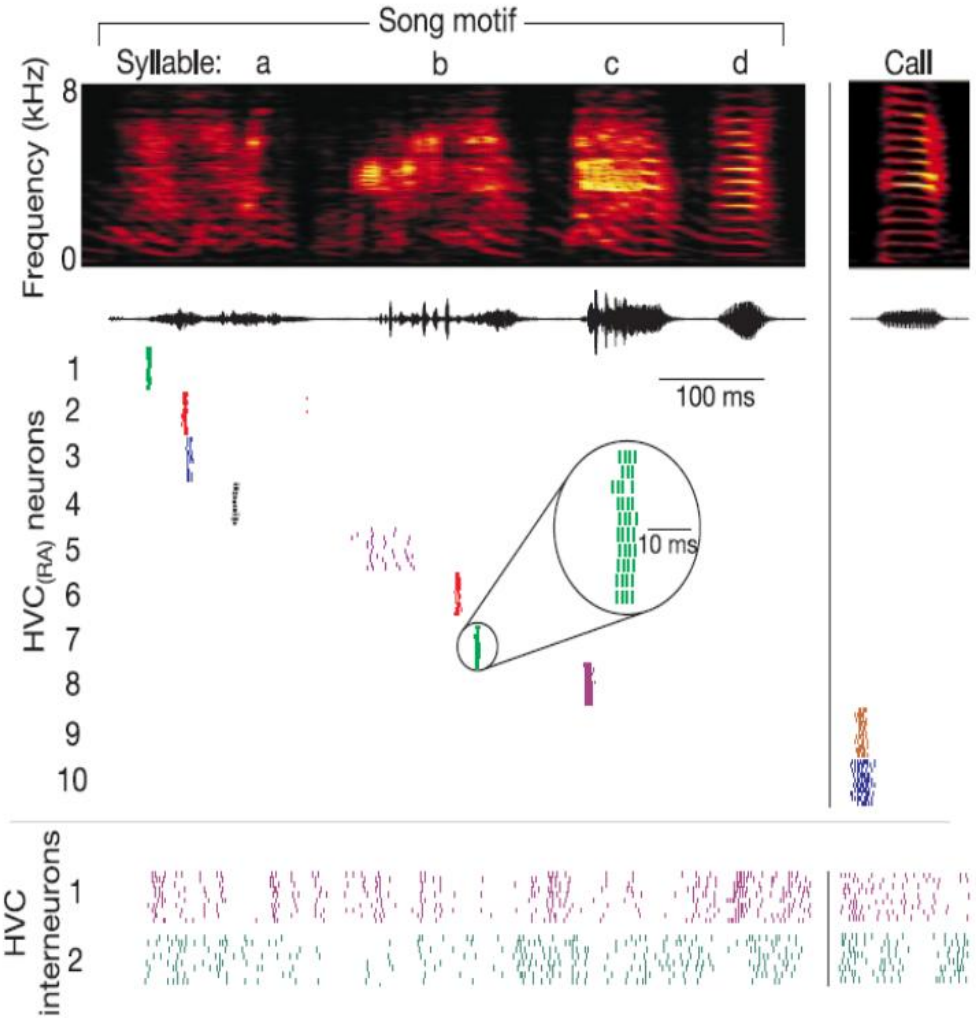


Zebra finch



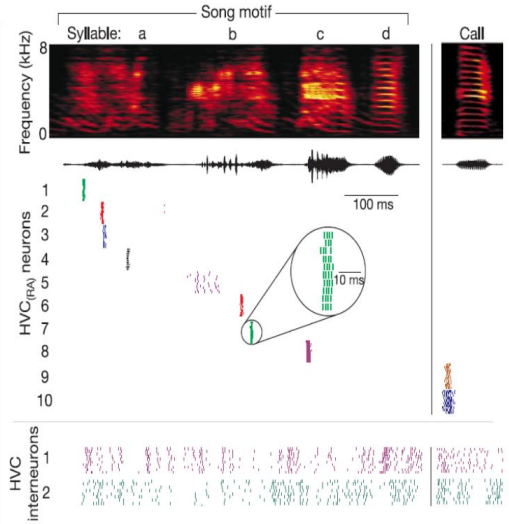
Hahnloser, Kozhevnikov, Fee (2002)

Visualizing spike timing across multiple neurons over time

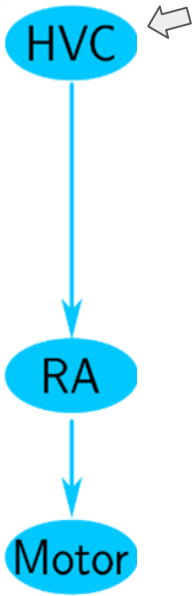


This is a raster plot

Experimental observation – Sparse, precisely timed spiking in HVC during singing

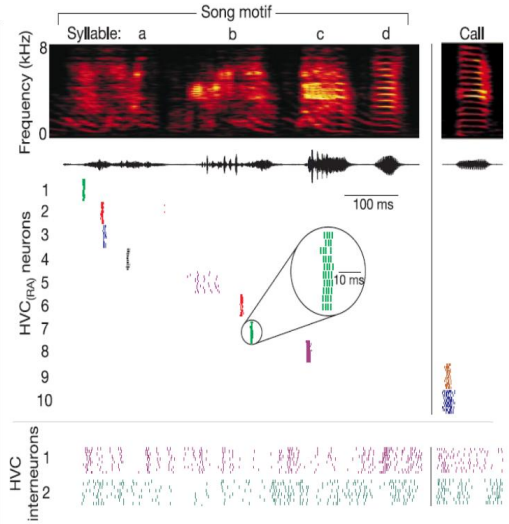


Zebra finch

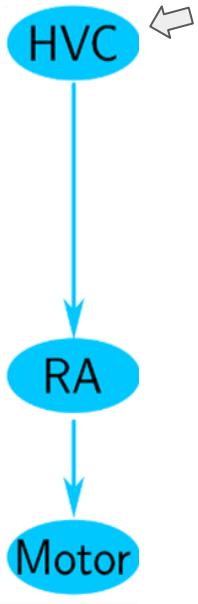


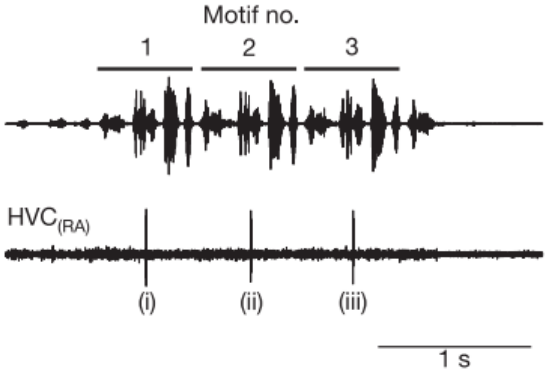
Hahnloser, Kozhevnikov, Fee (2002)

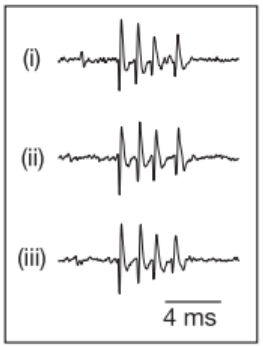
Experimental observation – Sparse, precisely timed spiking in HVC during singing



Zebra finch

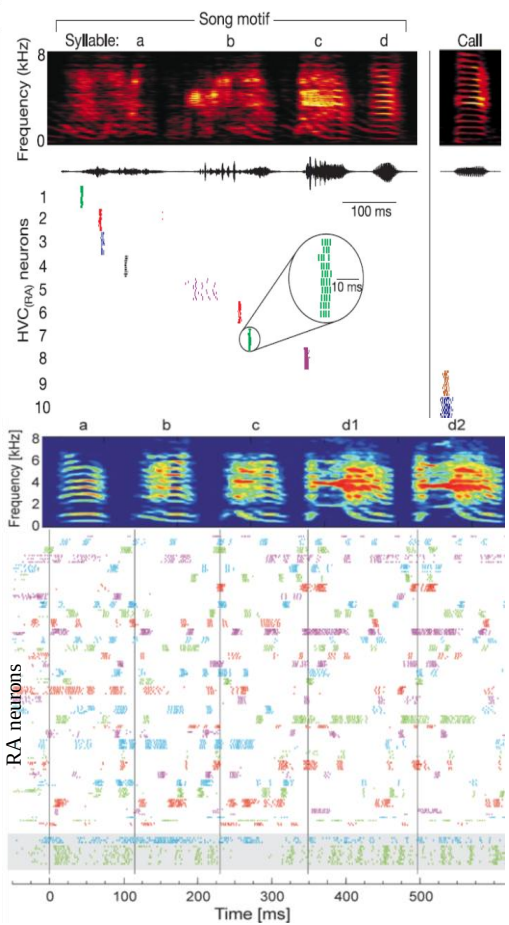




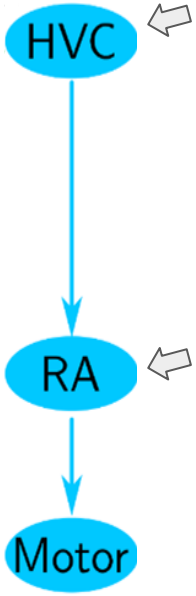


Hahnloser, Kozhevnikov, Fee (2002)

Experimental observation – Different neural firing patterns in HVC and RA during singing

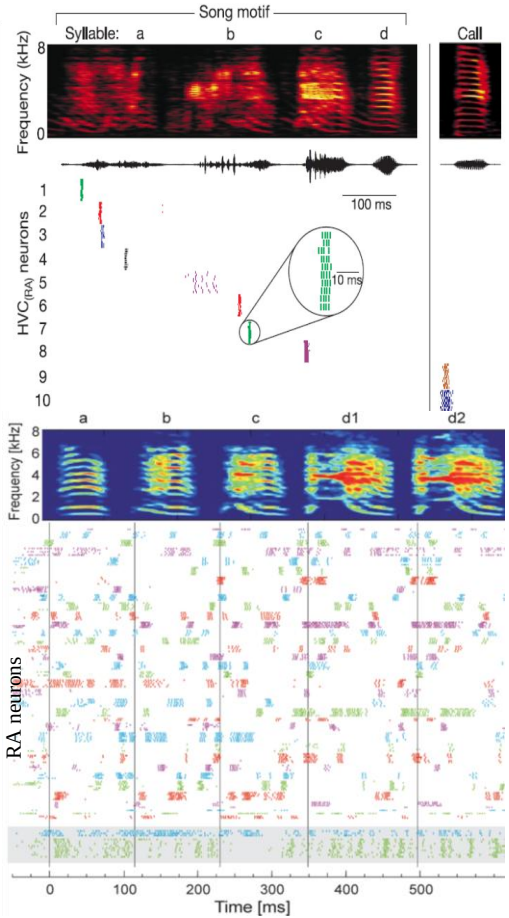


Zebra finch



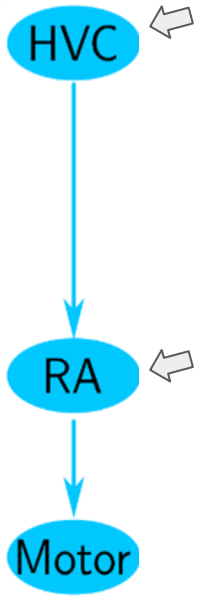
Hahnloser, Kozhevnikov, Fee (2002), Leonardo & Fee (2005)

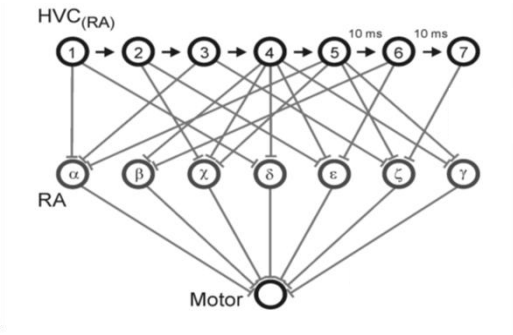
Experimental observation – Different neural firing patterns in HVC and RA during singing



Hahnloser, Kozhevnikov, Fee (2002), Leonardo & Fee (2005)

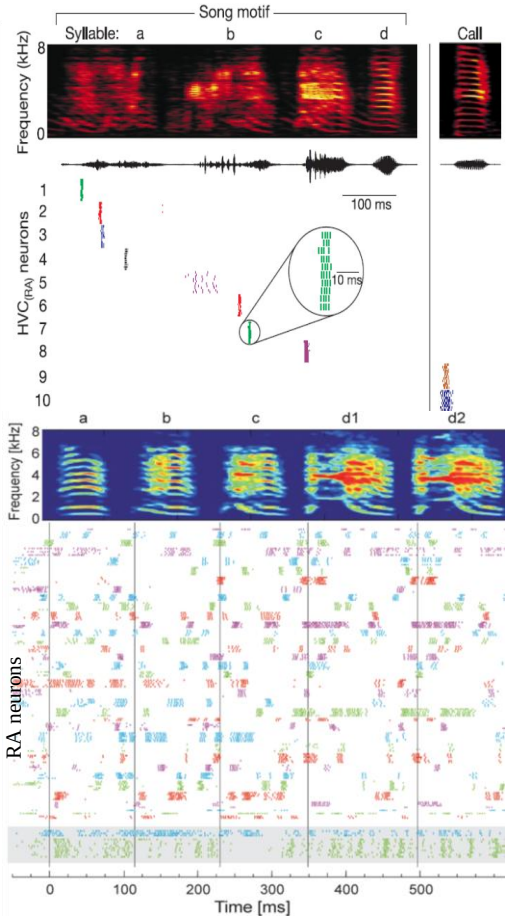
Zebra finch





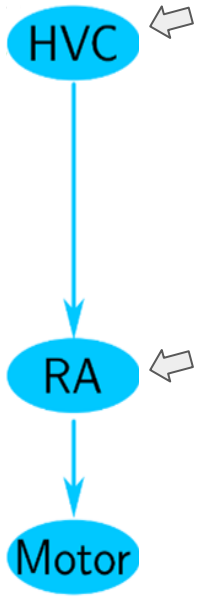
Leonardo & Fee (2005)

Experimental observation – Different neural firing patterns in HVC and RA during singing

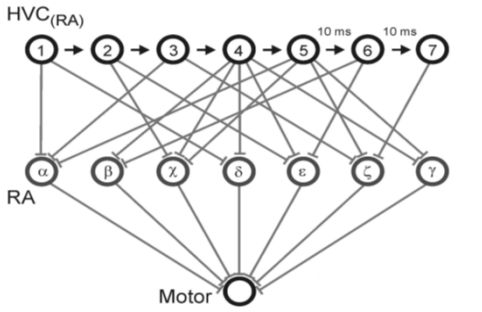


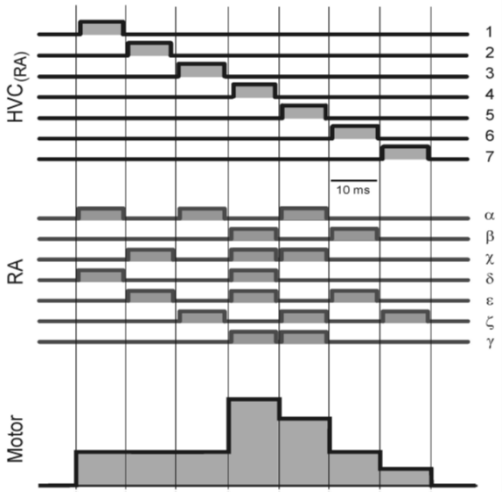
Hahnloser, Kozhevnikov, Fee (2002),

Zebra finch



Leonardo & Fee (2005)

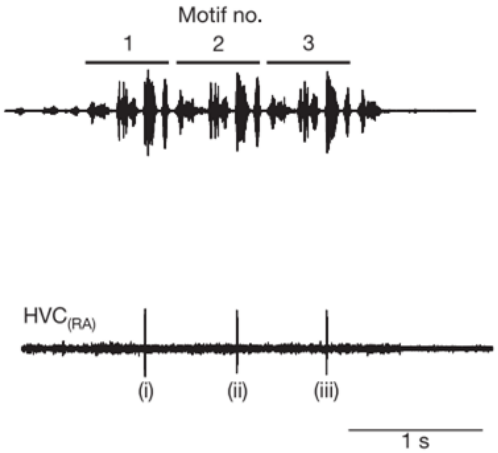




Leonardo & Fee (2005)

What kind of neural architecture can support the observed behaviour and neural recordings?

Individual syllables and motifs should exhibit temporal stability—once activated, they persist over a specific time window



Previously learned syllables and sequences should be faithfully reproduced retaining their acoustic and temporal structure

A bird should be capable of storing and reproducing a large repertoire of distinct syllables and their combinations

Birds should be able to acquire new syllables or novel sequences of syllables through auditory experience and practice.

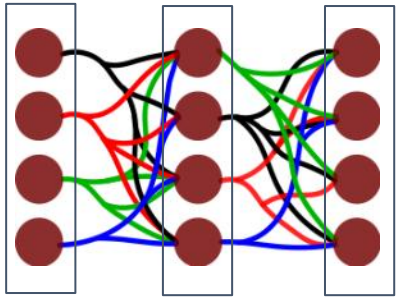
Propagate precise activity

Ensure that neurons fire once and only once during a motif

Do so reliably, sequentially, and robustly

Synfire Chains

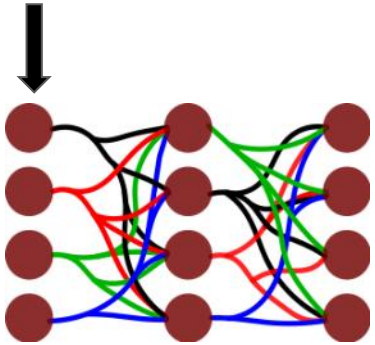
A synfire chain is a feed-forward network of neurons with many layers (or pools) which are sequentially ordered to form a chain



(Abeles, 1982)

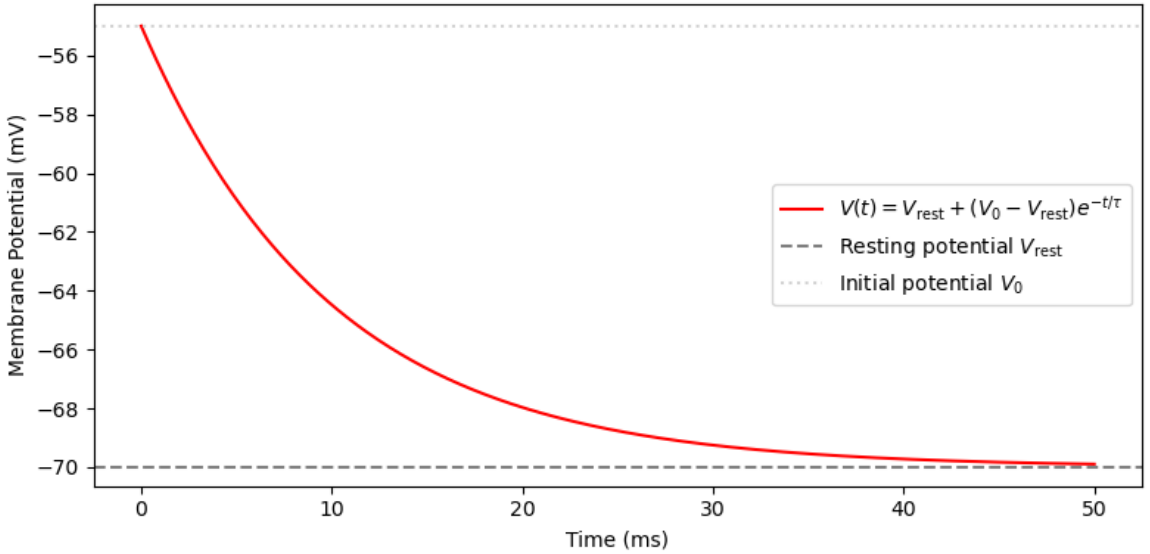
- All neurons in a layer make excitatory connections to all neurons in the next layer.
- Activating the neurons in the first layer sets off a chain reaction where each layer activates the subsequent layer.
- This leads to a signal of neural activity propagating down the chain

Leaky Integrate-and-Fire (LIF) Neuron

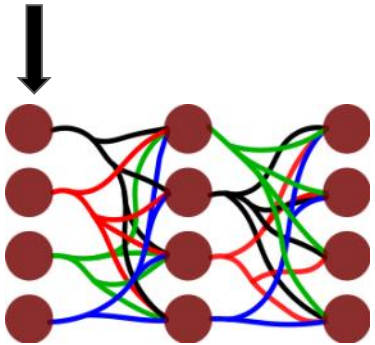


Resting membrane potential V_{rest}

$$\tau \frac{\mathrm{d}V(t)}{\mathrm{d}t} = V_0 - V(t)$$



Leaky Integrate-and-Fire (LIF) Neuron



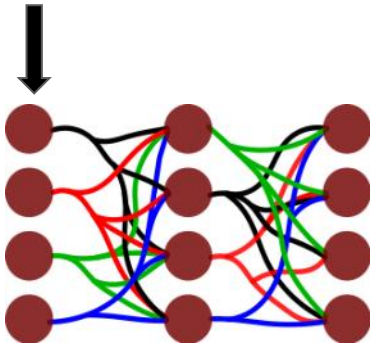
Resting membrane potential V_{rest}

$$\tau \frac{\mathrm{d}V}{\mathrm{d}t} = V_0 - V(t) + RI(t)$$

Spiking condition

$V(t) \geq V_{\text{thresh}}$ a spike is emitted, and the voltage is reset to V_{rest}

Leaky Integrate-and-Fire (LIF) Neuron



Resting membrane potential V_{rest}

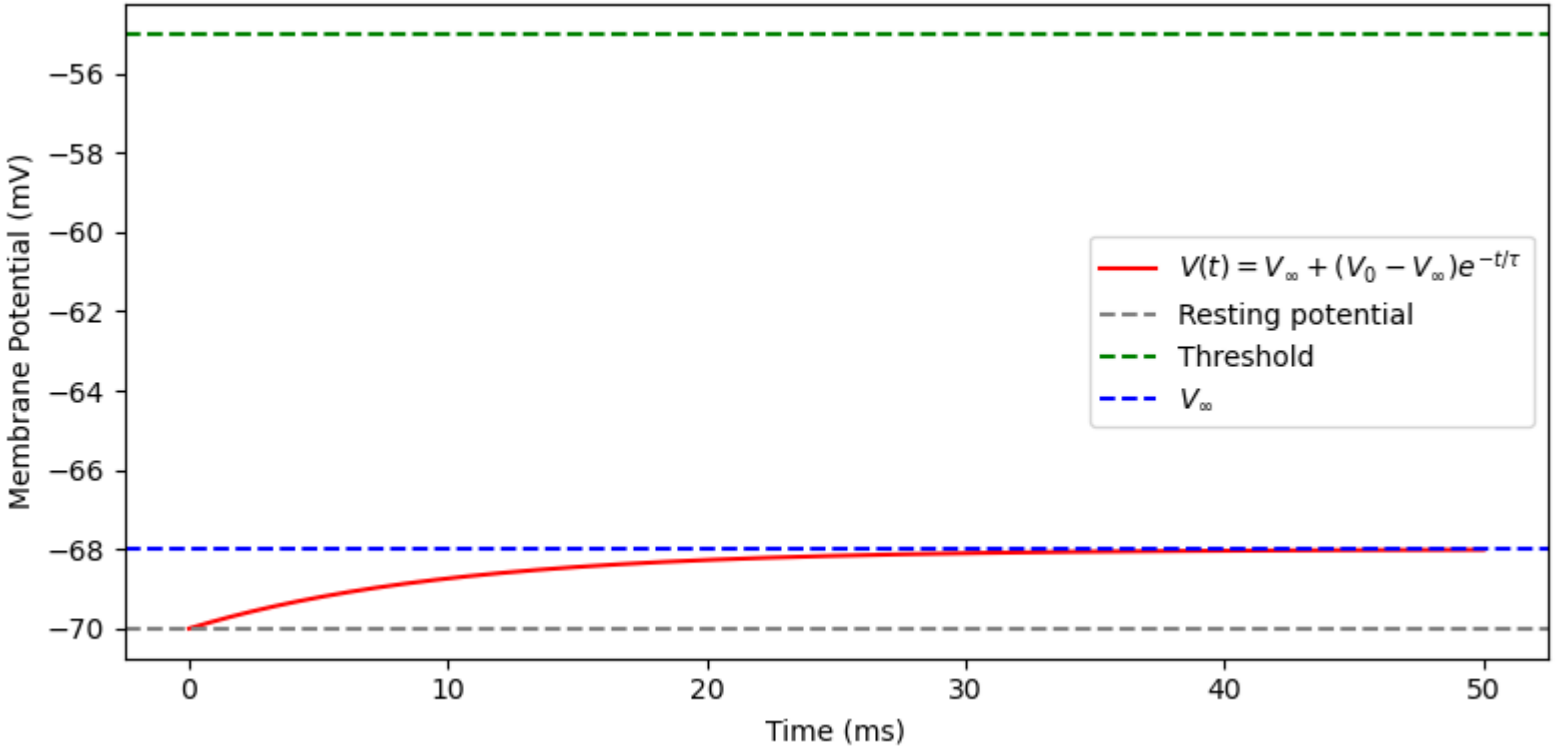
$$\tau \frac{\mathrm{d}V}{\mathrm{d}t} = V_0 - V(t) + RI$$ (constant synaptic input)

Spiking condition

$V(t) \geq V_{\text{thresh}}$ a spike is emitted, and the voltage is reset to V_{rest}

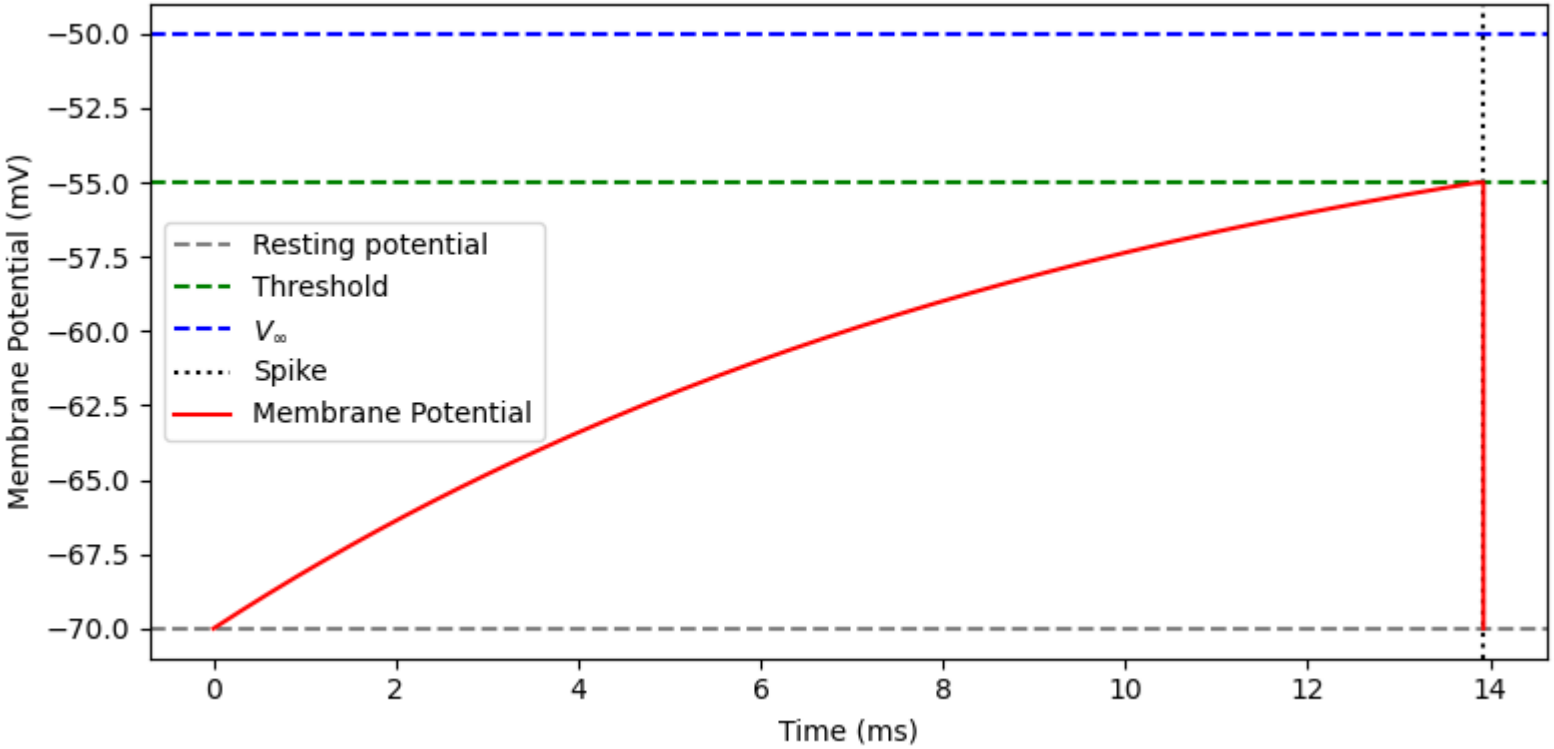
Leaky Integrate-and-Fire (LIF) Neuron with Constant Input Current

$\tau = 10\text{ms}, V_{\text{rest}} = -70\text{mV}, V_{\text{thresh}} = -55\text{mV}, V_0 = V_{\text{rest}}, I = 2(\text{arbitrary units}), R = 1(\text{normalized units})$



Leaky Integrate-and-Fire (LIF) Neuron with Constant Input Current

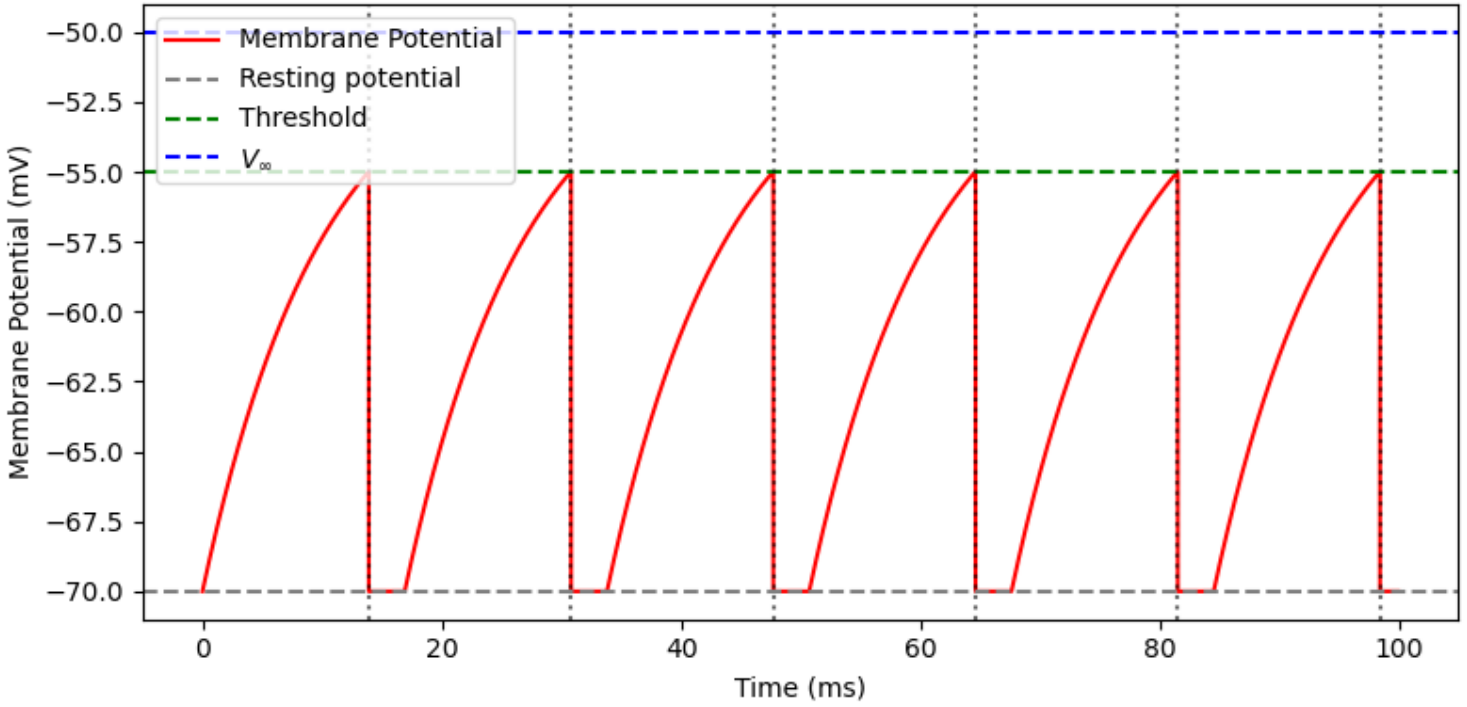
$\tau = 10\text{ms}, V_{\text{rest}} = -70\text{mV}, V_{\text{thresh}} = -55\text{mV}, V_0 = V_{\text{rest}}, I = 20(\text{arbitrary units}), R = 1(\text{normalized units})$



Leaky Integrate-and-Fire (LIF) Neuron with Constant Input Current

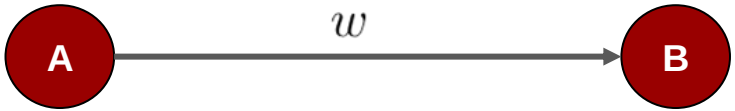
$\tau = 10\text{ms}, V_{\text{rest}} = -70\text{mV}, V_{\text{thresh}} = -55\text{mV}, V_0 = V_{\text{rest}}, I = 20(\text{arbitrary units}), R = 1(\text{normalized units})$

… and a refractory period of 3 ms



Refractory period is the time after a spike during which a neuron cannot spike again

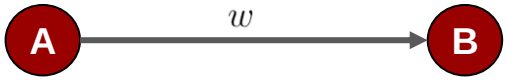
Coupling 2 LIF Neurons



$$\tau \frac{\mathrm{d}V}{\mathrm{d}t} = V_0 - V(t) + RI(t)$$

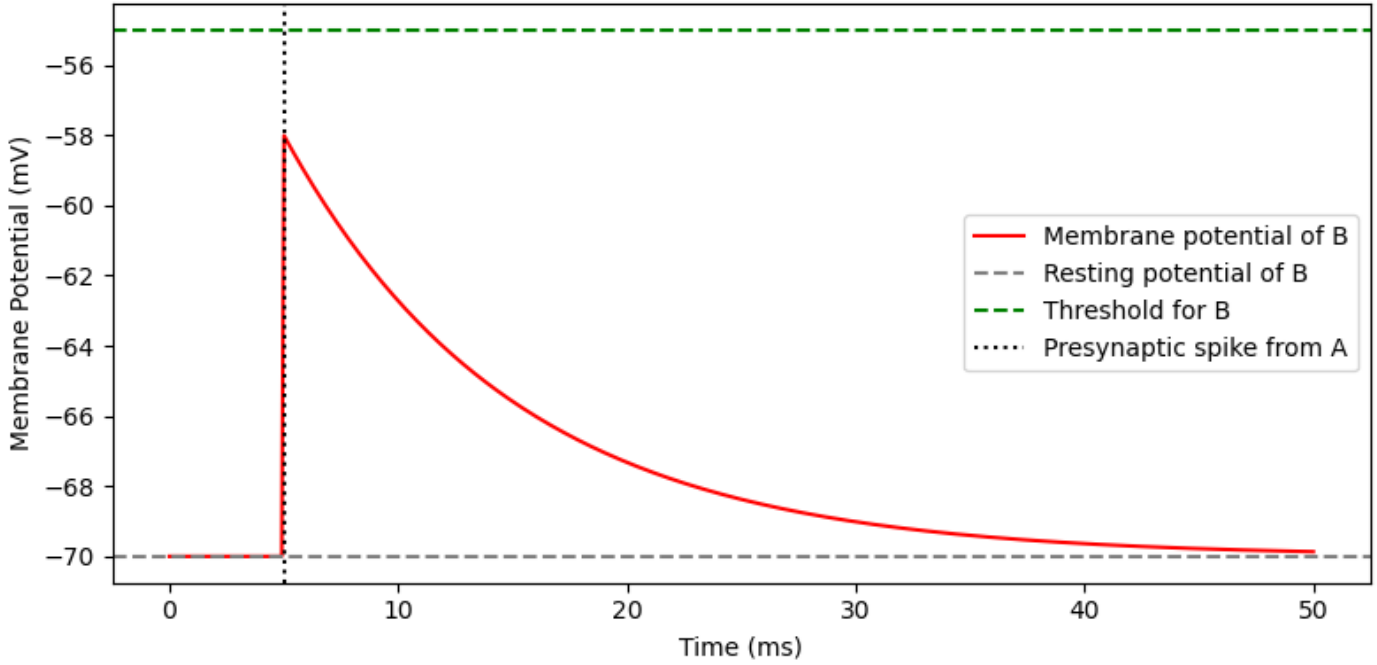
$$I(t) = w\delta(t - t_{\mathrm{spike}})$$

Coupling 2 LIF Neurons

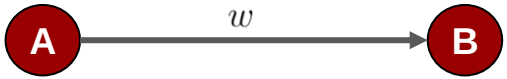


$$\tau \frac{\mathrm{d}V}{\mathrm{d}t} = V_0 - V(t) + RI(t)$$

$$I(t) = w\delta(t - t_{\text{spike}})$$

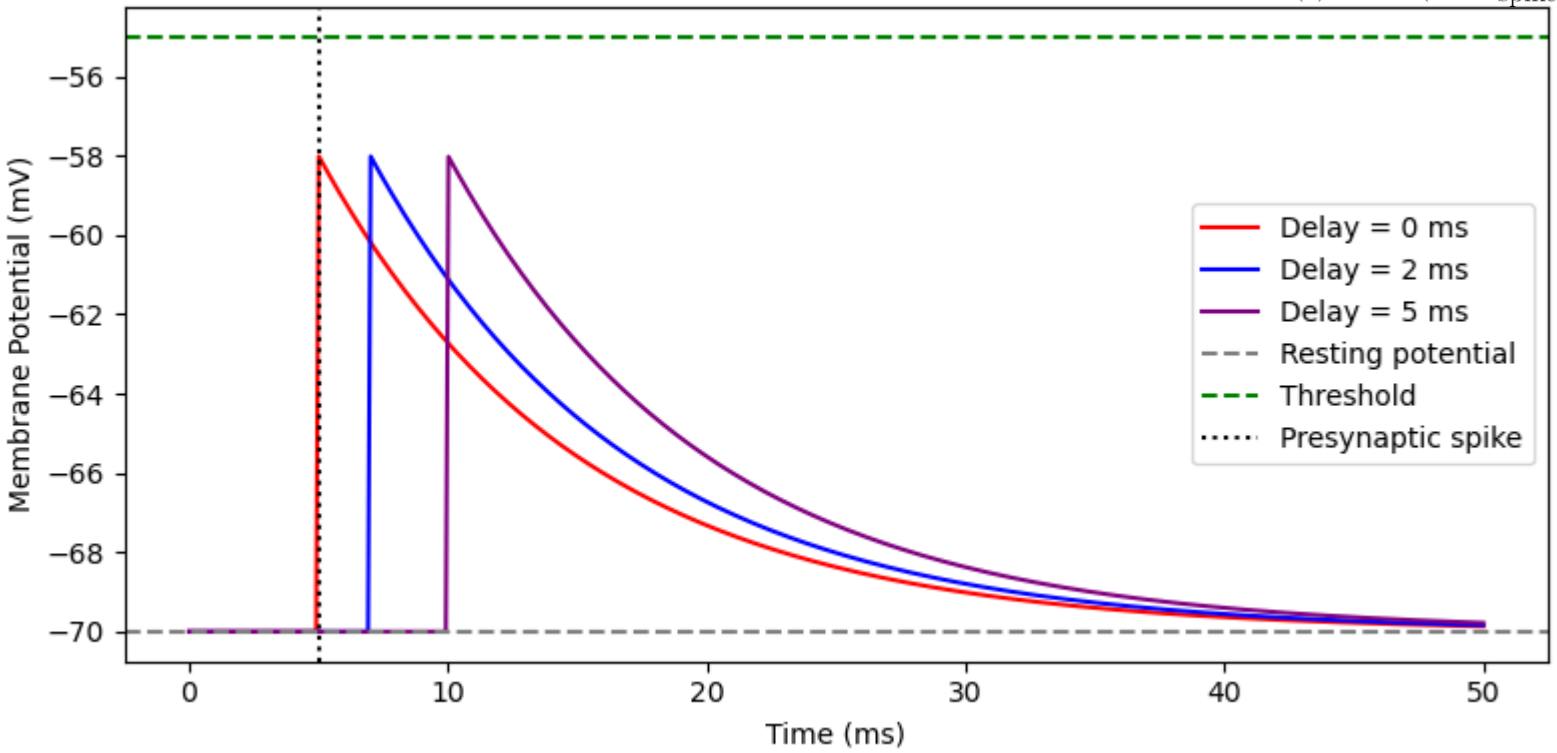


Synaptic Delays

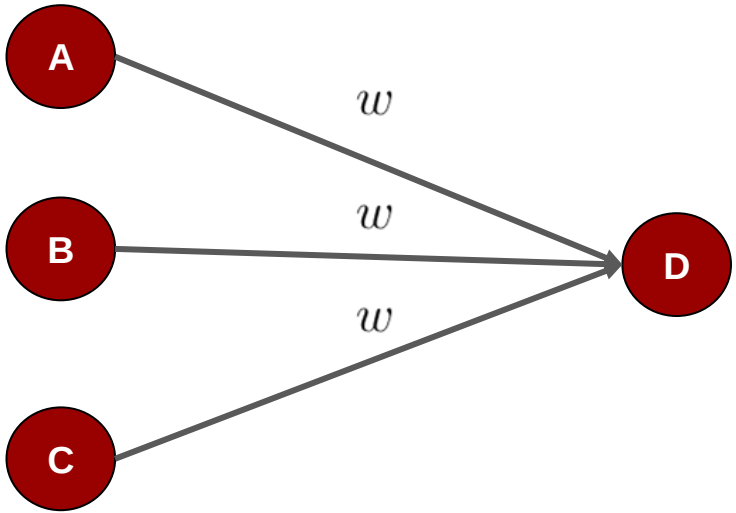


$$\tau \frac{\mathrm{d}V}{\mathrm{d}t} = V_0 - V(t) + RI(t)$$

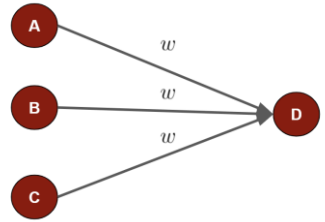
$$I(t) = w\delta(t - t_{\text{spike}})$$

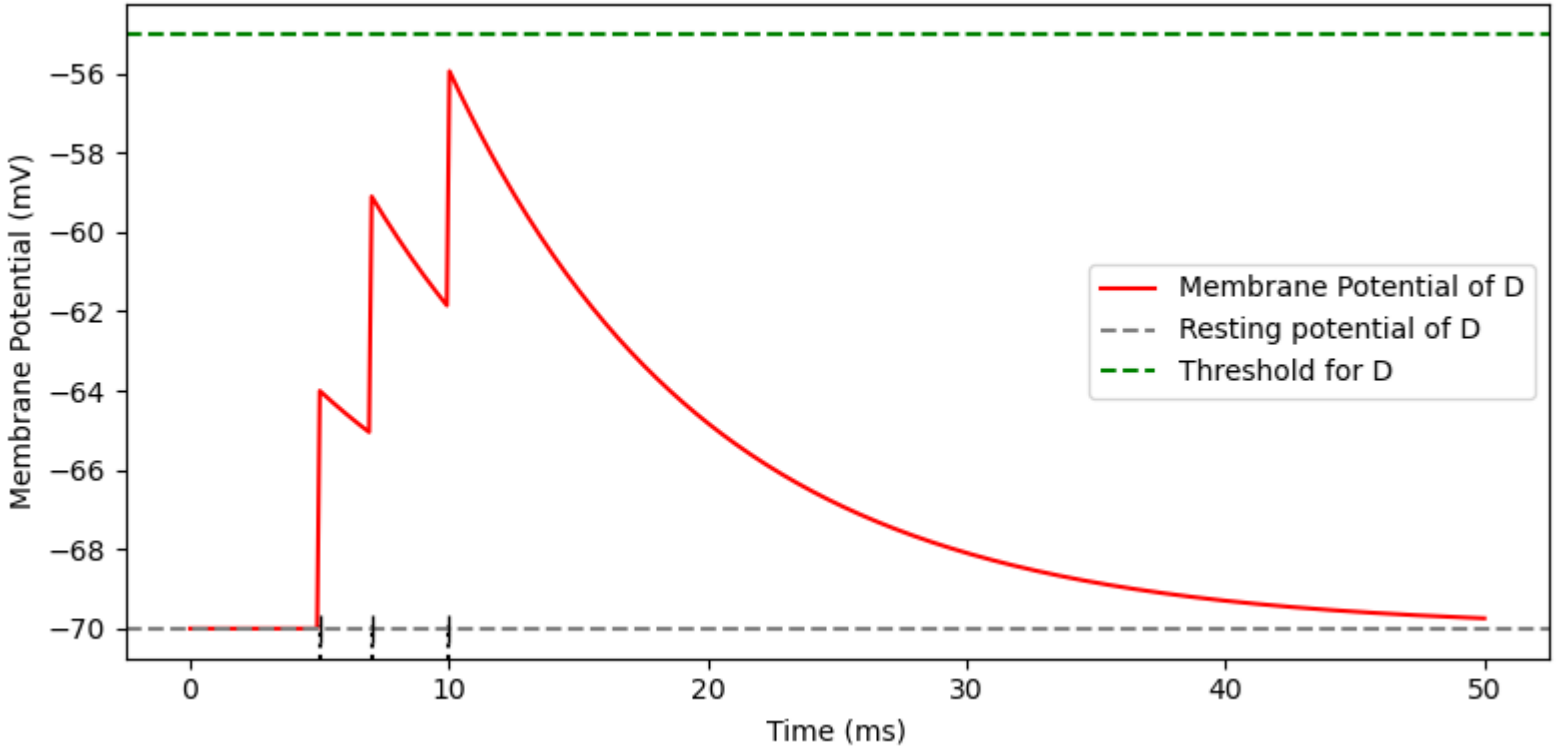


Input from multiple presynaptic neurons with synaptic delays

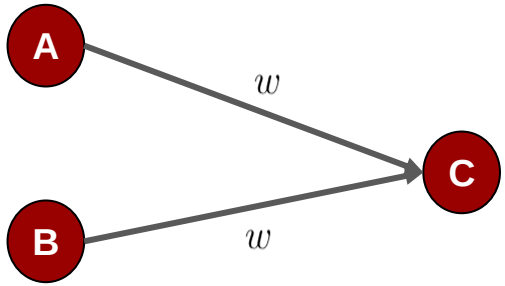


Input from multiple presynaptic neurons with synaptic delays





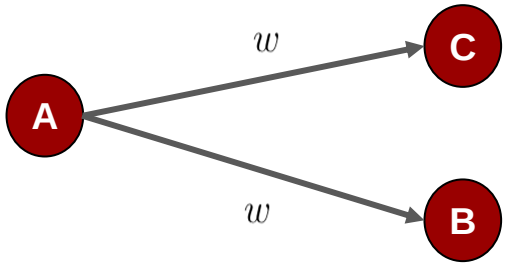
Jitter also matters



Jitter refers to small variations in spike timing between neurons that are part of the same pool.

Exercise: Say we add jitter to the second input. A spikes at t=0, and B spikes at t=δt. For what δt does the second input arrive too late to push the membrane potential of C above the threshold?

Input from a single neuron to multiple postsynaptic neurons

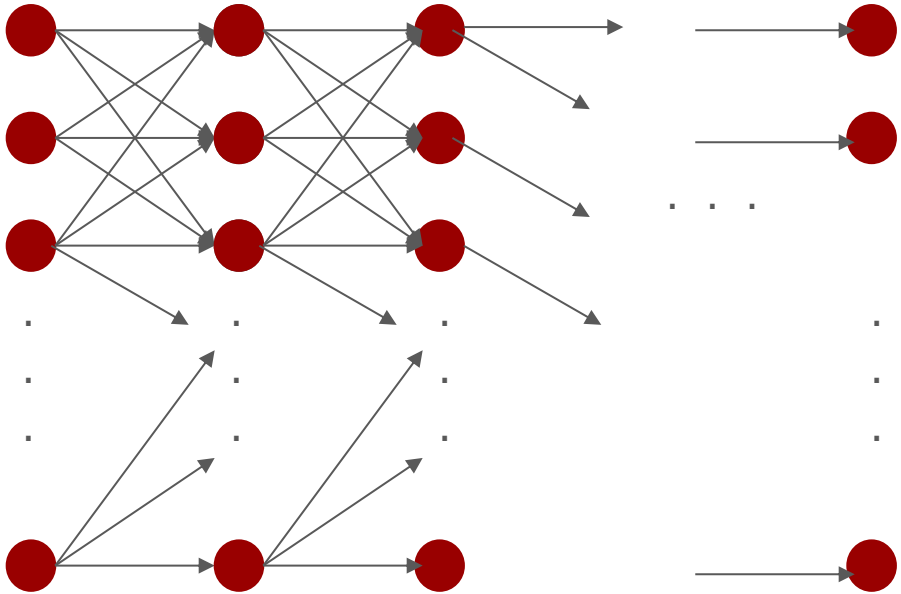


Synaptic weight needed for spiking
w = 16 mV

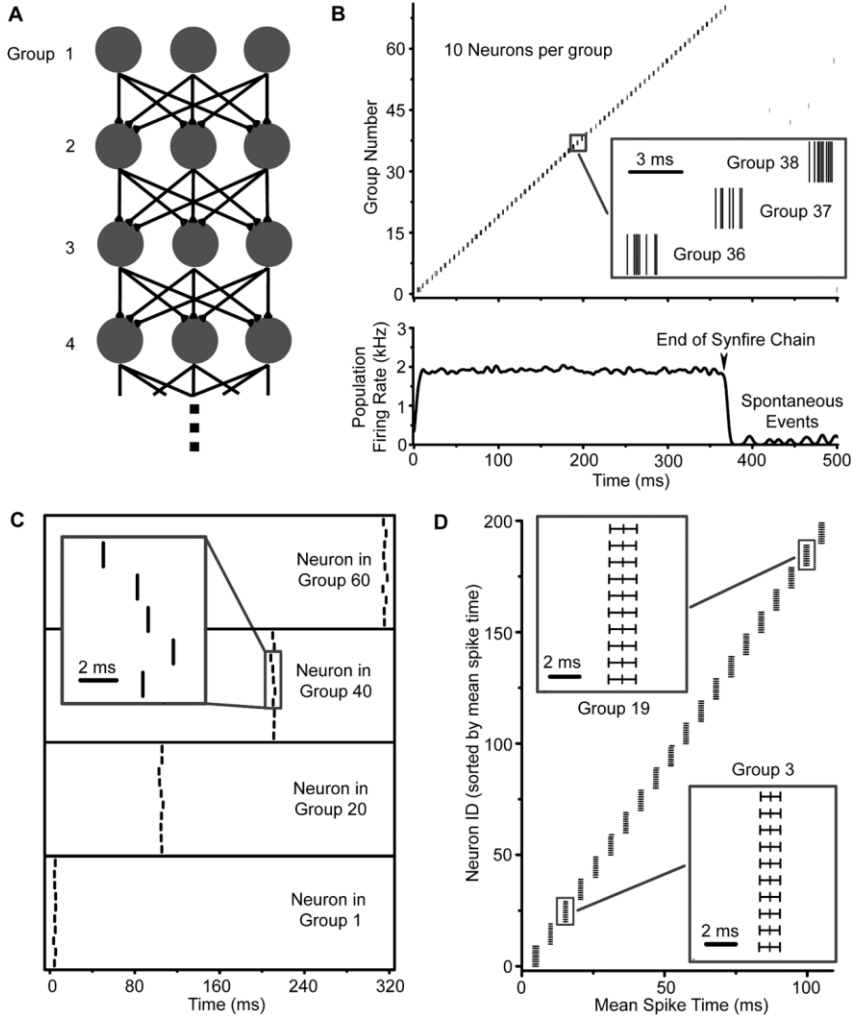
A spikes once. w=16 mV. Do B and C both spike (assuming all other conditions are ideal for spiking)?

Synfire Chain with Leaky Integrate-and-Fire (LIF) Neurons

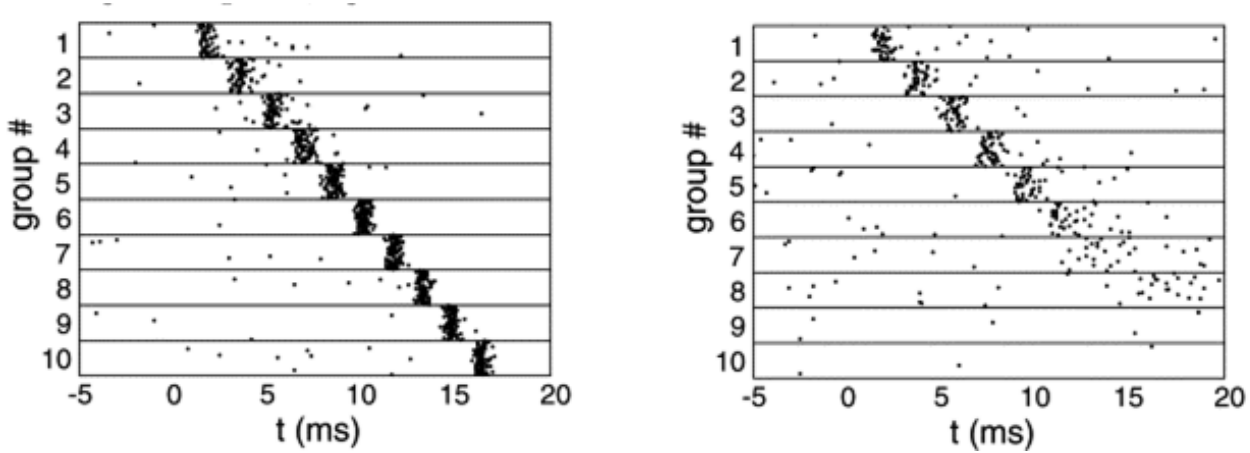
Synfire Chain with Leaky Integrate-and-Fire (LIF) Neurons



Synfire Chain and Spike Activity



(Jun & Jin, 2007)

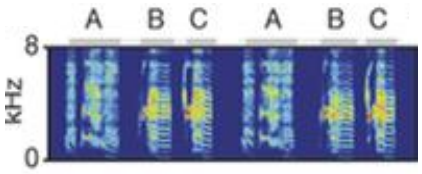


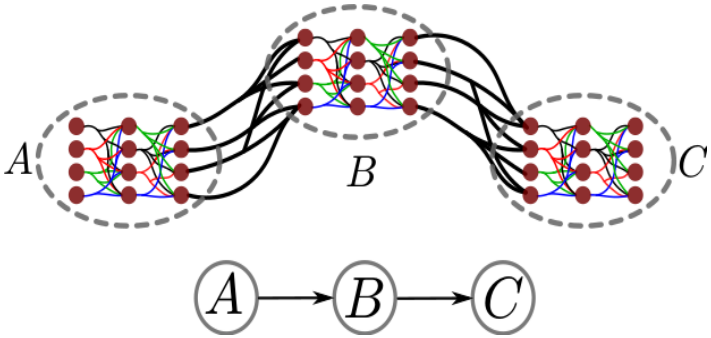
- A spike volley propagates if inputs are strong, synchronous, and well-timed.
- Jitter, insufficient weight, delay mismatches etc can lead to propagation failure in a synfire chain.
- Exercise: Simulate a synfire chain with different neuron models and check how crucial timing, decay, and synchrony are.

Synfire chains in HVC with every volley having one destination could encode the song syntax for deterministic song

Zebra finch

Deterministic song

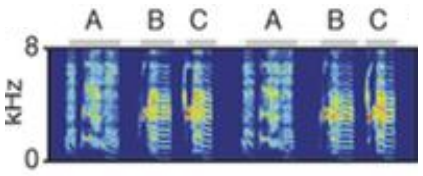




Song sequences or motor programs where one state can lead to multiple possible outcomes require branching and probabilistic transitions

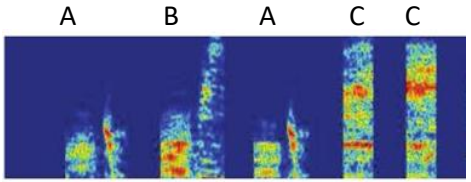
Zebra finch

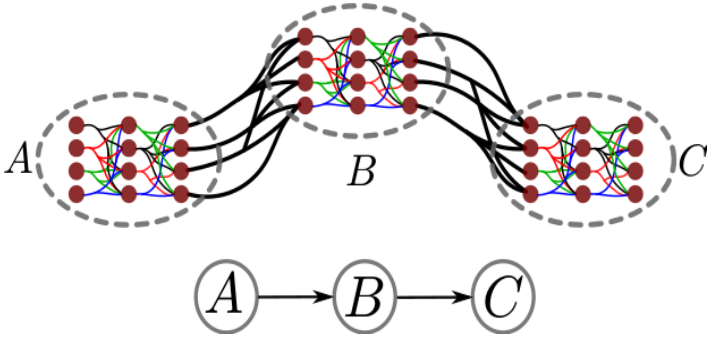
Deterministic song

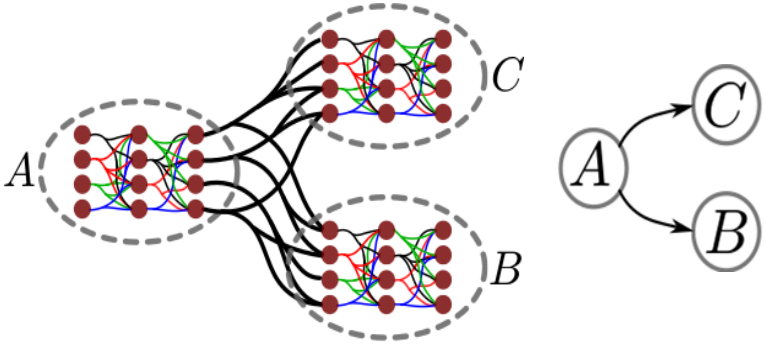


Bengalese finch

Stochastic song





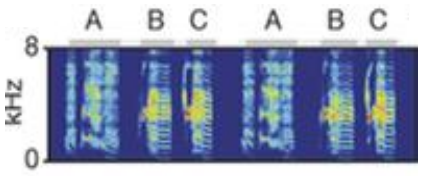


(Jin, 2009)

Song sequences or motor programs where one state can lead to multiple possible outcomes require branching and probabilistic transitions

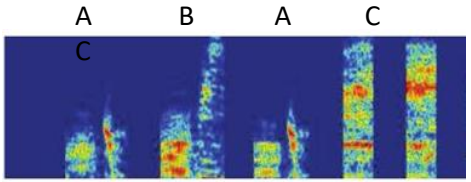
Zebra finch

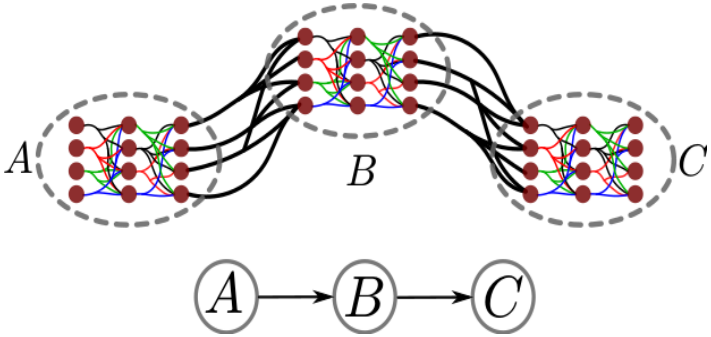
Deterministic song

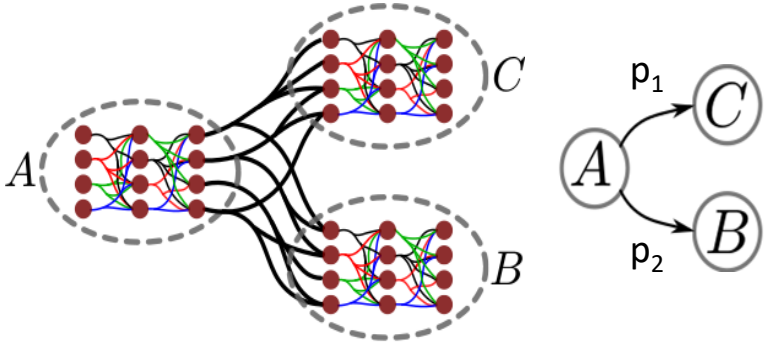


Bengalese finch

Stochastic song

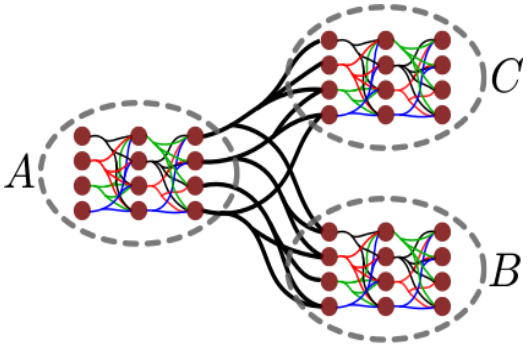




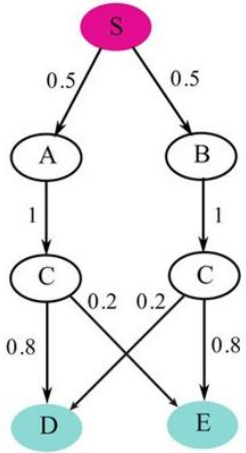


(Jin, 2009)

Brain's process Q



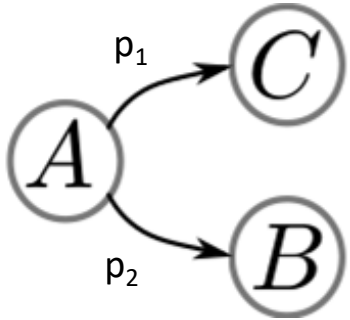
Syntax Model P



We want: P ≅ Q

How do we create probabilistic models of the syntax from behaviour (song)?

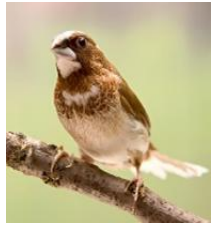
?



(Image: Richard Wilkinson as credited in New Scientist, 2014)

How are temporally precise sequences generated neurally?

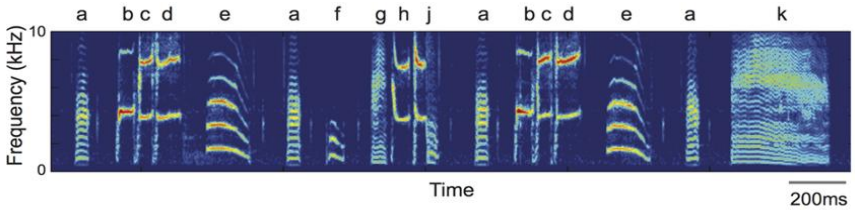
Tutorial 1: A neural model of birdsong sequence production



What is the syntax underlying these sequences?

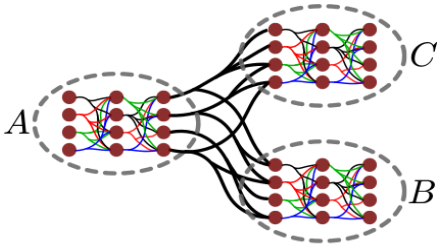
Tutorial 2: Hidden Markov Models for sequence modelling

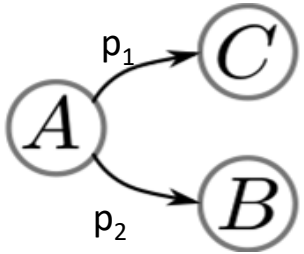
Stochastic song



How do models for the two inform each other?

Tutorial 3: Evaluating probabilistic models of sequence production

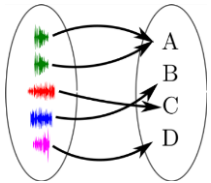




Tutorial 2

Hidden Markov Models for Sequence Modelling

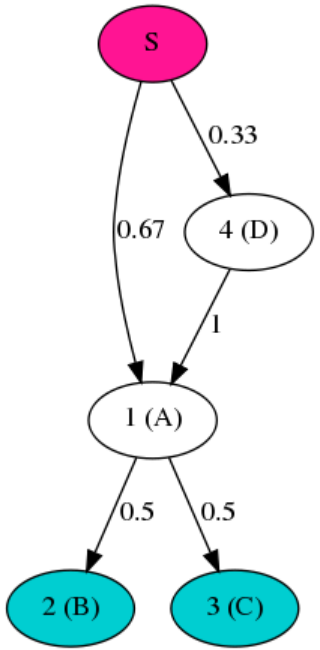
How do we create probabilistic models of the syntax from behaviour (song)?



A B
D A B
A C
D A B
D A B
A B
A C
A C
A B
A C
A C
D A B
A B
D A B

How do we create probabilistic models of the syntax from behaviour (song)?

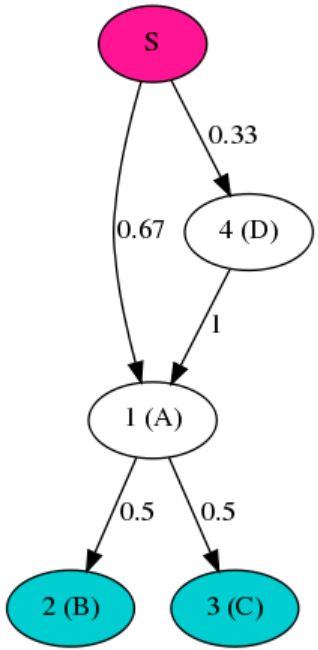
A B
D A B
A C
D A B
D A B
A B
A C
A C
A B
A C
A C
D A B
A B
D A B



First order Markov model

How do we create probabilistic models of the syntax from behaviour (song)?

A B
D A B
A C
D A B
D A B
A B
A C
A C
A B
A C
A C
D A B
A B
D A B

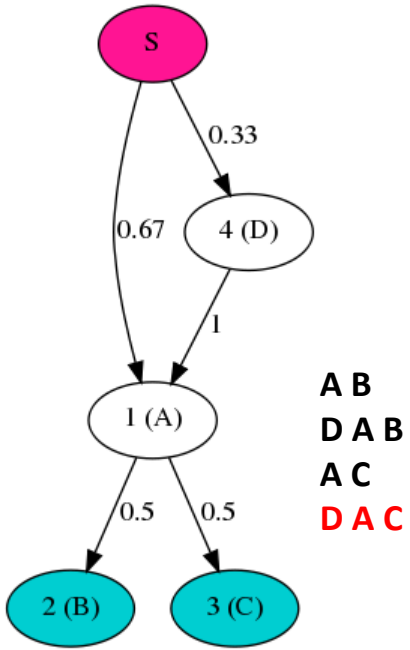


A B
D A B
A C

First order Markov model

How do we create probabilistic models of the syntax from behaviour (song)?

A B
D A B
A C
D A B
D A B
A B
A C
A C
A B
A C
A C
D A B
A B
D A B

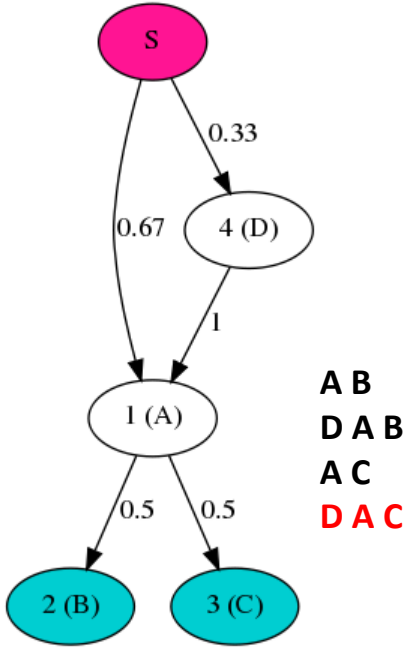


A B
D A B
A C
D A C

First order Markov model

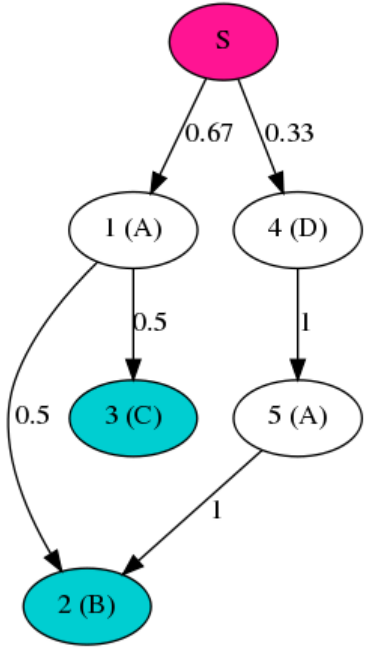
How do we create probabilistic models of the syntax from behaviour (song)?

A B
D A B
A C
D A B
D A B
A B
A C
A C
A B
A C
A C
D A B
A B
D A B



A B
D A B
A C
D A C

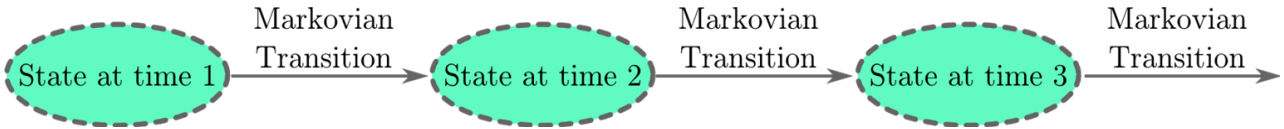
First order Markov model



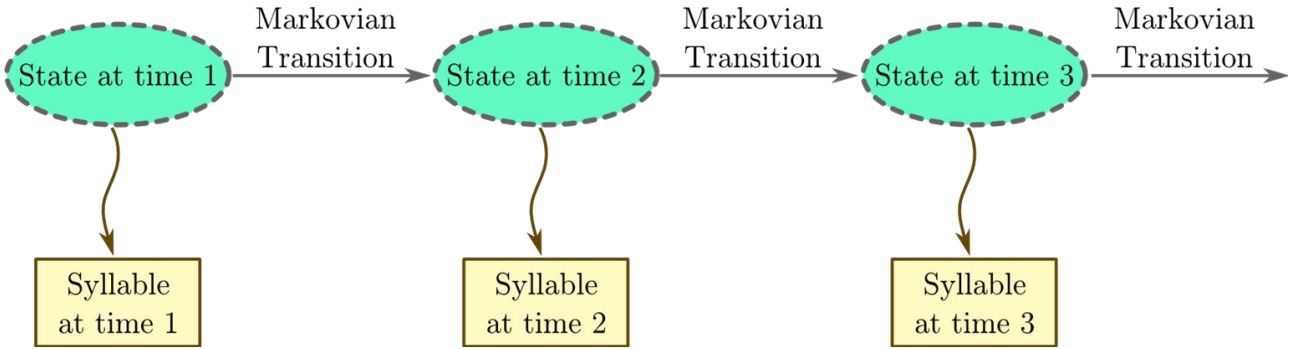
A B
D A B
A C

A model with hidden states

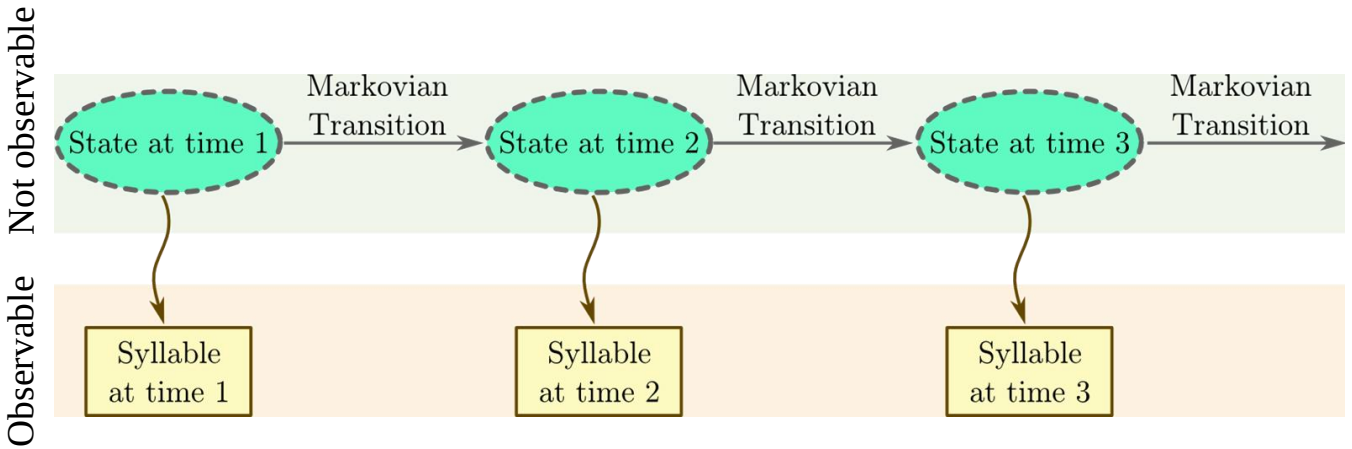
A non-observable process results in an observable sequence of symbols



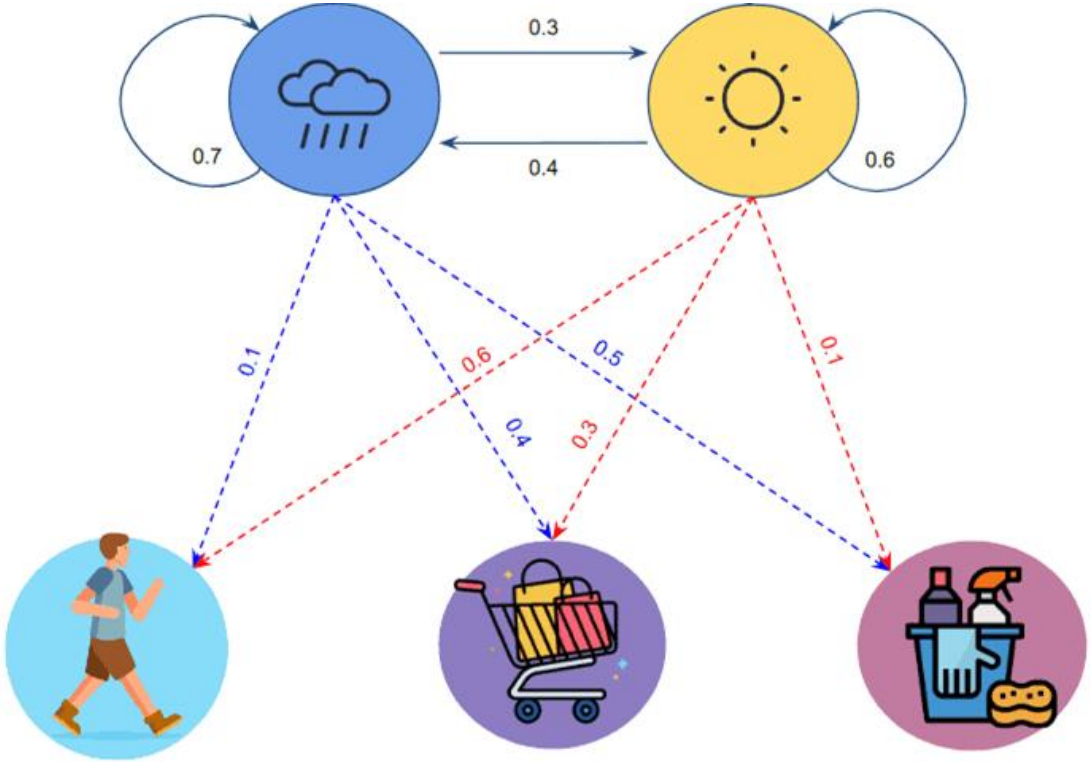
A non-observable process results in an observable sequence of symbols



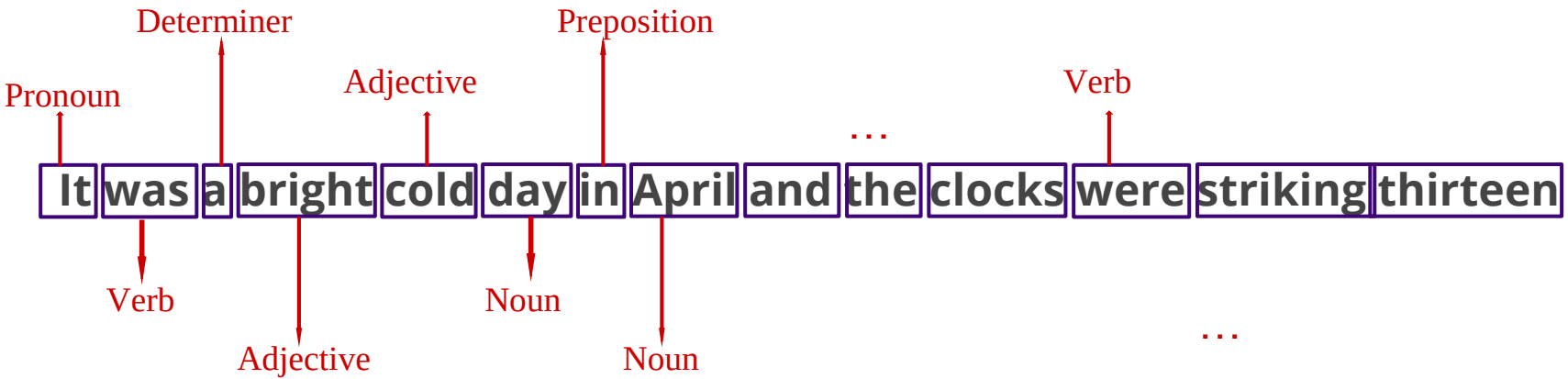
A non-observable process results in an observable sequence of symbols



Hidden Markov Model



baeldung.com/cs/hidden-markov-model


Pronoun
Determiner
Adjective
Preposition
…
Verb
It was a bright cold day in April and the clocks were striking thirteen
Verb
Adjective
Noun
Noun
…

Parameters of HMM

1. N, the number of states in the model. The set of all possible states is $S = \{S_1, S_2, ..., S_N\}$, and the state at time t as q_t.

2. M, the number of distinct observation symbols per state. The set of all possible output symbols is $V = \{v_1, v_2, ..., v_M\}$, and the output symbol at time t as O_t. The sequence of observed symbols is $O = O_1\ O_2\ ...\ O_T$.

3. The state transition probability distribution $A = \{a_{ij}\}$, where $a_{ij} = P[q_{t+1} = S_j \mid q_t = S_i]$, 1

4. The observation symbol probability distribution in state j, $B = \{b_j(k)\}$, where $b_j(k) = P[O_t = v_k \mid q_t = S_j]$, $1 \le j \le N$, $1 \le k \le M$.

5. The initial state distributions $\pi = \{\pi_i\}$, where $\pi_i = P[q_1 = S_i]$, $1 \le i \le N$.

$$A = \begin{array}{c} \\ 1 \\ \dots \\ i \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & j & \dots \end{array} \quad \dots N \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & a_{i,j} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$

$$B = \begin{array}{c} \\ 1 \\ \dots \\ j \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & k & \dots \end{array} \quad \dots M \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & b_j(k) & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$

Model $\lambda = (A, B, \pi)$

Three Problems of HMM

Problem 1: Given the observation sequence $O = O_1\ O_2\ ...\ O_T$ and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?

Problem 2: Given the observation sequence $O = O_1\ O_2\ ...\ O_T$ and the model $\lambda = (A, B, \pi)$ how do we choose a corresponding state sequence $Q = q_1 q_2 \ldots q_T$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?

Problem 3: How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

Three Problems of HMM

Problem 1: Given the observation sequence $O = O_1 O_2 \ldots O_T$ and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?

Problem 2: Given the observation sequence $O = O_1 O_2 \ldots O_T$ and the model $\lambda = (A, B, \pi)$ how do we choose a corresponding state sequence $Q = q_1 q_2 \ldots q_T$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?

Problem 3: How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

Forward-Backward Algorithm

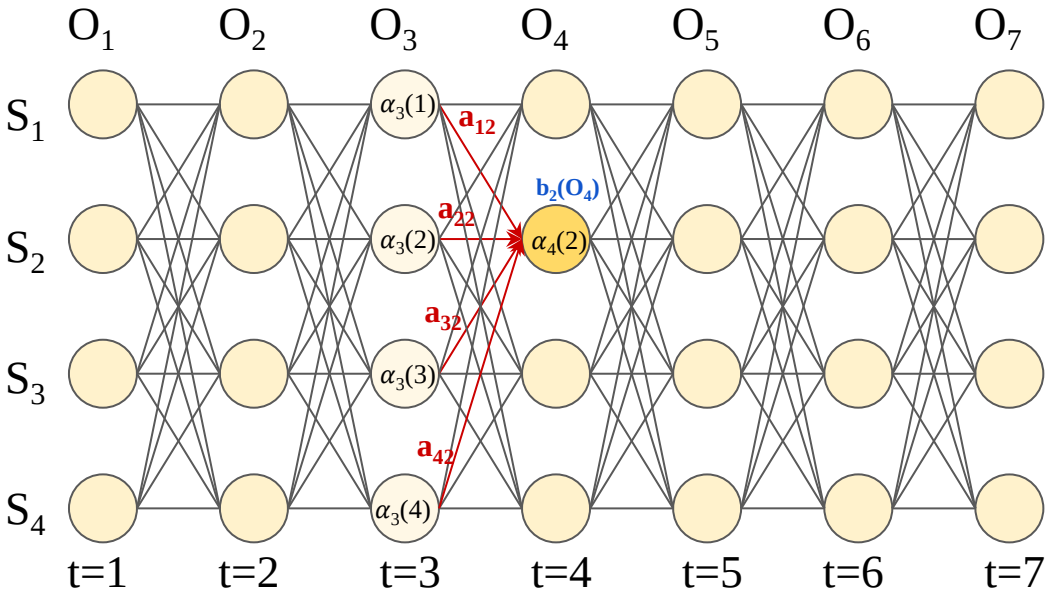
Forward variable α_t(i)
Probability of seeing the first t observations and ending up in state S_i

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \leq t \leq T-1, 1 \leq j \leq N$$

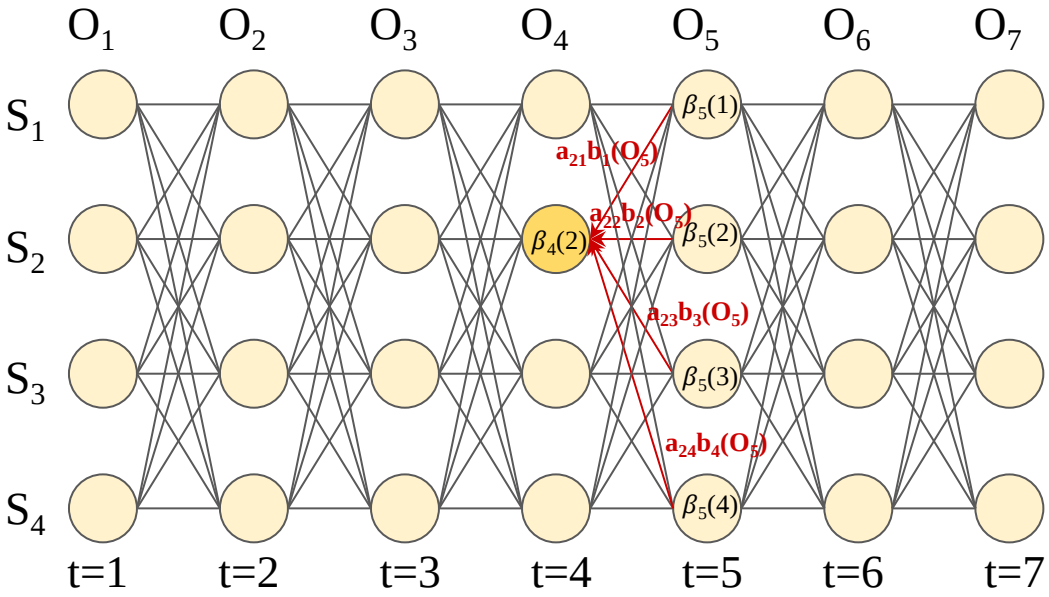
Backward variable β_t(i)
Probability of seeing the rest of the observations (from time t+1 to the end), given that you are in state S_i at time t

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

Trellis for computation of the forward variables



Trellis for computation of the backward variables



The probability of being in state S_i at time t, given the observation sequence O, and the model λ

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

The probability of being in state S_i at time t and state S_j at time t+1, given the observation sequence O, and the model λ

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

Baum-Welch Algorithm (An Expectation-Maximization Algorithm)

E-Step (E for Expectation)

Estimate how likely it is that the system was in each hidden state at each time, and how likely each state transition was, given the observed sequence and current model parameters

$\sum_{t=1}^{T-1} \gamma_t(i)$

can be interpreted as the expected number of times S_i is visited or the expected number of transitions made from state S_i

$\sum_{t=1}^{T-1} \xi_t(i,j)$

can be interpreted as the expected number of transitions from state S_i to state S_j

Baum-Welch Algorithm (An Expectation-Maximization Algorithm)

M-Step (M for Maximization)

Update the model parameters to maximize the expected complete-sequence log-likelihood based on the quantities computed in the E-step

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

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1, O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

$$\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$$

Baum-Welch Algorithm (An Expectation-Maximization Algorithm)

After the parameter update

$$\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$$

Importantly, each iteration guarantees

$$P(O \mid \bar{\lambda}) \geq P(O \mid \lambda)$$

(proof by Baum and collaborators in the 1970s)

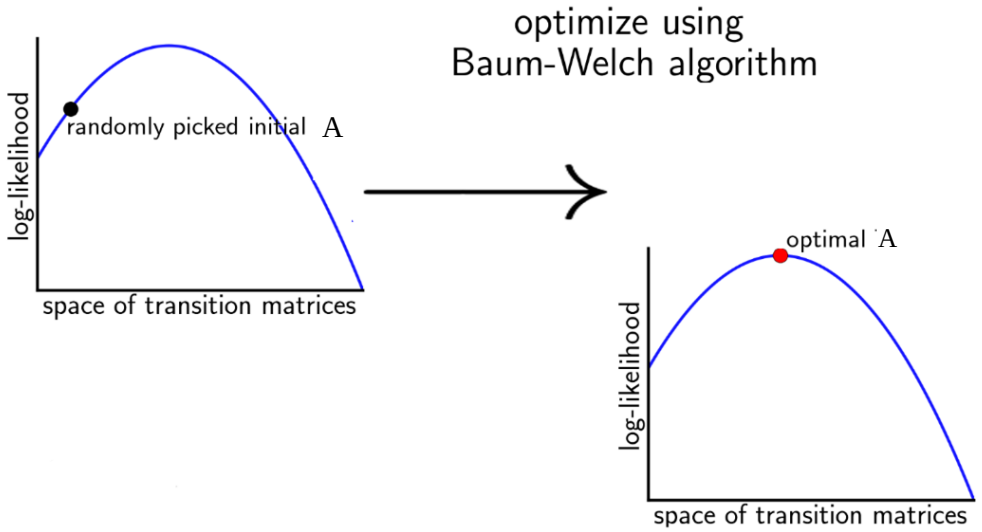
Typically we keep track of

$$\log P(O \mid \bar{\lambda})$$

The E-M process is repeated again and again until the change in parameters is smaller than some pre-defined threshold

$$A = \begin{array}{c} \\ 1 \\ \dots \\ i \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & j & \dots \end{array} \quad \dots N \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & a_{i,j} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$

$$B = \begin{array}{c} \\ 1 \\ \dots \\ j \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & k & \dots \end{array} \quad \dots M \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & b_j(O_k) & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$



A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition

LAWRENCE R. RABINER, FELLOW, IEEE

Although initially introduced and studied in the late 1960s and early 1970s, statistical methods of Markov source or hidden Markov modeling have become increasingly popular in the last several years. There are two strong reasons why this has occurred. First the models are very rich in mathematical structure and hence can form the theoretical basis for use in a wide range of applications. Second the models, when applied properly, work very well in practice for several important applications. In this paper we attempt to carefully and methodically review the theoretical aspects of this type of statistical modeling and show how they have been applied to selected problems in machine recognition of speech.

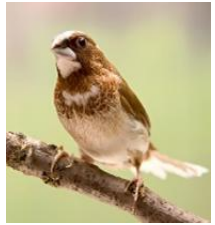
In this case, with a good signal model, we can simulate the source and learn as much as possible via simulations. Finally, the most important reason why signal models are important is that they often work extremely well in practice, and enable us to realize important practical systems—e.g., prediction systems, recognition systems, identification systems, etc., in a very efficient manner.

These are several possible choices for what type of signal model is used for characterizing the properties of a given signal. Broadly one can dichotomize the types of signal

(1989)

How are temporally precise sequences generated neurally?

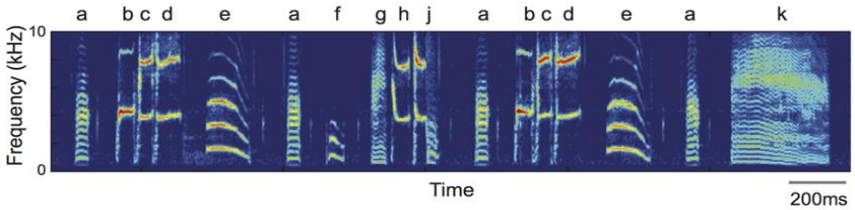
Tutorial 1: A neural model of birdsong sequence production



What is the syntax underlying these sequences?

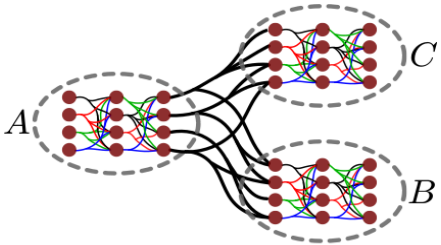
Tutorial 2: Hidden Markov Models for sequence modelling

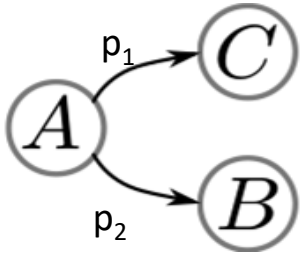
Stochastic song



How do models for the two inform each other?

Tutorial 3: Evaluating probabilistic models of sequence production

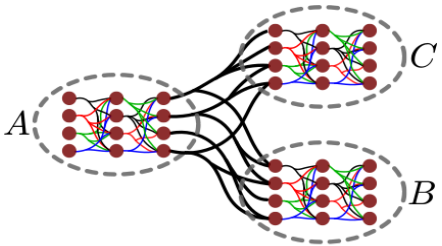


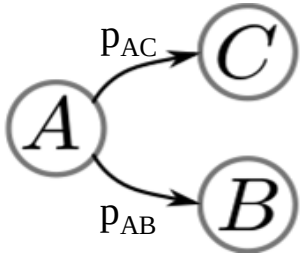


Tutorial 3

Evaluating Probabilistic Models of Sequence Production

Bridging Levels





Neural models are mechanistic

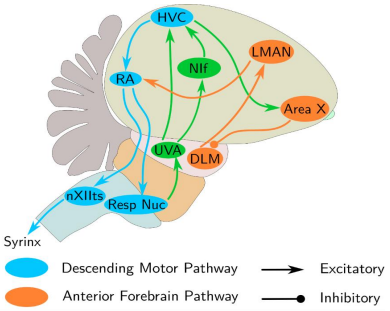
Process explanation

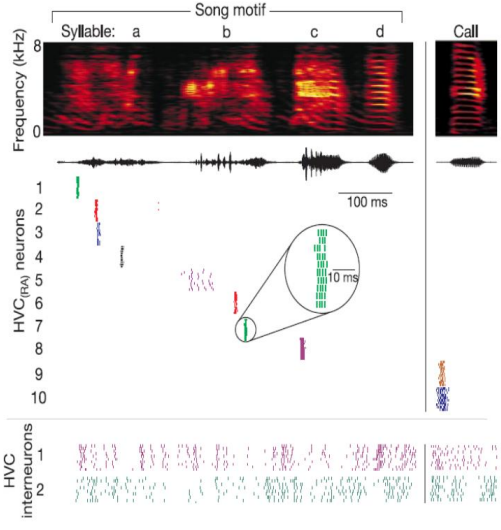
Probabilistic models are phenomenological

Pattern description

Together they allow us to bridge implementation and behaviour

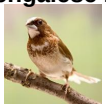
Networks that keep time

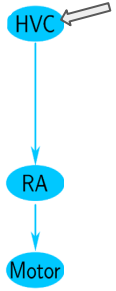




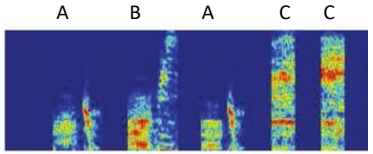
Zebra finch

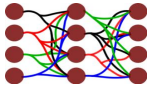
Bengalese finch



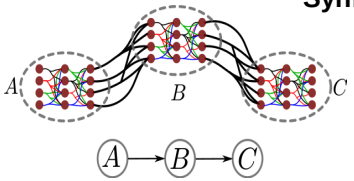


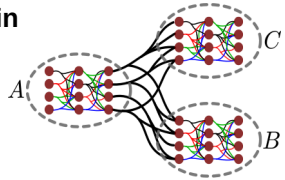
Stochastic song



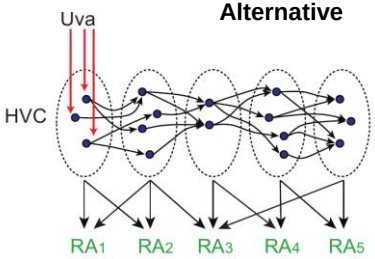


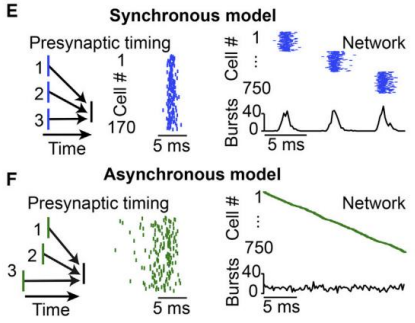
Synfire chain





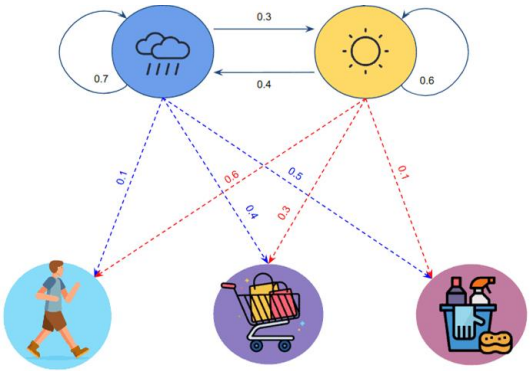
Alternative





(Egger & Tupikov et al, 2020)

Inference of Probabilistic Models: HMM and the Baum-Welch Algorithm



$$A = \begin{array}{c} \\ 1 \\ \dots \\ i \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & j & \dots \end{array} \quad \dots \; N \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & a_{i,j} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$

$$B = \begin{array}{c} \\ 1 \\ \dots \\ j \\ \dots \\ N \end{array} \begin{array}{c} \begin{array}{cccc} 1 & \dots & k & \dots \end{array} \quad \dots \; M \\ \begin{bmatrix} \dots & \dots & \dots & \dots \\ \dots & b_j(k) & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \end{array}$$

Goal: Estimate parameters $\lambda = (A, B, \pi)$ of a Hidden Markov Model from observed data.

Initialize model parameters $\lambda = (A, B, \pi)$

Expectation Step (E-step)
For each time step t, we calculate
$\alpha_t(i)$: forward probability = $P(O_1,...,O_t, q_t = i \mid \lambda)$
$\beta_t(i)$: backward probability = $P(O_{t+1},...,O_T \mid q_t = i, \lambda)$
$\gamma_t(i)$: state occupancy = $P(q_t = i \mid O, \lambda)$
$\xi_t(i,j)$: state transition = $P(q_t = i, q_{t+1} = j \mid O, \lambda)$

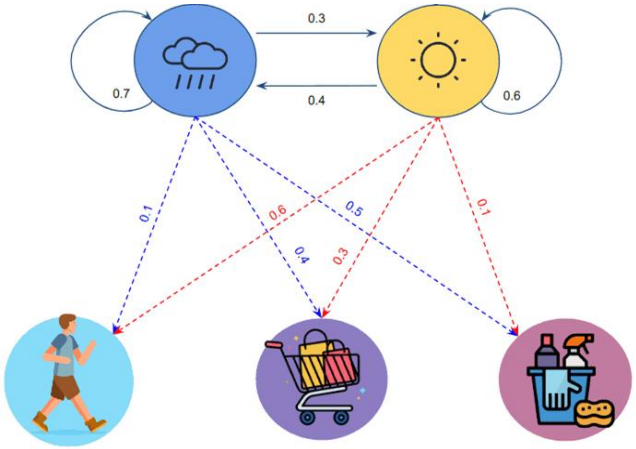
Maximization Step (M-step)
Update parameters

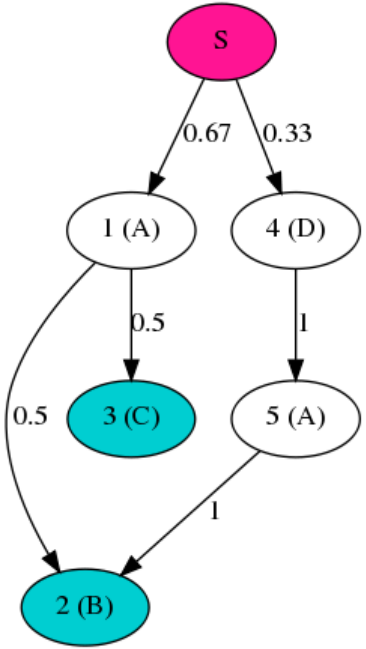
$$\bar{\pi}_i = \gamma_1(i) \quad \bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad \bar{b}_j(k) = \frac{\sum_{t=1,\, O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Iterate E-step and M-step until log-likelihood converges

Guarantees non-decreasing likelihood
Converges to a local optimum
Requires scaling for numerical stability
Form of the HMM is assumed to be known

Hidden Markov Model (HMM) vs Partially Observable Markov Model (POMM)



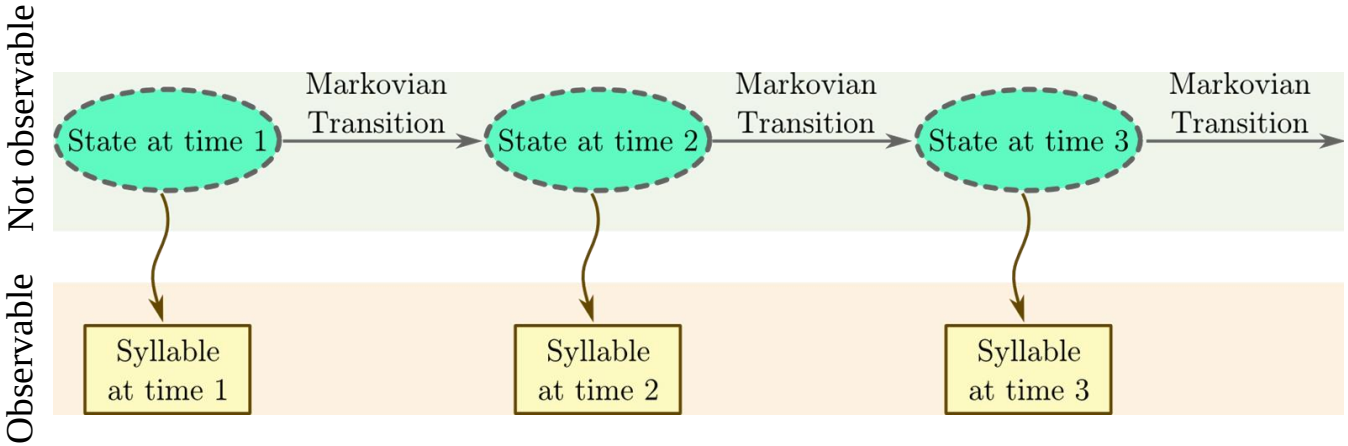


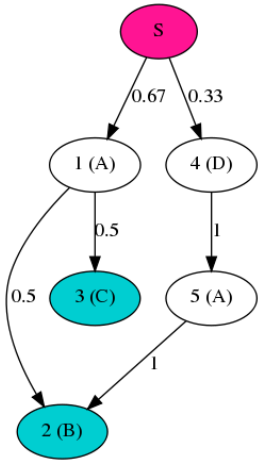
HMM: Each state can emit all symbols

POMM: Each state can emit only one symbol

Multiple states may correspond to the same symbol

Hidden Markov Model (HMM) vs Partially Observable Markov Model (POMM)





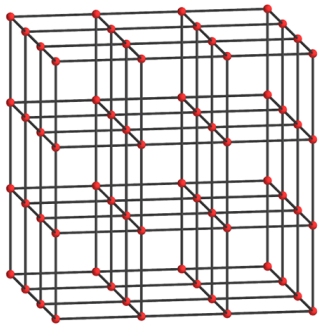
$T =$

	α	1	2	3	4	5	ω
α	0.67	0	0	0.33	0	0	0
1	0	0	0.5	0.5	0	0	0
2	0	0	0	0	0	0	1
3	0	0	0	0	0	0	1
4	0	0	0	0	0	1	0
5	0	0	1	0	0	0	0
ω	0	0	0	0	0	0	1

$E =$

	α	A	B	C	D	ω
α	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	0	0
4	0	0	0	0	1	0
5	0	1	0	0	0	0
ω	0	0	0	0	0	1

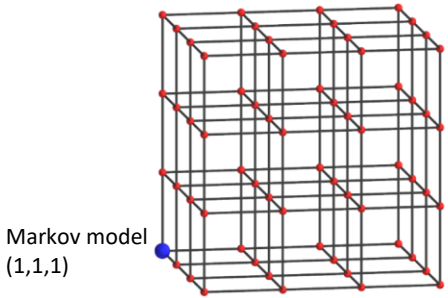
Grid-search to find the optimal POMM

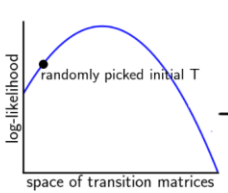


Discrete lattice of dimension # of syllables

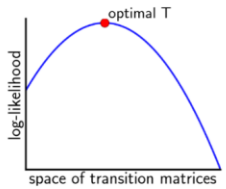
total # of states ↓

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ t_{41} & t_{42} & t_{43} & t_{44} \end{bmatrix}$$

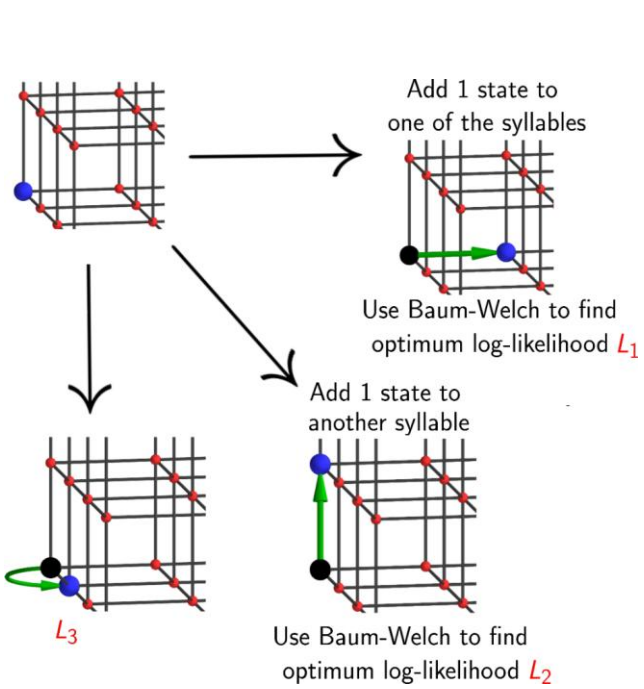


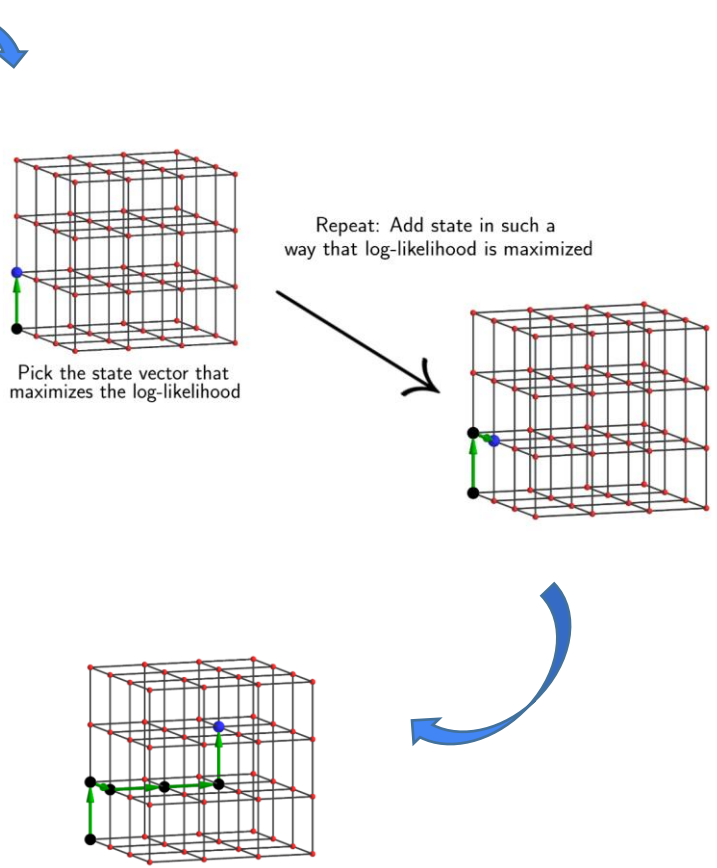


optimize using
Baum-Welch algorithm

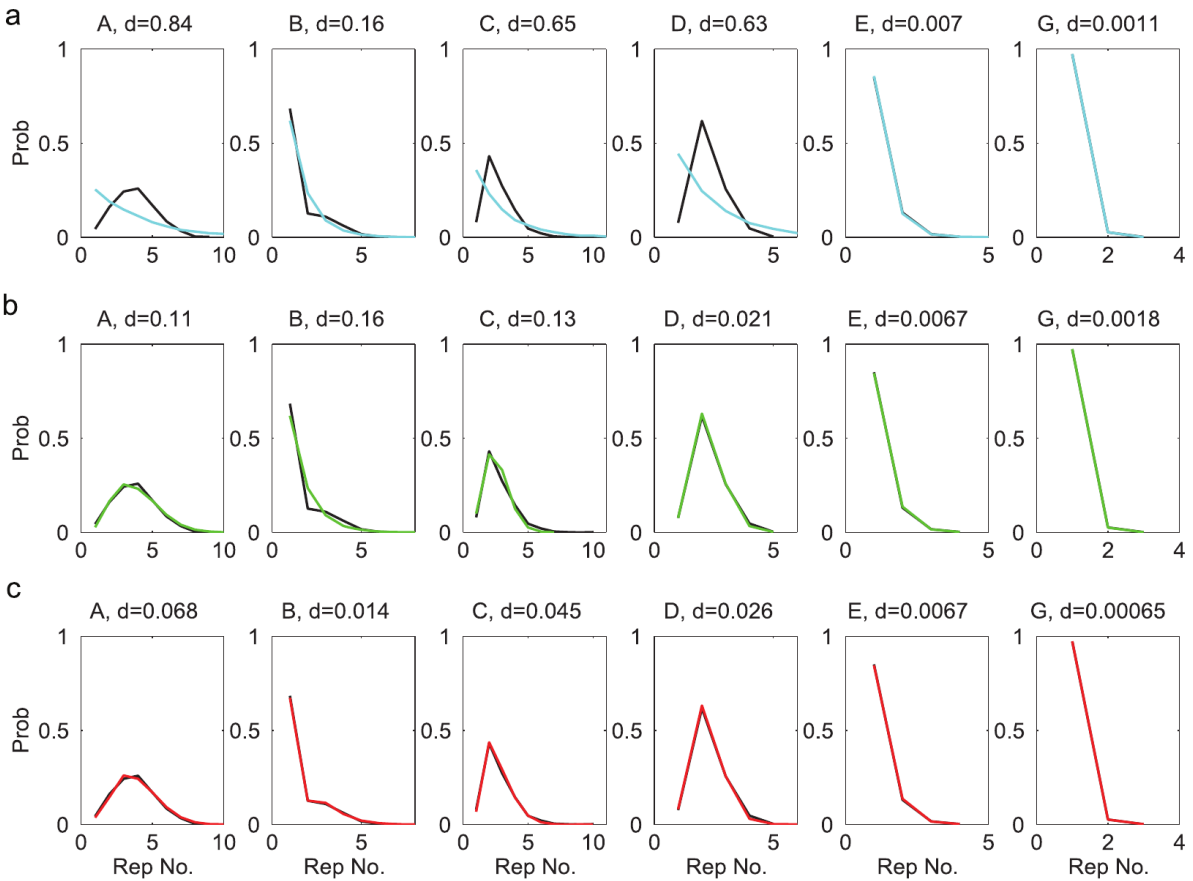


Grid-search to find the optimal POMM





Testing models by comparing sequence statistics: Repeat distributions

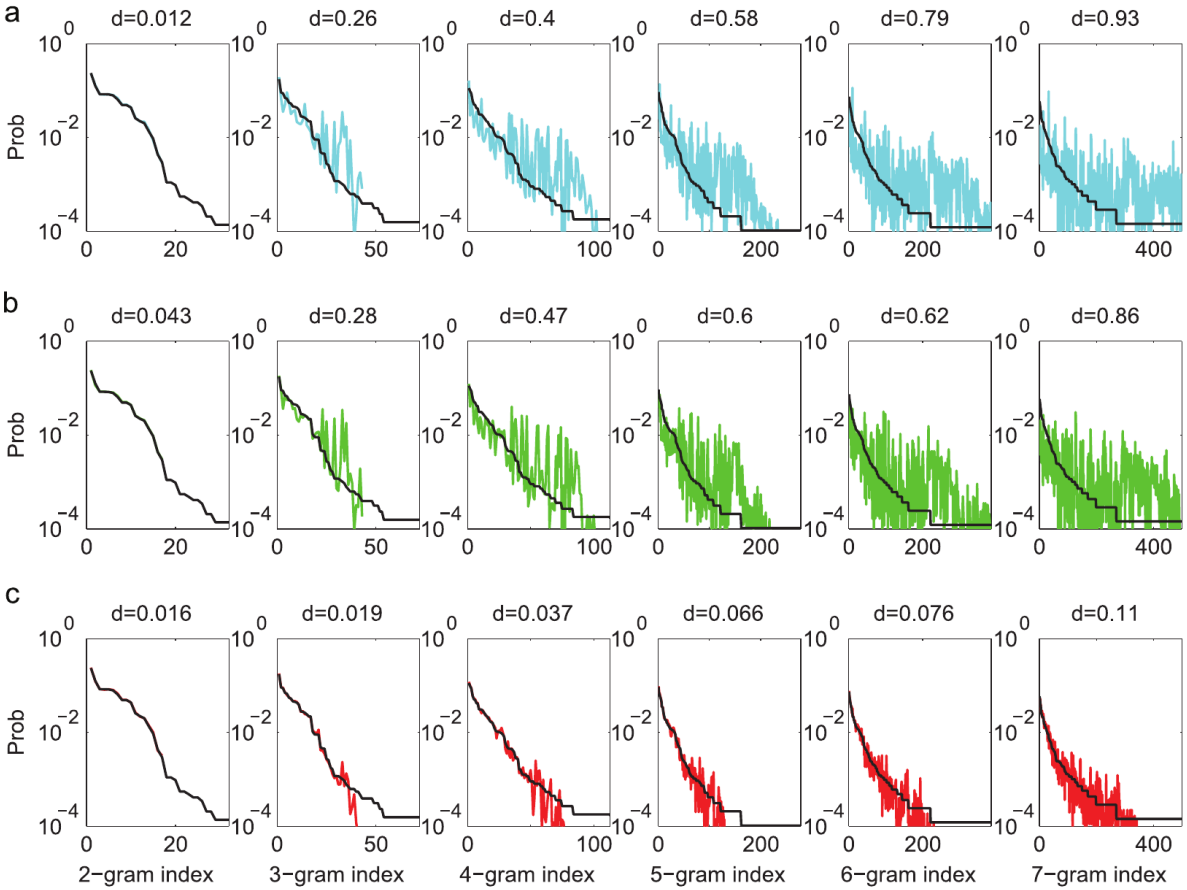


Markov Model

Markov Model with Adaptation

POMM with Adaptation

Testing models by comparing sequence statistics: N-gram distributions

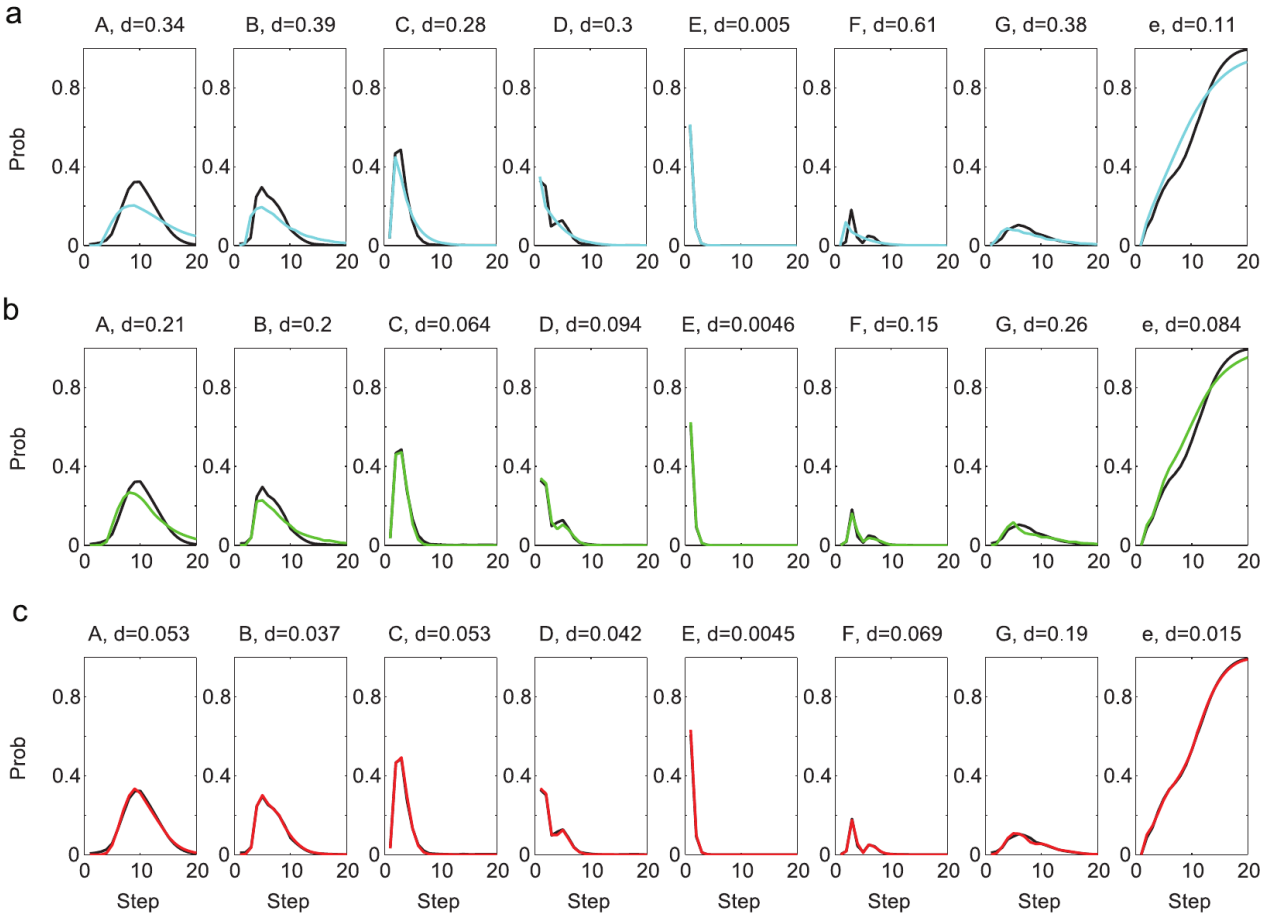


Markov Model

Markov Model with Adaptation

POMM with Adaptation

Testing models by comparing sequence statistics: Step distributions



Markov Model

Markov Model with Adaptation

POMM with Adaptation

Incompatibility of Markov Models

Observed Sequences

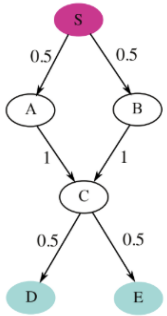
Example 1

seq	P
ACD	0.5
BCE	0.5

Markov Model

generated seqs

seq	P
ACD	0.25
ACE	0.25
BCD	0.25
BCE	0.25



Issue of over-generalization: The model predicts unobserved sequences

Sequence completeness P_c – the total probability of the model generating all unique sequences in the observed set

$$P_c = \sum_{i=1}^{M} P_i$$

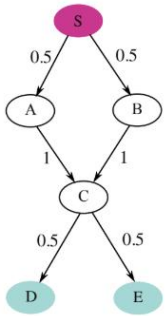
where M is the number of unique sequences

Incompatibility of Markov Models

Observed Sequences

Example 2

seq	P
ACD	0.4
ACE	0.1
BCD	0.1
BCE	0.4



generated seqs

seq	P
ACD	0.25
ACE	0.25
BCD	0.25
BCE	0.25

Issue of probability mismatch: The model predicts wrong probabilities

Total variation distance d – a measure of the difference in probabilities (Gibbs and Su, 2002)

$$d = \frac{1}{2}\sum_{i=1}^{M} \mid P_{i,o} - P_{i,m} \mid$$

where $P_{i,o}$ is the probability of the i^{th} sequence in gthe observed set and $P_{i,m}$ is the normalized probability of the i^{th} sequence computed with the model

Incompatibility of Markov Models

Sequence completeness P_c – the total probability of the model generating all unique sequences in the observed set

$$P_c = \sum_{i=1}^{M} P_i$$

where M is the number of unique sequences

Total variation distance d – a measure of the difference in probabilities (Gibbs and Su, 2002)

$$d = \frac{1}{2} \sum_{i=1}^{M} \mid P_{i,o} - P_{i,m} \mid$$

where $P_{i,o}$ is the probability of the i^{th} sequence in the observed set and $P_{i,m}$ is the normalized probability of the i^{th} sequence computed with the model

Combined

$$P_\beta = (1 - \beta)P_c + \beta(1 - d)$$

where β is a value between 0 and 1

Incompatibility of Markov Models

Observed Sequences

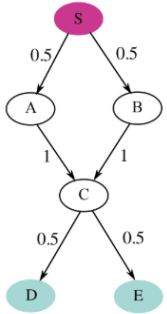
Example 1

seq	P
ACD	0.5
BCE	0.5

Markov Model

generated seqs

seq	P
ACD	0.25
ACE	0.25
BCD	0.25
BCE	0.25

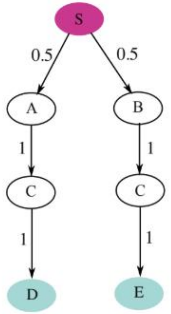


Pc=0.5, d=0, Pβ=0.6

POMM

generated seqs

seq	P
ACD	0.5
BCD	0.5

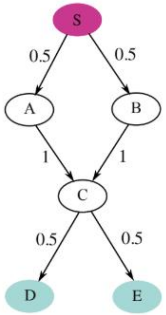


Pc=1, d=0, Pβ=1

Observed Sequences

Example 2

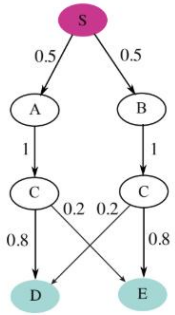
seq	P
ACD	0.4
ACE	0.1
BCD	0.1
BCE	0.4



Pc=1, d=0.3, Pβ=0.94

generated seqs

seq	P
ACD	0.25
ACE	0.25
BCD	0.25
BCE	0.25



Pc=1, d=0, Pβ=1

generated seqs

seq	P
ACD	0.4
ACE	0.1
BCD	0.1
BCE	0.4

Inferring POMMs

To search for a POMM compatible with the observed set, begin by constructing higher-order (order m) Markov models.

Observed Sequence αACDω

↓

Collect unique subsequences of length m, as well as subsequences up to length m that start from α

↓

α, αA, AC, CD, Dω

↓

Each unique subsequence is assigned to a state. The subsequence α is assigned to the start state, subsequences ending with ω are assigned to the end state. The remaining unique subsequences are assigned to distinct states, with the final syllable of each subsequence serving as the symbol for that state.

↓

Transition probabilities between these states are calculated by counting the number of transitions.

(Lu et al, 2025)

Inferring POMMs

The inferred POMM is equivalent to the m^{th} order Markov model.

The POMM is simplified by merging and deleting states associated with the same syllable. If two states are associated with the same syllable, and the probability distributions of subsequent sequences of length 15 or smaller are similar (cosine-similarity > 0.9), the two states are merged.

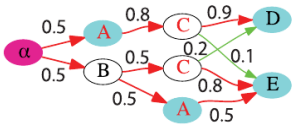
This is done until no further mergers are possible. Finally, state transitions with probabilities smaller than 0.01 are eliminated, and all states that are reached less than 0.005 times in all observed sequences are also eliminated.

The POMM is optimized using Baum-Welch and tested.

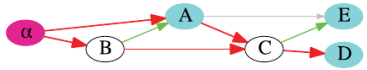
(Jin & Kozhevnikov, 2011)

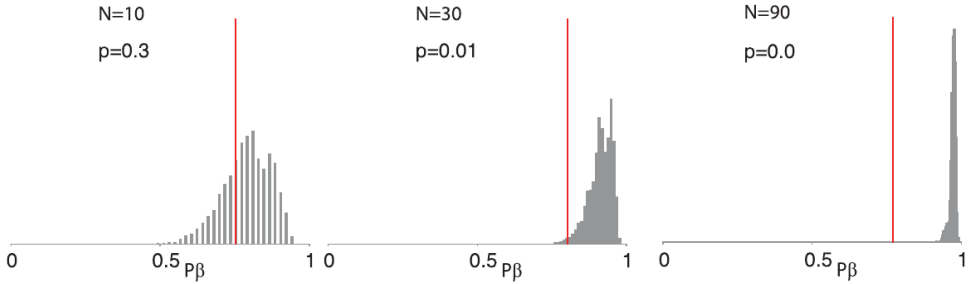
Statistically Testing Inferred Models

Ground Truth Model



Inferred Model





p is the probability that the observed P_β exceeds the P_β values of the generated sets

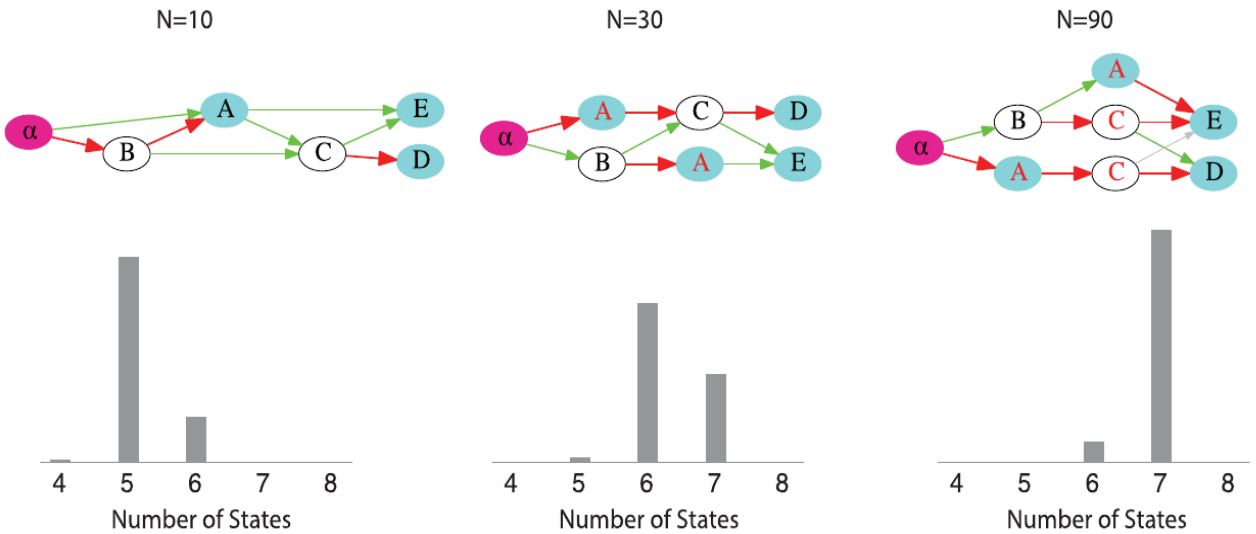
If $p < 0.05$, we infer that the observed P_β is unlikely to have been drawn from this distribution, then leading to the rejection of the inferred model for the observed set.

Conversely, if $p \geq 0.05$, the inferred model is not statistically rejected and is therefore accepted as a model for the observed set.

(Lu et al, 2025)

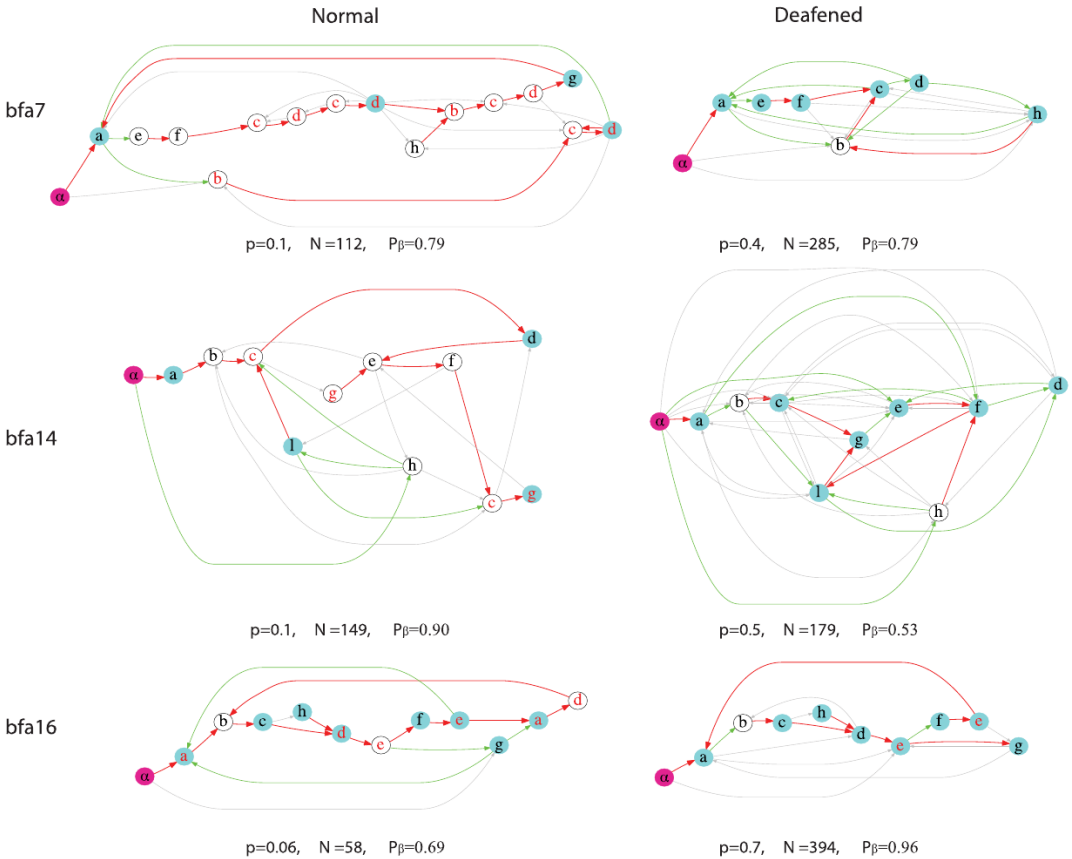
Statistically Testing Inferred Models

Inferred POMMS



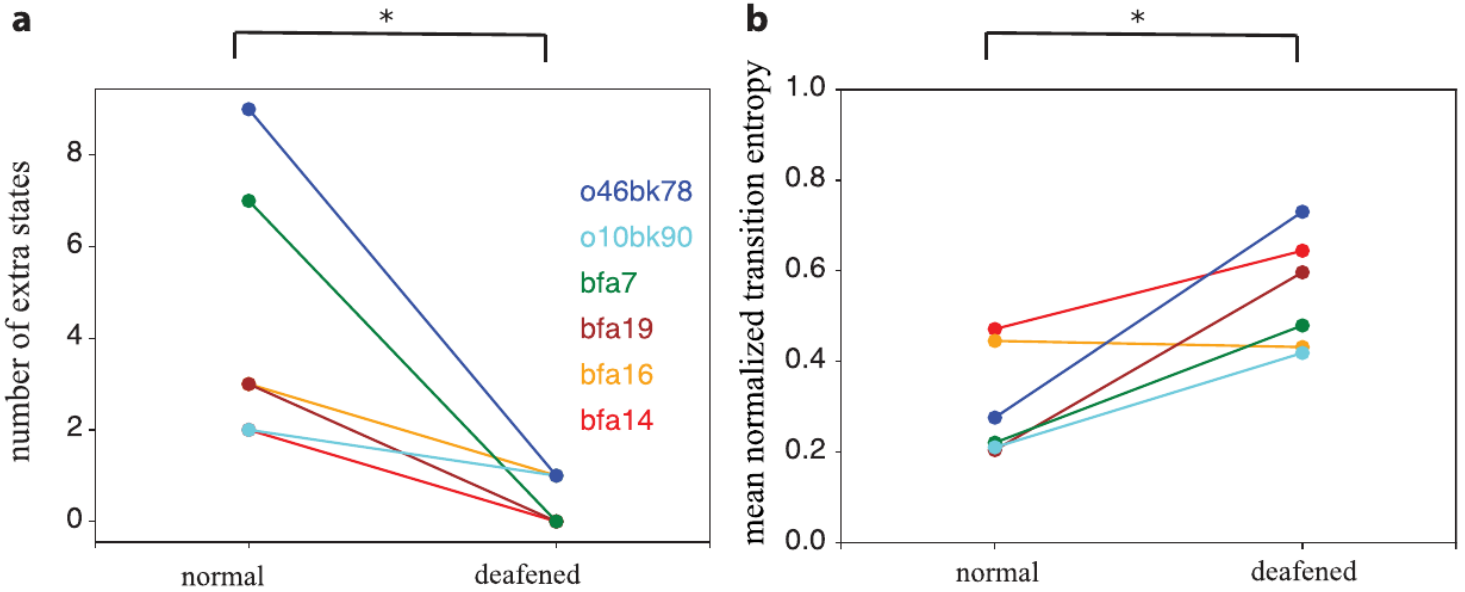
(Lu et al, 2025)

Using models to infer mechanisms



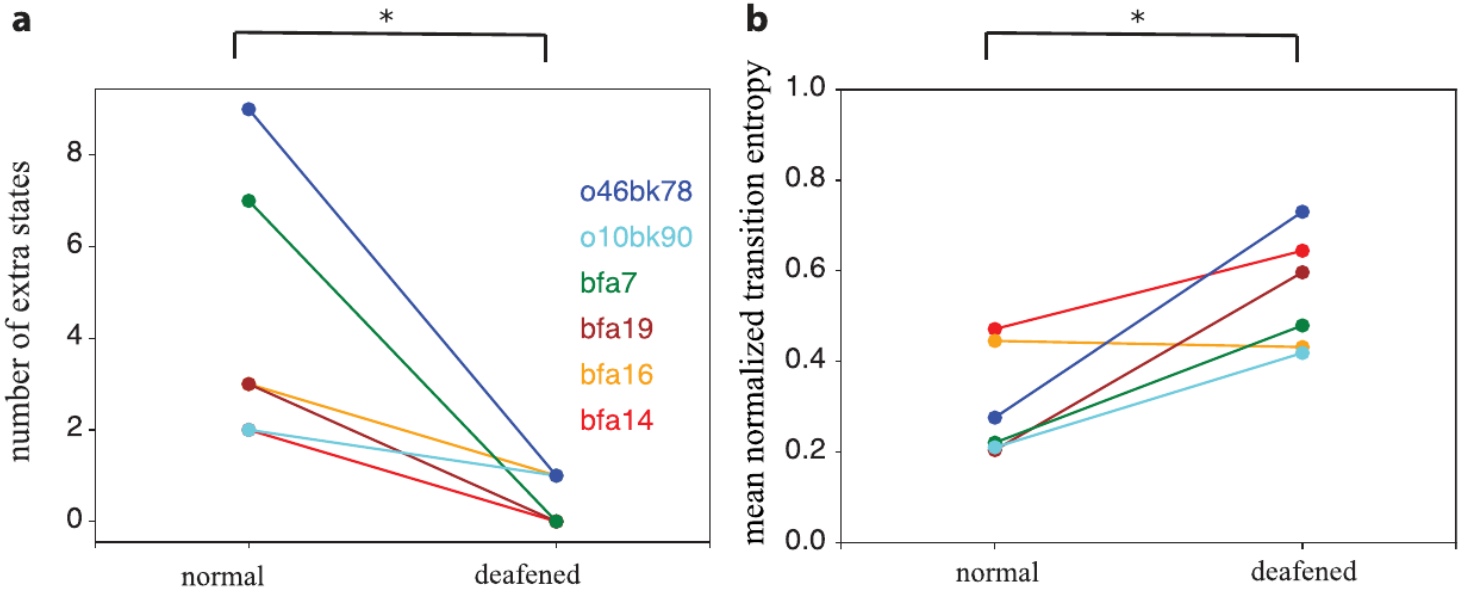
(Lu et al, 2025)

Using models to infer mechanisms



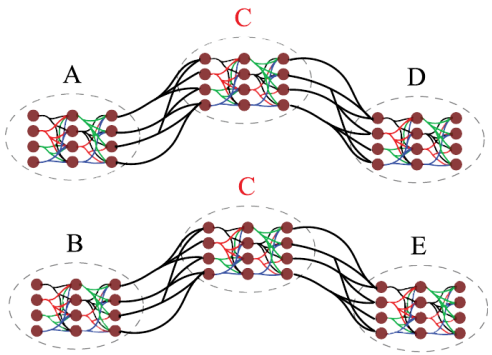
(Lu et al, 2025)

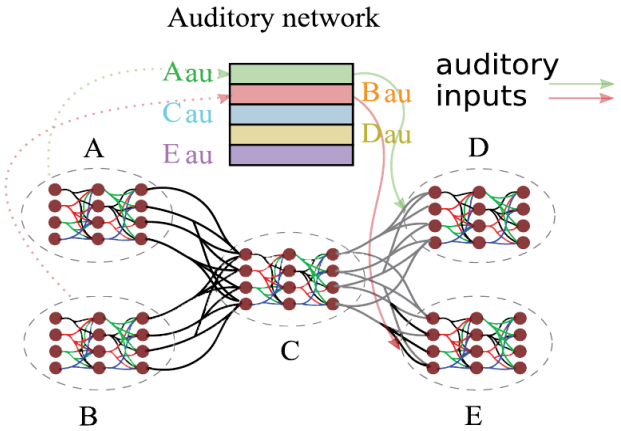
Using models to infer mechanisms

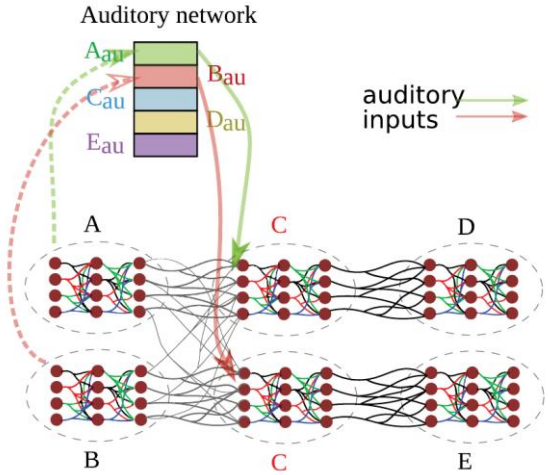


(Lu et al, 2025)

Using models to infer mechanisms







(Lu et al, 2025)

Thank you!

Language of bird (1920)
Nicholas Roerich

sumithra.surendralal@ssla.edu.in

Thank you!

Language of bird (1920)
Nicholas Roerich

sumithra.surendralal@ssla.edu.in