

Hidden Markov Models



BIPN 162

By the end of
today you will
be able to:

- Identify use cases for HMMs in neuroscience (and beyond)
- Represent state transitions in matrix format
- Describe how a mixture model can account for overdispersion
- Explain how Markov chains can be used to predict state transitions
- Describe the process of defining and implementing a HMM

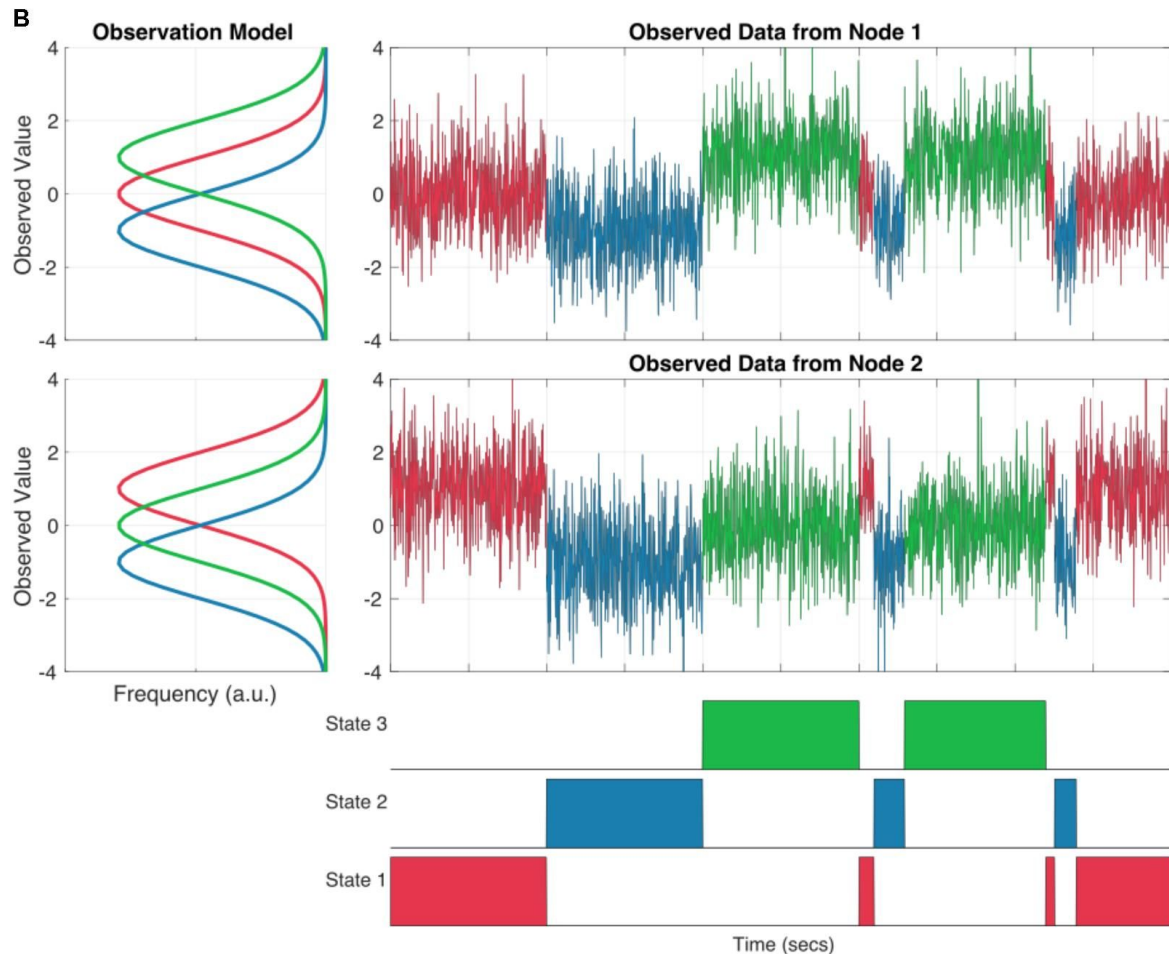
Nervous systems are **stochastic**

Randomly determined; having a random probability distribution or pattern that may be analyzed statistically but *may not be predicted precisely*.
(Oxford Dictionary)

Because of this, we need to take into account **uncertainty**.

In a stochastic environment, we use a **probabilistic framework**.

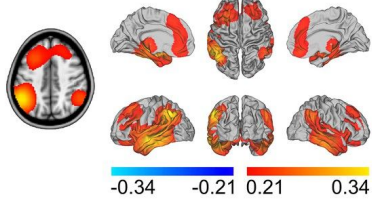
Put into math, we can say there *isn't* a perfect mapping between our observed data (x) and a world property (Θ): $\Theta_1 \rightarrow x_1$
... we need **inference!**



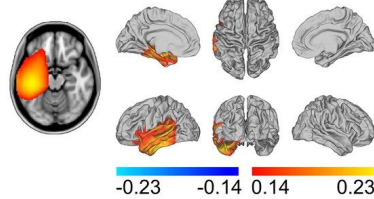
From [Quinn et al. 2018](#),
“Task-Evoked Dynamic
Network Analysis Through
Hidden Markov Modeling”
(which uses an open
dataset!)

HMMs used to determine states during resting in MEG

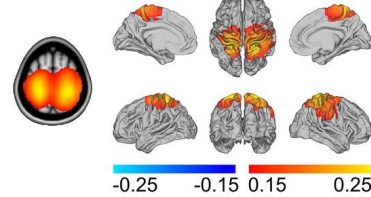
State 1 - Default Mode



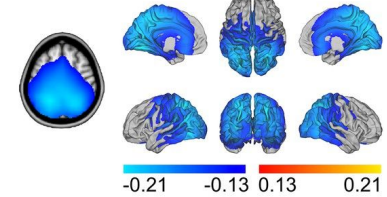
State 5 - L Temporal



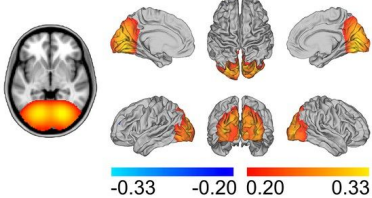
State 3 - Sensorimotor



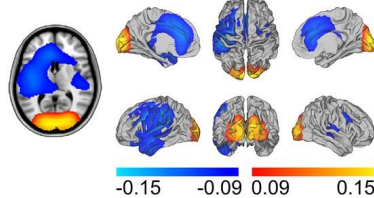
State 7 - Parietal



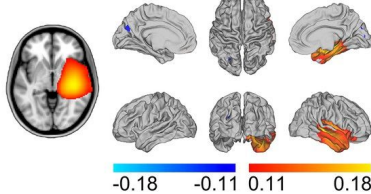
State 2 - Visual



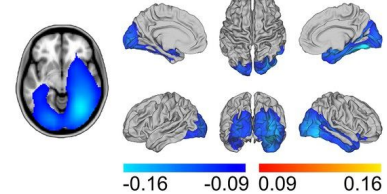
State 6 - Visual



State 4 - R Temporal



State 8 - Visual



GLM-HMM for mouse behavior

Article | Published: 07 February 2022

Mice alternate between discrete strategies during perceptual decision-making

[Zoe C. Ashwood](#) , [Nicholas A. Roy](#), [Iris R. Stone](#), [The International Brain Laboratory](#), [Anne E. Urai](#),
[Anne K. Churchland](#), [Alexandre Pouget](#) & [Jonathan W. Pillow](#) 

[Nature Neuroscience](#) **25**, 201–212 (2022) | [Cite this article](#)

24k Accesses | **45** Citations | **151** Altmetric | [Metrics](#)



Lead author
[Zoë Ashwood](#)

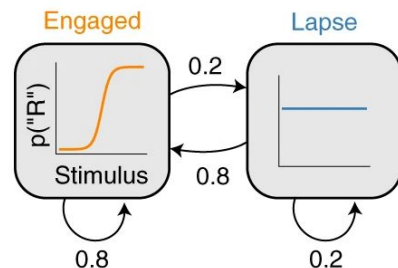
Abstract

Classical models of perceptual decision-making assume that subjects use a single, consistent strategy to form decisions, or that decision-making strategies evolve slowly over time. Here we present new analyses suggesting that this common view is incorrect. We analyzed data from mouse and human decision-making experiments and found that choice behavior relies on an interplay among multiple interleaved strategies. These strategies, characterized by states in a hidden Markov model, persist for tens to hundreds of trials before switching, and often switch multiple times within a session. The identified decision-making strategies were highly consistent across mice and comprised a single ‘engaged’ state, in which decisions relied heavily on the sensory stimulus, and several biased states in which errors frequently occurred. These results provide a powerful alternate explanation for ‘lapses’ often observed in rodent behavioral experiments, and suggest that standard measures of performance mask the presence of major changes in strategy across trials.

<https://www.nature.com/articles/s41593-021-01007-z>

We can also combine GLMs & HMMs

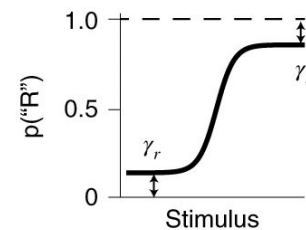
a Classic lapse model



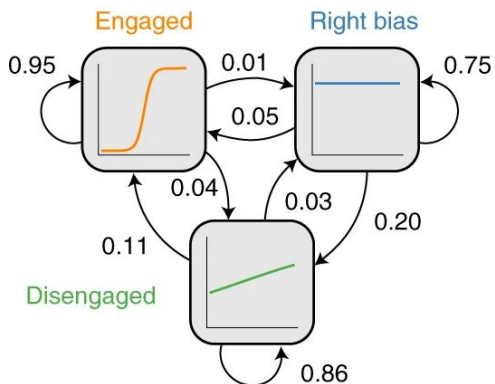
b Example state sequence



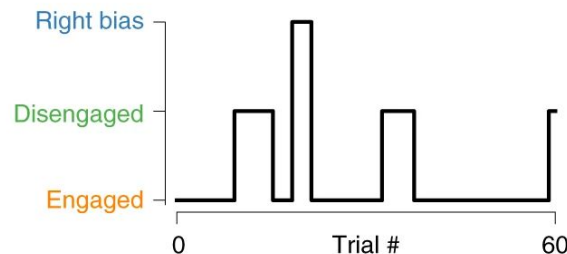
c Psychometric function



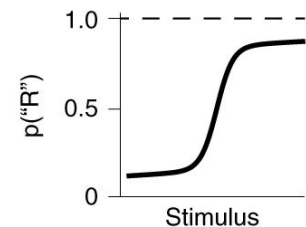
d Three-state GLM-HMM



e



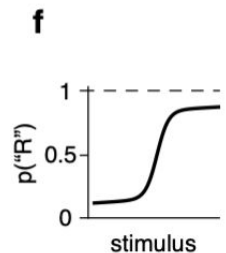
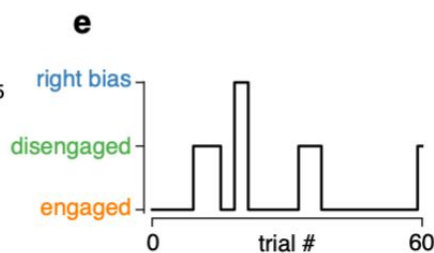
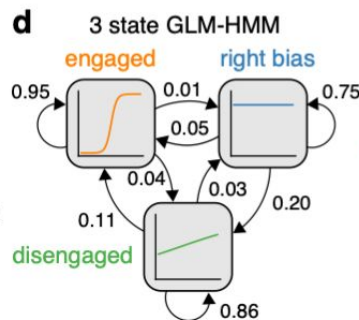
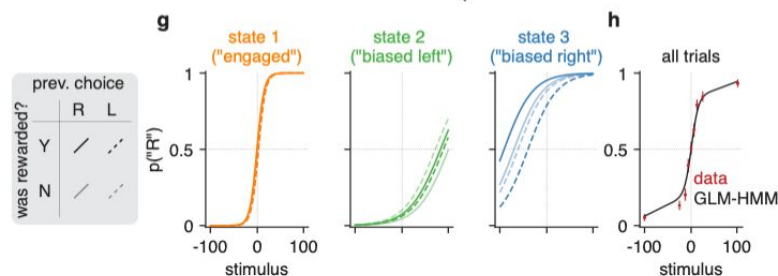
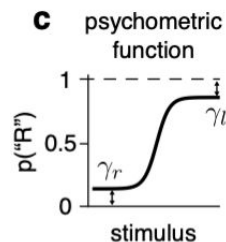
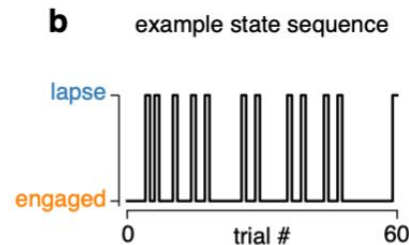
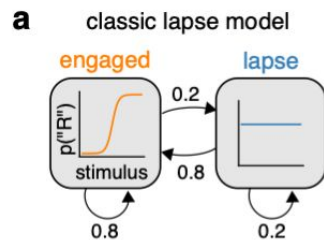
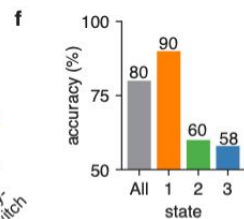
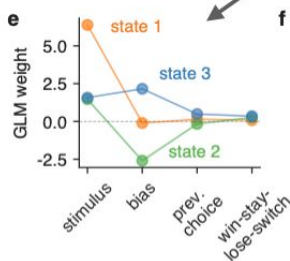
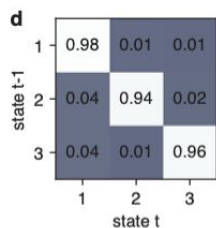
f



GLM-HMM

mouse behavior

GLM weights θ



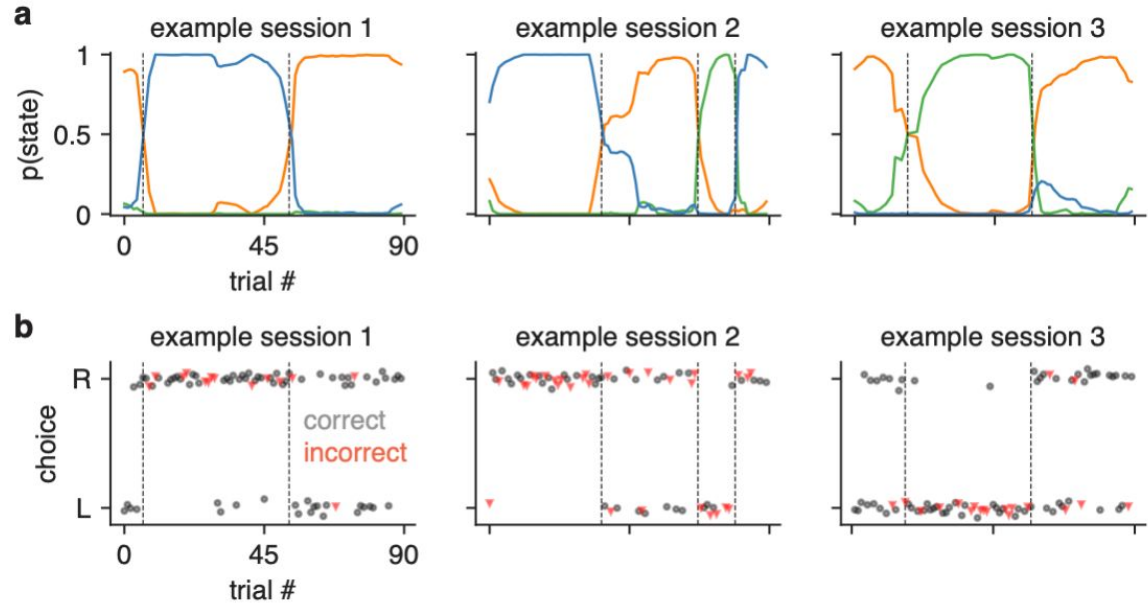
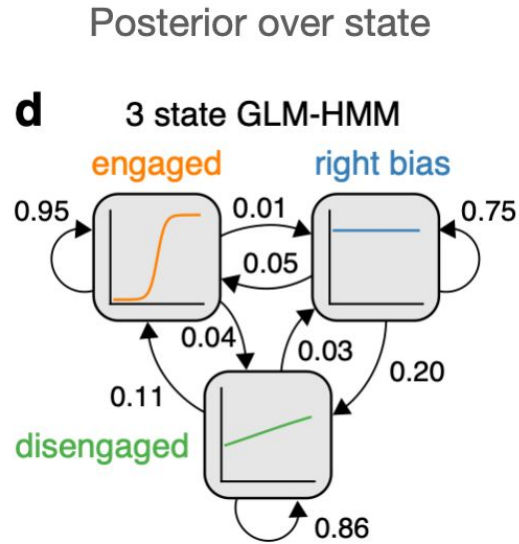
state transitions (dynamic prior)

$$P(S = i | S_{prev} = j)$$

Ashwood et al., *Nature Neuroscience*, 2022

GLM-HMM

mouse behavior



Ashwood et al., *Nature Neuroscience*, 2022

You can do their whole analysis here:

<https://github.com/zashwood/glm-hmm>

By the end of
today you will
be able to:

- Identify use cases for HMMs in neuroscience (and beyond)
- **Represent state transitions in matrix format**
- Describe how a mixture model can account for overdispersion
- Explain how Markov chains can be used to predict state transitions
- Describe the process of defining and implementing a HMM

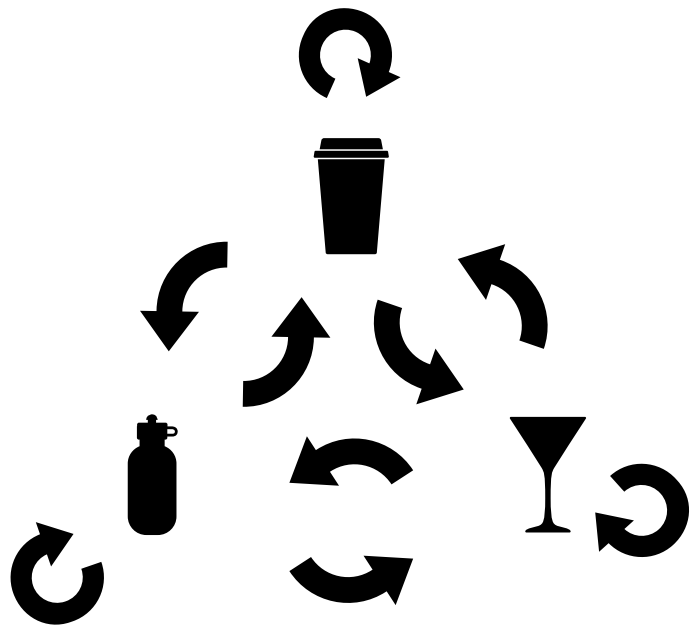
Discrete-state sequences

State transitions



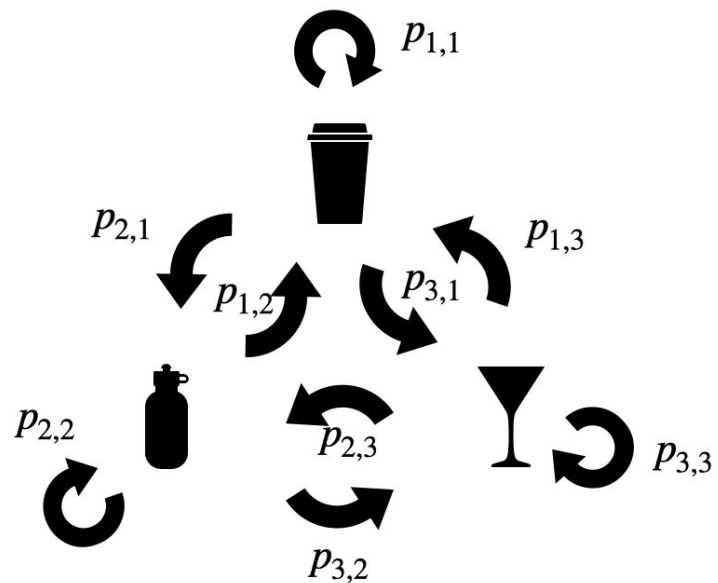
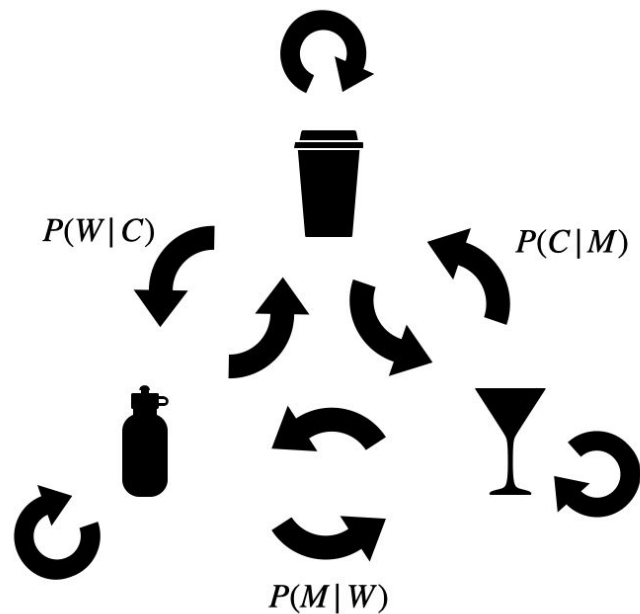
Discrete-state sequences

State transitions



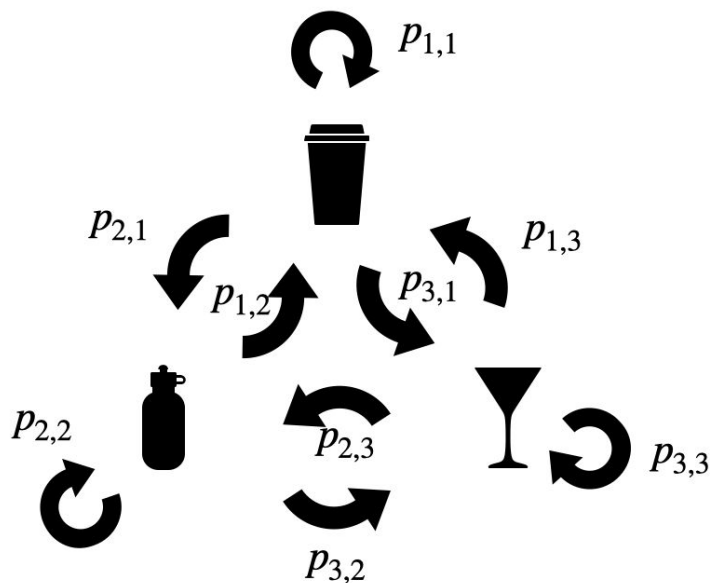
Discrete-state sequences

State transitions



Discrete-state sequences

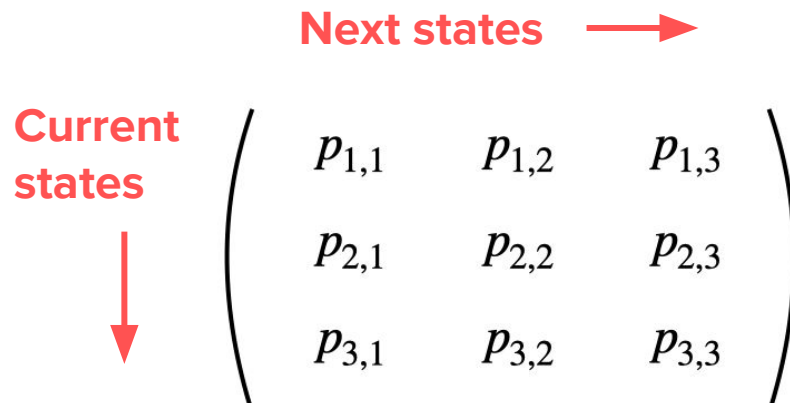
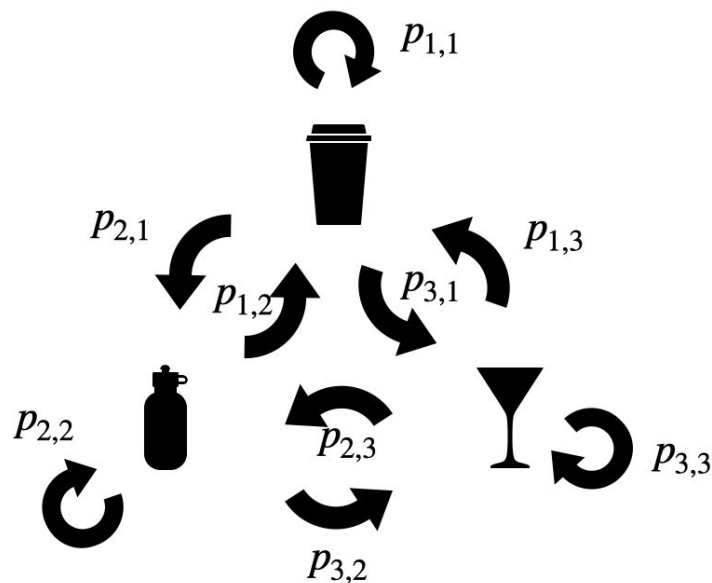
State transitions



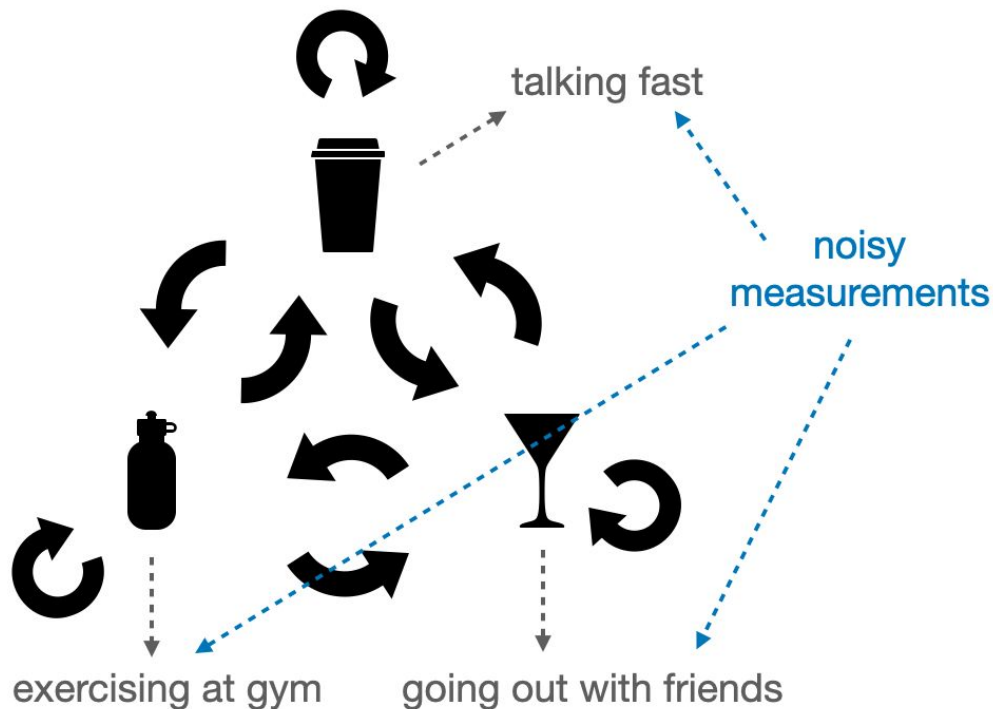
State transition matrix

$$\begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{pmatrix}$$

Markov process (state only depends on previous one)



Hidden Markov Model

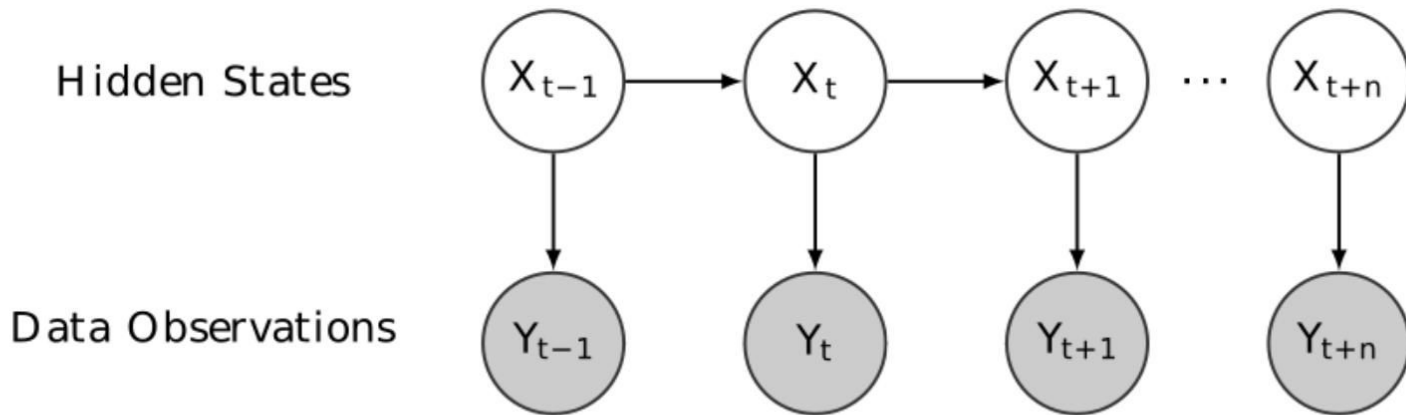


State transition matrix

$$\begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{pmatrix}$$

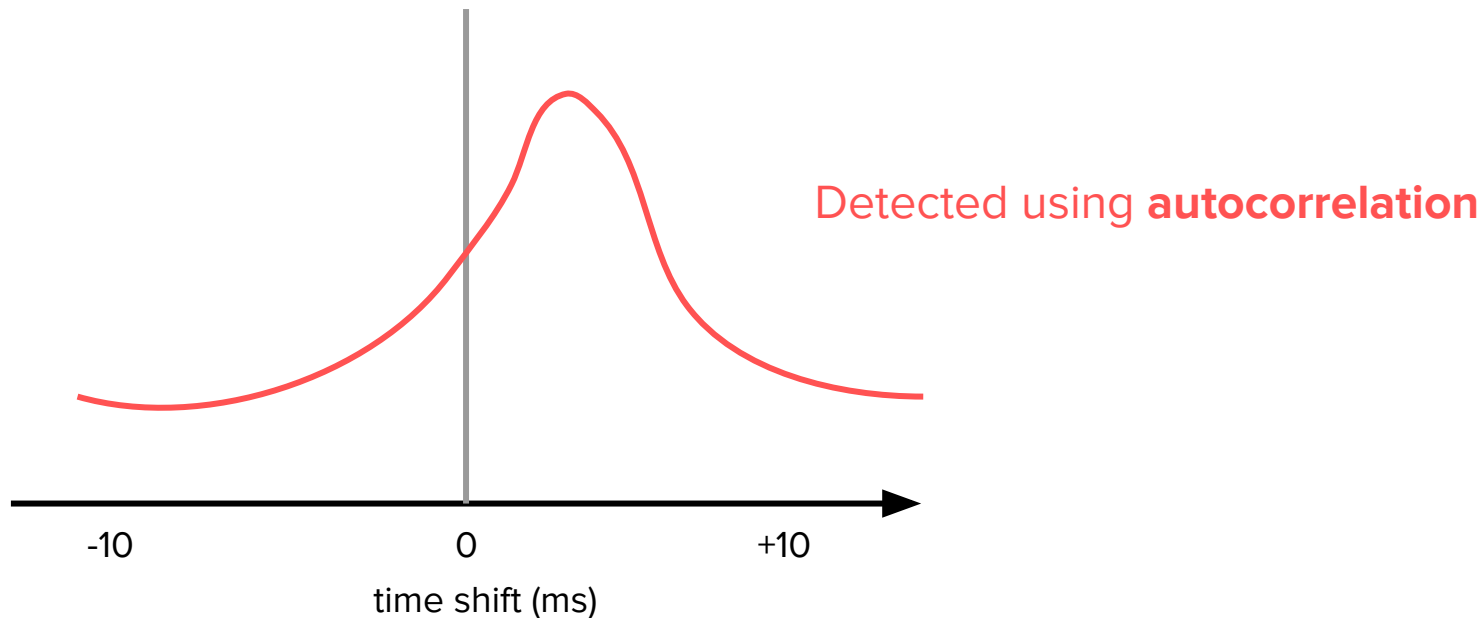
Need to infer transitions from
the measurements

Hidden Markov models (HMMs) are models in which the distribution that generates an observation depends on the state of the underlying and unobserved (i.e., *latent*) **Markov process**



Benefits of using HMMs

- Can be used when there is **serial dependence** (the value of a datapoint at one time is statistically dependent on another datapoint in another time; in other words: time series)



Benefits of using HMMs (*continued*)

- Fairly simple & mathematically tractable
- Likelihood is straightforward to compute
- Can deal with **overdispersion** in distributions
 - the variance of the data is greater than the theoretical variance of the probability distribution being used to describe the data generation process
 - most apparent in count and presence/absence data, often modeled with a **Poisson distribution**
 - In a true Poisson distribution, the variance is equal to the mean

By the end of
today you will
be able to:

- Identify use cases for HMMs in neuroscience (and beyond)
- Represent state transitions in matrix format
- Describe how a mixture model can account for overdispersion
- Explain how Markov chains can be used to predict state transitions
- Describe the process of defining and implementing a HMM

Discrete distributions:

Poisson distribution

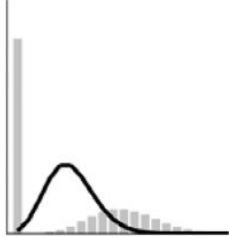
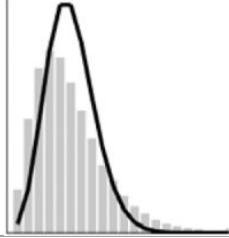
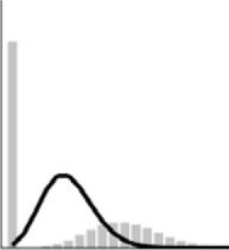
- Gives the probability of an event happening a certain number of times (k) within a given interval of time or space
- Used for count data (e.g., # of cases of cancer; number of spikes)
- Every event is *independent* of the previous one (and should happen at a constant average rate)
- λ is a parameter corresponding to the average outcome of \mathbf{x} (sometimes \mathbf{k})

(In neural modeling, λ is often the overall spike rate)

Example: If you know 750 people, 1% of all people in the population are named Chris, and you are as likely to know Chris' as anyone else, the number of Chris' you know could be modeled as a Poisson distribution with expectation (λ) = 7.5.

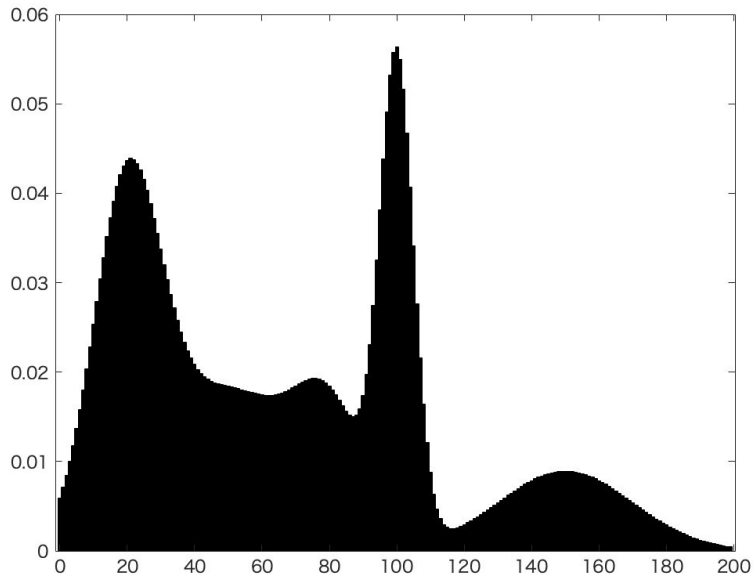
To find the probability of knowing exactly 5 people named Chris out of 750, you would plug in $\lambda = 7.5$ and $\mathbf{x} = 5$ into the formula:

$$P(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Cause of overdispersion	Example	Histograms of overdispersed count data relative to the Poisson
Spatial/temporal clustering	Observations only occur in a small part of the area/time sampled	
Heterogeneity among sampling units	Expected number of observations varies randomly	
Measurement error	Failure to detect samples	

Mixture models can account for overdispersion

- Mixture models can accommodate unobserved heterogeneity
- We can generate counts using *two* different distributions with two different means (λ_1 & λ_2)
- Then, we can determine which mean to use with a **parameter process**
- We can then estimate the best parameters for our mixture distribution using maximum likelihood estimation (MLE) or an expectation-maximization (EM) algorithm



Gaussian mixture for image segmentation ([Wikipedia](#));
see also [Mixture Models book](#)

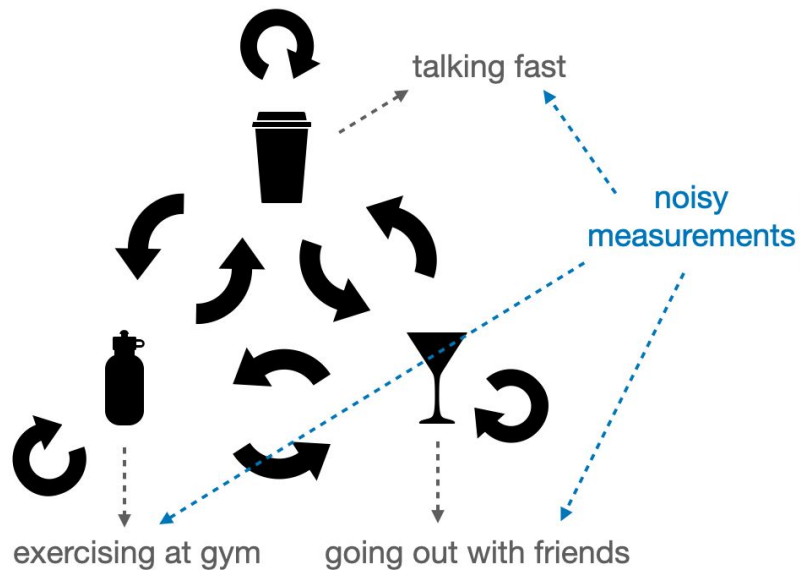
By the end of
today you will
be able to:

- Identify use cases for HMMs in neuroscience (and beyond)
- Represent state transitions in matrix format
- Describe how a mixture model can account for overdispersion
- **Explain how Markov chains can be used to predict state transitions**
- Describe the process of defining and implementing a HMM

Markov chains

- Markov chains have one key property: the present state entirely determines the transition to the next state
 - In other words, history is irrelevant!
 - Transition to the next state is **probabilistic**
- Systems obeying this property can be described as Markovian
- *Examples:*
 - Coffee → Water → Alcohol example (maybe)
 - Opening and closing of ion channels: these are *either* open or closed, and the transition is probabilistic (two states)
- Markov chains have **transitional probabilities**

Transitional probabilities



State transition matrix

$$\begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} \\ p_{2,1} & p_{2,2} & p_{2,3} \\ p_{3,1} & p_{3,2} & p_{3,3} \end{pmatrix}$$

Need to infer transitions from
the measurements

State transition matrix

$$\begin{pmatrix} 0.5 & 0.3 & 0.2 \\ 0.2 & 0.8 & 0.8 \\ 0.7 & 0.8 & 0.9 \end{pmatrix}$$

Example: predicting the weather

	day $t + 1$	
day t	rainy 🌧️	sunny ☀️
rainy 🌧️	0.9	0.1
sunny ☀️	0.6	0.4



$$\mathbf{A} = \begin{pmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{pmatrix}$$

Example: predicting the weather

If today is a sunny day, the distribution of **today's** weather (the **initial state**) is $\pi_0 = P(\text{rainy}), P(\text{sunny}) = (0, 1)$

On the second day, the distribution is $\pi_0 * A = (0.6, 0.4)$

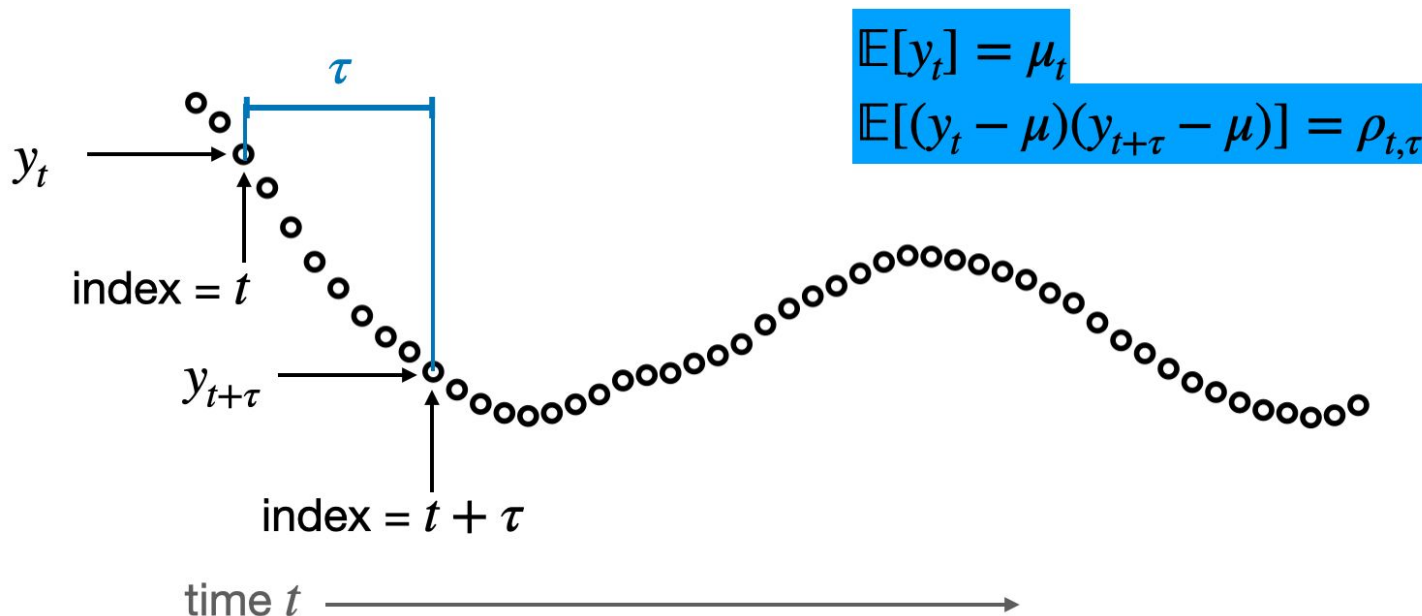
On the third day, the distribution is $\pi_0 * A = (0.78, 0.22)$

$$A = \begin{pmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{pmatrix}$$

Post-multiplication:
dot product of vector &
first column, then second
column

Importantly, these time series have **stationarity**

the statistical properties of a process generating the time series do not change over time; property of a **stochastic process**



By the end of
today you will
be able to:

- Identify use cases for HMMs in neuroscience (and beyond)
- Represent state transitions in matrix format
- Describe how a mixture model can account for overdispersion
- Explain how Markov chains can be used to predict state transitions
- **Describe the process of defining and implementing a HMM**

Step #1 of HMM: Define the model

- **States:** Determine the set of possible hidden states in your system. These states are not directly observable but influence the observed data.
- **Observations:** Define the set of possible observable symbols or values.
- **Initial State Distribution:** Specify the initial probabilities of the system starting in each hidden state.
- **Transition Probabilities:** Define the probabilities of transitioning from one hidden state to another.
- **Emission Probabilities:** Specify the probabilities of emitting a particular observation symbol from each hidden state.
 - For example, when it rains, it's wet outside.

Step #2 of HMM: Train the model

- **Goal:** Estimate the values of the initial state distribution, transition probabilities and emission probabilities that best explain the observed data.
- **Algorithms:**
 - **Baum-Welch Algorithm:** An iterative Expectation-Maximization (EM) algorithm commonly used for training HMMs.
 - **Maximum Likelihood Estimation (MLE):** Another approach to estimate the parameters based on maximizing the likelihood of the observed data given the model.

Step # 3: Decoding

- **Goal:** Given an observation sequence, determine the most likely sequence of hidden states that generated it.
- **Algorithms:**
 - **Viterbi Algorithm:** A dynamic programming algorithm that finds the most likely hidden state path.
 - **Maximum a posteriori (MAP):** bayesian based approach

Step #4: Evaluate

- **Goal:** Evaluate the performance of the trained HMM on new or unseen data.
- **Metrics:**
 - **Forward Algorithm:** Calculates the probability (usually log likelihood) of the observed sequence given the model parameters
 - Implemented in `hmmlearn model.score`
 - Raw values of log likelihood are not particularly useful on their own; they are good for model comparison though!
 - **Backward Algorithm:** Used in conjunction with the forward algorithm to calculate posterior probabilities of hidden states given the observed sequence.

Resources & additional reading

[Tutorial 2: Markov Processes — Neuromatch Academy: Computational Neuroscience](#)

[Tutorial 2: Hidden Markov Model — Neuromatch Academy: Computational Neuroscience](#)

[Mixture models for overdispersed data | Ecological Statistics: Contemporary theory and application | Oxford Academic](#)

Zucchini, MacDonald, & Langrock, *Hidden Markov Models for Time Series*