# Discrete-State Dynamical Systems: a Visit in the Zoo

Herbert Jaeger

JACOBS UNIVERSITY

## Part 1: So Many Views on Dynamics

JACOBS UNIVERSITY

## Literature

If you have about 6000 Euros to spare, the Encyclopedia of Complexity and Systems Science (R. A. Meyers, ed.), Springer Verlag 2008, is the definite compilation of dynamical systems knowledge (~600 detailed articles, > 10K pages, 11 volumes).

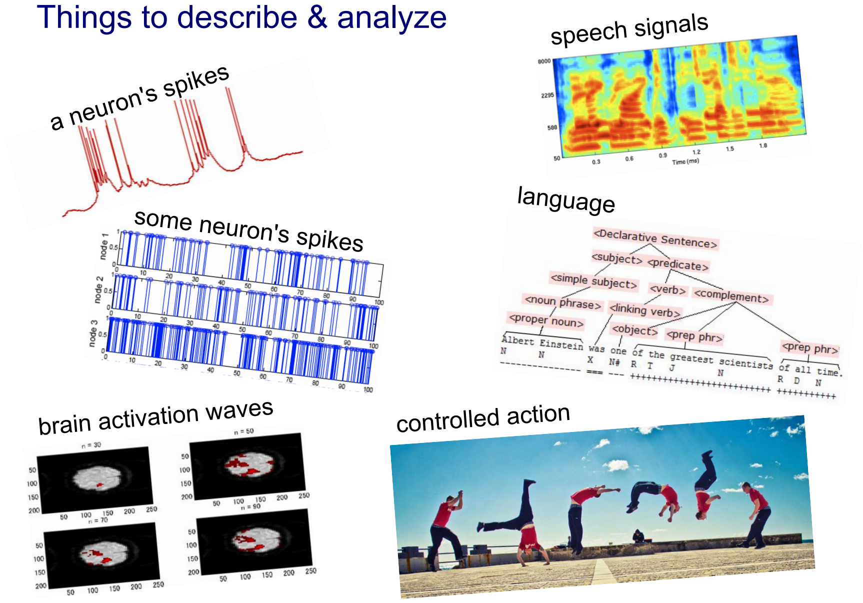
A bit more on the affordable side, I found an easily accessible online tutorial on dynamical systems methods for end-users in biology, akin in spirit to this course (though with a more narrow coverage on "classical" DS topics):

Arnold J. Mandell and Karen A. Selz, AN INTUITIVE GUIDE TO THE IDEAS AND METHODS OF DYNAMICAL SYSTEMS FOR THE LIFE SCIENCES. Online manuscript,
http://cieloinstitute.org/images/
Dynamical_Systems_for_the_Neurobiological_Sciences,_1998.pdf

**Our Mission:** The goal of the Institute is to apply dynamical systems, ergodic-measure theory and statistical mechanics to problems involving brain and behavior. As part of this mission we work to develop novel pharmacological interventions in medical disorders using computational techniques for the analysis, design and physiological testing of polypeptides and proteins, as well as developing dynamical systems approaches to experimental systems in neuroscience and in cell biology.



## Things to describe & analyze




http://kybele.psych.cornell.edu/~edelman/, http://www.ifp.illinois.edu/~rajaram1/poisson.html, http://myguide.bagarinao.com/2011/11/06, http://staffwww.dcs.shef.ac.uk/people/N.Ma/ , http://www.scientificpsychic.com/grammar/enggram1.html, http://www.pxleyes.com/blog/2012/03/

## What is a dynamical system?

A DS is any real or artificial or formal system that evolves over time.

It is almost impossible not to *be* a dynamical system! (because *being* happens in time)



## Examples

- A water molecule, a waterdrop, a river, an ocean
- The Universe
- A calcium channel, a synapse, a dendrite, a neuron, a microcircuit, ... a brain, a nervous system, a body
- Life on earth
- A bitstream, a network of communicating signal sources, a language generating program, a society of linguistic agents
- Mathematics (as a growing body of theorems and proofs)
- You
- What you think about you

*In sum:*

- Everything that is not dead or boring.

There isn't and there can't be a universal theory of dynamical systems. We have to face a diversity of methods.

## Types of dynamical systems and/or modeling methods

Two fundamental decisions before modeling starts:

- **selection**: what subsystem is modeled
- **perspective**: what aspects of that subsystem are modeled

These decisions mandate the use of very different modeling tools.

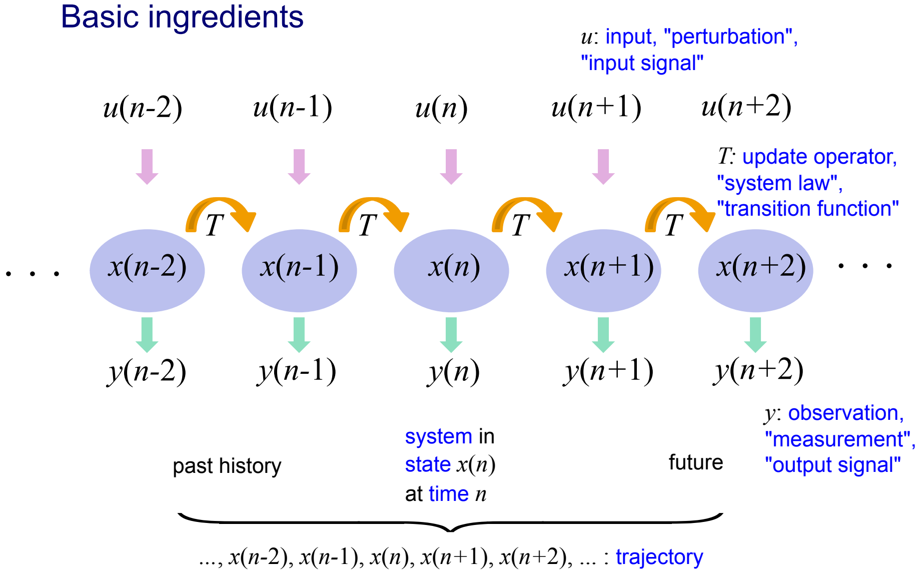
| | | |
|---|---|---|
| **Symbolic**<br>Texts, event and action sequences, DNA, conceptual reasoning | ⟷ | **Numerical**<br>Physiological models, psychometrical measurements, motor control |
| **Deterministic**<br>Electrodynamics, artificial neural networks, mean-field models | ⟷ | **Non-deterministic**<br>Language competence models, grammatical sequence generation |
| | ⟷ | **Stochastic**<br>spike trains, speech, language performance models |
| **Autonomous**<br>Sleep dynamics (?), central pattern generator models (?), circadian clocks (?) | ⟷ | **Non-autonomous**<br>well, ...almost every real-life system |



## ... continued

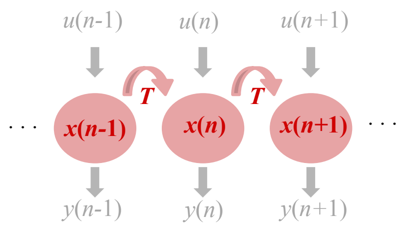
| | | |
|---|---|---|
| **low-dimensional**<br>Hodgkin-Huxley or FitzHugh-Nagumo model of neurons, oscillator models | ⟷ | **high-dimensional**<br>network-level modeling, modeling of cognitive processes |
| **discrete time**<br>state-switching models, models learnt from sampled data | ⟷ | **continuous time**<br>classical neuron models, mean-field models of collective dynamics |
| **linear**<br>"classical" analysis of neural dynamics as signals | ⟷ | **non-linear**<br>neural pattern generators, chaotic dynamics, coupled oscillators |
| **homogeneous**<br>fully or sparsely connected neural network | ⟷ | **non-homogeneous**<br>modular or hierarchical neural circuits and architectures |
| **stationary**<br>neural noise, speech (long timescale), fruit fly in Andrew Straw's virtual arena | ⟷ | **non-stationary**<br>learning processes, speech (short timescale), adaptation processes |
| | ⟷ | **evolutionary**<br>language evolution, ontogenesis, cell differentiation |

## Basic ingredients



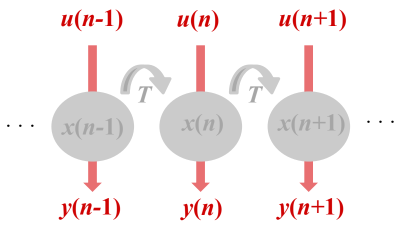



## The natural science view

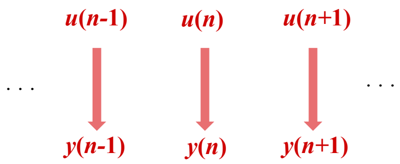



- ever since Newton
- objective: understand "the system"
- focus on modeling the system state $x$ and the system law $T$
- classical formalism: ordinary differential equations (ODEs) for $T$ (continuous time), $x \in \mathbb{R}^n$
- classical approach: isolate system in experimental designs, minimizing role of perturbations $u$, making experiments reproducible
- that is, try to ensure that system can be treated as an autonomous system
- main role of output: measurable, that is, a vehicle to infer back to state $x$

## The engineering view




"state-based representation"




that's what a classically trained engineer calls a "system", or "filter"

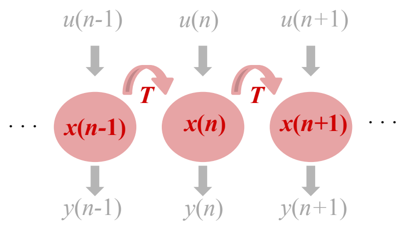
- the classical perspective taken by signal processing and control engineers
- objective: design and build useful input-output devices ("filters", "controllers")
- focus on the mathematical relationships between input signals $u(n)$ and output signals $y(n)$
- engineers love linear input-output relationships: highly developed arsenal of mathematical methods ("frequency domain" modeling)
- "state-based representations" are only one (and not the classical) of the modeling approaches in the signal processing field.

**Literature**

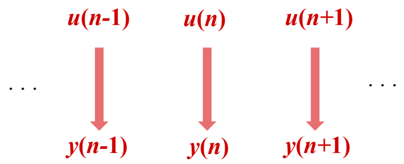
There are many textbooks on signal processing. Free online textbooks that try to be intuitive and do not require much maths are listed at the end of http://en.wikipedia.org/wiki/Signal_processing. A more rigorous yet accessible and comprehensive textbook treatment is provided by the online lecture notes of the MIT open course

Alan Oppenheim, and George Verghese. *6.011 Introduction to Communication, Control, and Signal Processing, Spring 2010*. (Massachusetts Institute of Technology: MIT OpenCourseWare), http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-011-introduction-to-communication-control-and-signal-processing-spring-2010/

## A first comparison




- physicists / neurophysiologists / neurologists / neurolinguists want to model neural architectures / mechanisms
- tools from dynamical systems theory proper, stochastic processes and information theory

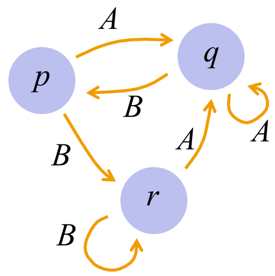



- cognitive scientists / behavioural neuroscientists / linguists / BCI engineers want to model neural function / performance
- tools from signal processing, machine learning, CS, stochastic processes and information theory

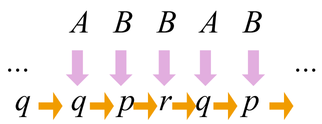
It's good to be familiar with *both*.

## Part 2: A Zoo of Finite-State Models

## Deterministic finite-state automata (DFA)




example trajectory:



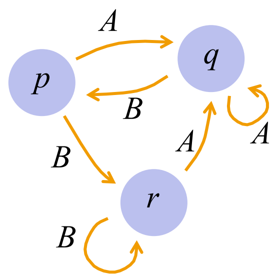

A DFA is defined by:

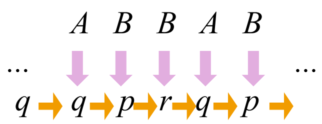
- a finite set of states, e.g. $Q = \{p, q, r\}$
- a finite set of input symbols, e.g. $\Sigma = \{A, B\}$
- a transition function $T: Q \cdot \Sigma \rightarrow Q$, can be written as table, e.g.

| | $A$ | $B$ |
|---|---|---|
| $p$ | $q$ | $r$ |
| $q$ | $q$ | $p$ |
| $r$ | $q$ | $r$ |

- A DFA defines input-sequence dependent state trajectories

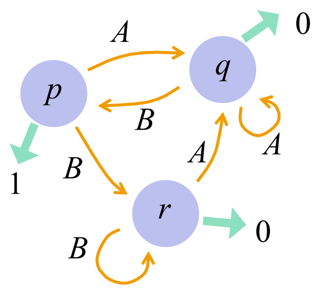
## DFAs, comments



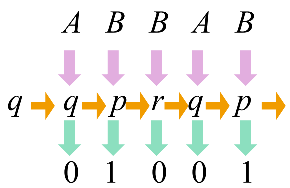




- deterministic
- can be used e.g. for modeling ion channel states or an agent's model of how the world $Q$ reacts on agent's actions $\Sigma$
- simplicity is deceptive: a PC can be considered a DFA, with more-than-astronomical-sized (but finite) $Q = \{$`all possible logic gate state combinations`$\}$
- states are "fully observable"
- inferring a DFA from observed trajectories is easy
- DFAs are standard tool for theoretical CS, then used only for finite sequences ("words")

## Moore- and Mealy-Machines




example Moore trajectory:




A Moore Machine is a DFA, equipped additionally with

- a finite set of output symbols, e.g. $O = \{0, 1\}$
- a translation (observation) function $\rho: Q \rightarrow O$

A Mealy Machine is similar, except the observations are "emitted" from transition arrows: $\rho: Q \cdot \Sigma \rightarrow O$

Both are deterministic.

Efficient methods to infer Moore / Mealy machines from input-output data are known.

JACOBS UNIVERSITY

**Literature**

There are many textbooks covering finite automata – they form a core part of theoretical CS and any theoretical CS textbook will cover them (among other topics). A classic is

Hopcroft, John E. , Motwani, Rajeev, and Ullman, Jeffrey: *Introduction to Automata Theory*, 2nd edition. Addison-Wesley, 2001

Textbook concentrating on finite automata which covers both the use of automata for finite-word languages and for infinite-sequence languages

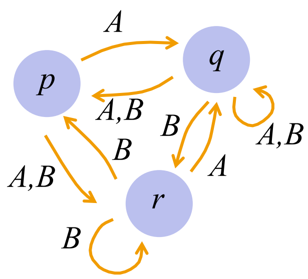
A. de Vries: Finite Automata: Behavior and Synthesis. Elsevier, 2014

State-of-the-art entrance paper for learning Mealy machines from data:

Steffen, B., Howar, F., & Merten, M. (2011). Introduction to active automata learning from a practical perspective. In *Formal Methods for Eternal Networked Software Systems* (pp. 256-296). Springer Berlin Heidelberg. http://ls5-www.cs.tu-dortmund.de/cms/en/research/papers/introduction-to-automata-learning-sfm2011.pdf

JACOBS UNIVERSITY

## Non-deterministic finite-state automata (NFA)




example trajectories:

$A \quad B \quad B \quad A \quad B$

… ↓ ↓ ↓ ↓ ↓ …

$q \rightarrow q \rightarrow p \rightarrow r \rightarrow q \rightarrow p \rightarrow$

or $q \rightarrow p \rightarrow r \rightarrow r \rightarrow q \rightarrow r \rightarrow$

A NFA is defined by:

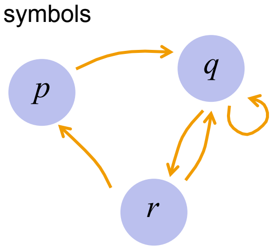
- a finite set of states, e.g. $Q = \{p, q, r\}$
- a finite set of input symbols, e.g. $\Sigma = \{A, B\}$
- a transition function $T: Q \cdot \Sigma \rightarrow \mathrm{Pot}(Q)$, (Pot: power set), e.g.

| | $A$ | $B$ |
|---|---|---|
| $p$ | $\{q, r\}$ | $\{r\}$ |
| $q$ | $\{p, q\}$ | $\{p, q, r\}$ |
| $r$ | $\{q\}$ | $\{p, r\}$ |

JACOBS UNIVERSITY

## NFAs, comments

special case: no input symbols



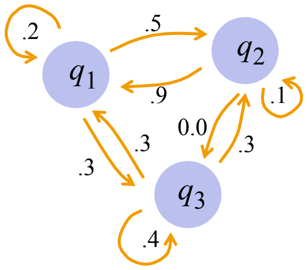

example trajectory:

$q \rightarrow q \rightarrow r \rightarrow p \rightarrow q \rightarrow r \rightarrow$

- an NFA can yield several (typically, infinitely many) different state sequences on given input sequence
- no probabilities involved; a given state sequence cannot be said to be "more probable" than another
- this is called a non-deterministic system, as opposed to "deterministic" and to "stochastic"
- nondeterministic models capture what is possible vs. what is impossible to observe
- special case: no input (or equivalently, one-element input set)

JACOBS UNIVERSITY

## Finite-dimensional Markov chains




$$\boldsymbol{p} = \begin{pmatrix} 0.5 \\ 0.0 \\ 0.5 \end{pmatrix}$$

A (finite-dimensional) Markov chain (MC) is defined by:

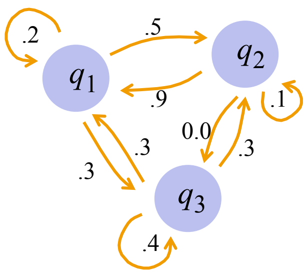
- a finite set of states, e.g. $Q = \{q_1, q_2, q_3\}$
- an initial state distribution $\boldsymbol{p} \in Prob(Q)$, where $Prob(Q)$ is the set of probability distributions over $Q$
- a transition kernel $T$: $Q \rightarrow Prob(Q)$
- $T$ can be written as stochastic transition matrix ("Markov matrix"), e.g.

| | $q_1$ | $q_2$ | $q_3$ |
|---|---|---|---|
| $q_1$ | 0.2 | 0.5 | 0.3 |
| $q_2$ | 0.9 | 0.1 | 0.0 |
| $q_3$ | 0.3 | 0.3 | 0.4 |

rows sum to 1

JACOBS UNIVERSITY

## Finite-dimensional MCs, comments 1




$$\boldsymbol{p} = \begin{pmatrix} 0.5 \\ 0.0 \\ 0.5 \end{pmatrix}$$

- used to describe trajectories that start at time $n = 0$
- probability that a trajectory starts with

  $q_{i0}, q_{i1}, \ldots, q_{in}$

  is

$$P(X_0 = q_{i_0}, \ldots, X_n = q_{i_n}) =$$
$$= P(X_0 = q_{i_0}) \cdot P(X_1 = q_{i_1} \mid X_0 = q_{i_0}) \cdot \ldots \cdot$$
$$\cdot P(X_n = q_{i_n} \mid X_{n-1} = q_{i_{n-1}})$$
$$= P(X_0 = q_{i_0}) \cdot \prod_{k=1}^{n} P(X_k = q_{i_k} \mid X_{k-1} = q_{i_{k-1}})$$
$$= \boldsymbol{p}(i_0)\, T(i_0, i_1) \ldots T(i_{n-1}, i_n)$$

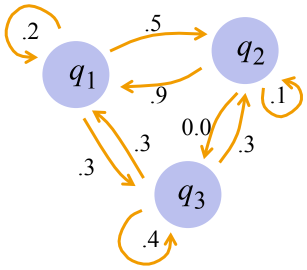
- a MC specifies a stochastic process

$$(X_n)_{n=0,1,2,\ldots}$$

  with values in $Q$

JACOBS UNIVERSITY

## Finite-dimensional MCs, comments 2



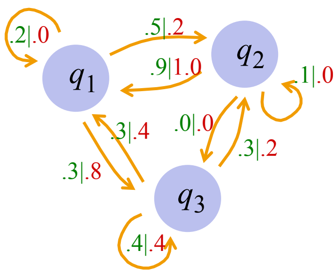

$$\boldsymbol{p} = \begin{pmatrix} 0.5 \\ 0.0 \\ 0.5 \end{pmatrix}$$

- the crucial defining property to make a finite-valued stochastic process a Markov chain: the Markov property

$$P(X_{n+1} = q_{i_{n+1}} \mid X_0 = q_{i_0}, \ldots, X_n = q_{i_n}) = \\ = P(X_{n+1} = q_{i_{n+1}} \mid X_n = q_{i_n})$$

- "what's going to happen next (probabilities to observe $q_{in+1}$) only depends on current state $q_{in}$, not on previous state history"
- MCs are "memoryless" systems

## Controlled Markov chains




$$\boldsymbol{p} = \begin{pmatrix} 0.5 \\ 0.0 \\ 0.5 \end{pmatrix}$$

In a controlled MC the transition probabilities are switched by inputs. Components:

- a finite set of states, e.g. $Q = \{q_1, q_2, q_3\}$
- a finite set of inputs ("control actions"), e.g. $A = \{a_1, a_2\}$
- an initial state distribution $\boldsymbol{p} \in Prob(Q)$
- for each $a \in A$, a transition kernel $T_a: Q \to Prob(Q)$

| $T_{a_1}$ | $q_1$ | $q_2$ | $q_3$ |
|---|---|---|---|
| $q_1$ | 0.2 | 0.5 | 0.3 |
| $q_2$ | 0.9 | 0.1 | 0.0 |
| $q_3$ | 0.3 | 0.3 | 0.4 |

| $T_{a_2}$ | $q_1$ | $q_2$ | $q_3$ |
|---|---|---|---|
| $q_1$ | 0.0 | 0.2 | 0.8 |
| $q_2$ | 1.0 | 0.0 | 0.0 |
| $q_3$ | 0.4 | 0.2 | 0.4 |

- Update mechanism: switch transition kernel according to current input symbol

**Literature**

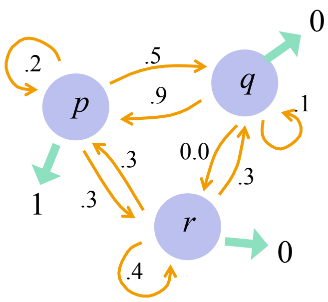
A classical monograph on controlled stochastic processes (general rigorous mathematical theory, not restricted to controlled MCs):

Gihman, I.I. and Skorohod, A.V., Controlled Stochastic Processes. Springer Verlag 1979

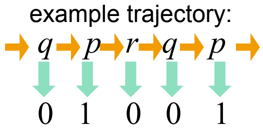
Controlling stochastic systems is, of course, also of prime importance in control engineering. In this field, the controlled systems often are systems that emit observable output, which is then included in the analysis and methods. A textbook:

R. F. Stengel, Stochastic optimal control: theory and application. John Wiley and Sons, 1986

## Hidden Markov models (HMMs)




example trajectory:



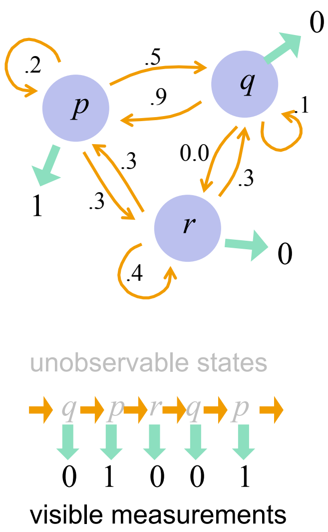

Defining components:

- a finite set of states, e.g. $Q = \{p, q, r\}$
- a finite set of outputs ("observables", "visibles"), e.g. $O = \{0, 1\}$
- an initial state distribution $\boldsymbol{p} \in Prob(Q)$
- a transition kernel $T : Q \rightarrow Prob(Q)$
- for every state $q \in Q$ and observable $o \in O$ , an emission probability $P(o|q)$ to observe $o$ when the hidden Markov state trajectory passes through $q$

  or, equivalently,

  an emission function $\rho : Q \rightarrow O$ (as shown in example)

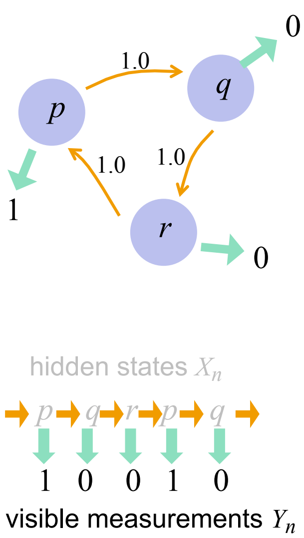
## HMMs, comments 1




- widely and naturally applicable, because an experimenter often can't directly observe states $q$, only make measurements $o$ of them
- available experimental data are only trajectories of observables, states are unobservable
- model inference task: from (empirical) measurement data (e.g., 0 1 0 0 1) infer underlying stochastic state transition system, that is...
- ... explain data by generative mechanism
- Example: $Q$ = "brain states", $O$ = "uttered phonemes"
- Example: $Q$ = "state of bistable neuron", $O$ = "spike"

JACOBS UNIVERSITY

## HMMs, comments 2



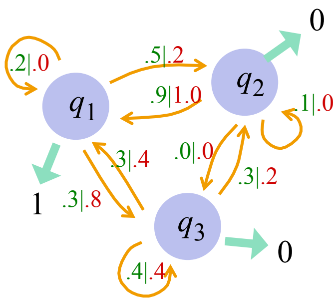

- The visible trajectories (values of random variables $Y_n$) "have memory":

$$P(Y_{n+1} = 1 \mid Y_n = 0) \qquad = 0.5$$
$$P(Y_{n+1} = 1 \mid Y_n = 0, Y_{n-1} = 0) = 1.0$$

- The observables $(Y_n)_{n=0,1,2,...}$ form a stochastic process in their own right, but this process does not have the Markov property

JACOBS UNIVERSITY

## Controlled hidden Markov models, aka POMDPs




- crossover of controlled MCs and HMMs
- also (widely) known as Partially Observable Markov Decision Processes (POMDPs)
- a basic tool in theory of autonomous agents / robotics / reinforcement learning in the machine learning sense
- In that context, a POMDP constitutes the agent's world model:
  - $Q$: external world states
  - $A$: agent's actions in world
  - $O$: sensory feedback from world
- methods available for learning a POMDP from $A$-$O$ (action – sensor-feedback) timeseries data
- seems a natural model class to me also for neural dynamics and animal behavior



**Literature**

My favorite tutorial text on POMDPs, set in a context of agent learning and reinforcement learning:

Kaelbling, L.P., Littman, M.L., Cassandra, A.R., Planning and acting in partially observable stochastic domains. Artificial Intelligence 101 (1998), 99-134