Lecture 2: Neural primitives of thought

Samuel Gershman

Harvard University

Roadmap

- Assembling cognitive algorithms from neural primitives requires us to abstract away from many details.

Roadmap

- Assembling cognitive algorithms from neural primitives requires us to abstract away from many details.
- We'll develop simple formalizations that capture key aspects of this abstraction.

The leaky integrate-and-fire neuron

- The leaky integrate-and-fire (LIF) model formalizes the postsynaptic membrane as a resistor-capacitor circuit that can be charged up by input current (with some leak) and then discharged when a spike occurs.

The leaky integrate-and-fire neuron

- The leaky integrate-and-fire (LIF) model formalizes the postsynaptic membrane as a resistor-capacitor circuit that can be charged up by input current (with some leak) and then discharged when a spike occurs.
- The membrane potential $\mu(t)$ obeys the following dynamics:

$$C\dot{\mu} = \frac{\mu^0 - \mu(t)}{R} + I(t),$$

where $I(t)$ is the input current at time t, $\mu(t)$ is the membrane potential, $\dot{\mu}$ is its temporal derivative, μ^0 is the resting potential, R is the membrane resistance, and C is the membrane capacitance.

The leaky integrate-and-fire neuron

- The leaky integrate-and-fire (LIF) model formalizes the postsynaptic membrane as a resistor-capacitor circuit that can be charged up by input current (with some leak) and then discharged when a spike occurs.
- The membrane potential $\mu(t)$ obeys the following dynamics:

$$C\dot{\mu} = \frac{\mu^0 - \mu(t)}{R} + I(t),$$

  where $I(t)$ is the input current at time t, $\mu(t)$ is the membrane potential, $\dot{\mu}$ is its temporal derivative, μ^0 is the resting potential, R is the membrane resistance, and C is the membrane capacitance.
- Linear integration: $I(t) = \sum_d w_d z_d(t)$, where $z_d(t)$ is the spike train of presynaptic neuron d, and w_d is the synaptic strength.

The leaky integrate-and-fire neuron

- Membrane resistance (R): determined by the number of open ion channels.

The leaky integrate-and-fire neuron

- Membrane resistance (R): determined by the number of open ion channels.
- Membrane capacitance (C): determined by the surface area of the membrane.

The leaky integrate-and-fire neuron

- Membrane resistance (R): determined by the number of open ion channels.
- Membrane capacitance (C): determined by the surface area of the membrane.
- Sometimes we will refer to $\tau = RC$, the membrane time constant (how quickly the membrane responds to a change in input current).

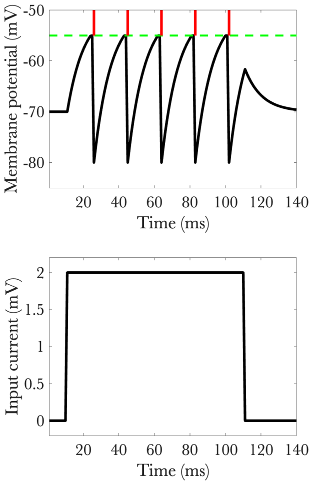
The leaky integrate-and-fire neuron

- When $\mu(t)$ crosses a threshold θ, a spike is emitted and the membrane potential is reset to $\mu^{\text{reset}} < \mu^0$.

The leaky integrate-and-fire neuron

- When $\mu(t)$ crosses a threshold θ, a spike is emitted and the membrane potential is reset to $\mu^{\text{reset}} < \mu^0$.
- Brief post-spike refractory period during which spiking is suppressed.

Leaky integrate-and-fire dynamics with step input



Noise

- Highly regular spiking when cells are recorded in a slice preparation, where the inputs can be precisely controlled [Mainen & Sejnowski, 1995].

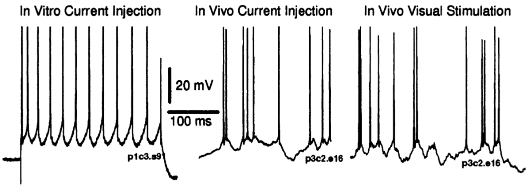
Noise

- Highly regular spiking when cells are recorded in a slice preparation, where the inputs can be precisely controlled [Mainen & Sejnowski, 1995].
- Highly irregular spiking when cells are recorded *in vivo* (i.e., by inserting electrodes into the intact brain), following either current injection or the presentation of a stimulus.

Noise

- Highly regular spiking when cells are recorded in a slice preparation, where the inputs can be precisely controlled [Mainen & Sejnowski, 1995].
- Highly irregular spiking when cells are recorded *in vivo* (i.e., by inserting electrodes into the intact brain), following either current injection or the presentation of a stimulus.
- Hypothesis: irregularity arises from "random" synaptic background activity from other inputs, which is present *in vivo* but absent in the slice preparation.

Voltage traces of neurons recorded in visual cortex



[Holt et al. 1996]

Noisy LIF model

- Adding membrane potential noise $\epsilon(t)$:

$$C\dot{\mu} = \frac{\mu^0 - \mu(t)}{R} + I(t) + \epsilon(t).$$

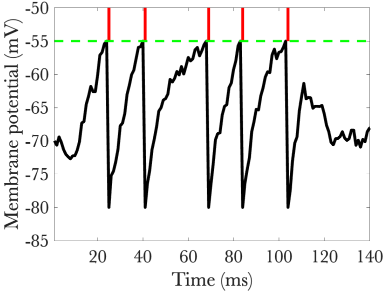
Noisy LIF model

- Adding membrane potential noise $\epsilon(t)$:

$$C\dot{\mu} = \frac{\mu^0 - \mu(t)}{R} + I(t) + \epsilon(t).$$

- If the noise reflects the summation of many independent excitatory and inhibitory currents, then the Central Limit Theorem implies that the noise should be approximately Gaussian: $\epsilon(t) \sim \mathcal{N}(0, \sigma^2)$.

LIF neuron with step input and membrane potential noise



The linear-nonlinear Poisson (LNP) model

- The LNP model is a statistical abstraction which captures important aspects of real neurons despite sacrificing the mechanistic description of spiking generation. It's sometimes easier to work with than the LIF model.

The linear-nonlinear Poisson (LNP) model

- The LNP model is a statistical abstraction which captures important aspects of real neurons despite sacrificing the mechanistic description of spiking generation. It's sometimes easier to work with than the LIF model.
- To derive the model, we first transform the LIF model into a temporal point process, a collection of random variables (spikes in this case) with probabilities specified as a function of time.

The linear-nonlinear Poisson (LNP) model

- The LNP model is a statistical abstraction which captures important aspects of real neurons despite sacrificing the mechanistic description of spiking generation. It's sometimes easier to work with than the LIF model.
- To derive the model, we first transform the LIF model into a temporal point process, a collection of random variables (spikes in this case) with probabilities specified as a function of time.
- We then use the point process to express a static model over an integration window.

The linear-nonlinear Poisson (LNP) model

- The expected firing rate at time t (the intensity function) under the LIF can be approximated by:

$$\rho(t) \propto \exp\left(-\frac{[\bar{\mu}(t) - \theta]^2}{\sigma^2}\right)$$

where $\bar{\mu}(t) = \mathbb{E}[\mu(t)]$ is the expected membrane potential at time, and σ^2 is its steady-state variance.

The linear-nonlinear Poisson (LNP) model

- The expected firing rate at time t (the intensity function) under the LIF can be approximated by:

$$\rho(t) \propto \exp\left(-\frac{[\bar{\mu}(t) - \theta]^2}{\sigma^2}\right)$$

where $\bar{\mu}(t) = \mathbb{E}[\mu(t)]$ is the expected membrane potential at time, and σ^2 is its steady-state variance.

- If we assume that the number of spikes within any interval $[t, t + \Delta]$ is Poisson-distributed with rate $\int_t^{t+\Delta} \rho(t')dt'$, and that the number of spikes is independent across disjoint intervals, then we arrive at the inhomogeneous Poisson process with intensity function $\rho(t)$.

The linear-nonlinear Poisson (LNP) model

- The expected firing rate at time t (the intensity function) under the LIF can be approximated by:

$$\rho(t) \propto \exp\left(-\frac{[\bar{\mu}(t) - \theta]^2}{\sigma^2}\right)$$

  where $\bar{\mu}(t) = \mathbb{E}[\mu(t)]$ is the expected membrane potential at time, and σ^2 is its steady-state variance.
- If we assume that the number of spikes within any interval $[t, t + \Delta]$ is Poisson-distributed with rate $\int_t^{t+\Delta} \rho(t')dt'$, and that the number of spikes is independent across disjoint intervals, then we arrive at the inhomogeneous Poisson process with intensity function $\rho(t)$.
- "Linear" refers to linear synaptic integration. "Nonlinear" refers to the nonlinear mapping from expected membrane potential to the intensity function.

Poisson spiking statistics

Poisson spiking is widely assumed by models. It exhibits two distinctive properties that are similar to real neurons:

Poisson spiking statistics

Poisson spiking is widely assumed by models. It exhibits two distinctive properties that are similar to real neurons:

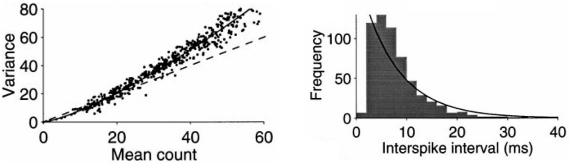
- The ratio between the variance and the mean of the spike count (also known as the *Fano factor*) is close to 1 [Tolhurst et al 1983], although this relationship tends to break down for high spike counts.

Poisson spiking statistics

Poisson spiking is widely assumed by models. It exhibits two distinctive properties that are similar to real neurons:

- The ratio between the variance and the mean of the spike count (also known as the *Fano factor*) is close to 1 [Tolhurst et al 1983], although this relationship tends to break down for high spike counts.
- The interspike interval is approximately exponentially distributed. Note, however, that the refractory period following spikes implies that the very short interspike intervals are not possible; because it is peaked at 0, the exponential distribution always overestimates the frequencies of these short intervals.

Poisson spiking statistics



[Shadlen & Newsome 1998]

Spikes vs. rates

Two theoretical perspectives on the neural code (i.e., how neurons encode information):

- Rate code: precise spike timing is discarded—only firing rates matter.
- Spike timing code: individual spike times matter beyond rates.

Evidence for a spike timing code

- As discussed, spike timing can be highly reproducible under certain conditions—a necessary precondition for using spike timing to communicate information reliably.

Evidence for a spike timing code

- As discussed, spike timing can be highly reproducible under certain conditions—a necessary precondition for using spike timing to communicate information reliably.
- Discarding spike timing information from neural data reduces the performance of an optimal stimulus decoder $p(s|x)$ to levels well below animal behavioral performance, whereas a model that incorporates spike timing is able to match animal performance [Jacobs et a. 2009; Mackevicius et al. 2012; Zuo et al. 2015]

Decoders

- A stimulus decoder is a distribution $p(s|x)$, where s is the stimulus and x is some measure of neural activity. An optimal decoder uses Bayes' rule to obtain $p(s|x)$.

Decoders

- A stimulus decoder is a distribution $p(s|x)$, where s is the stimulus and x is some measure of neural activity. An optimal decoder uses Bayes' rule to obtain $p(s|x)$.
- Showing that we can decode stimulus information from neural activity means that the information must be represented by those neurons.

Decoders

- A stimulus decoder is a distribution $p(s|x)$, where s is the stimulus and x is some measure of neural activity. An optimal decoder uses Bayes' rule to obtain $p(s|x)$.
- Showing that we can decode stimulus information from neural activity means that the information must be represented by those neurons.
- The decoder can also be viewed as a model of neural computation: some downstream neurons can be conceptualized as decoding information from upstream neurons.

Decoders

- If a neuroscientist's decoder matches behavioral performance, it means that it is doing approximately as well as the brain's decoder.

Decoders

- If a neuroscientist's decoder matches behavioral performance, it means that it is doing approximately as well as the brain's decoder.
- This provides us with a recipe for reverse engineering what information is being used by the brain's decoder.

Decoders

- If a neuroscientist's decoder matches behavioral performance, it means that it is doing approximately as well as the brain's decoder.
- This provides us with a recipe for reverse engineering what information is being used by the brain's decoder.
- The studies cited above show that using spike timing yields a better match to behavior than firing rate, suggesting that spike timing is used by the brain's decoder.

Limits of spike timing

- The best data for the behavioral relevance of spike timing comes from early sensory processing, before timing information has been erased over multiple synaptic transmissions.

Limits of spike timing

- The best data for the behavioral relevance of spike timing comes from early sensory processing, before timing information has been erased over multiple synaptic transmissions.
- By the time signals reach higher-level cortex, firing rate may be the only reliable source of information.

Limits of spike timing

- The best data for the behavioral relevance of spike timing comes from early sensory processing, before timing information has been erased over multiple synaptic transmissions.
- By the time signals reach higher-level cortex, firing rate may be the only reliable source of information.
- Away from the sensory periphery, firing rates can be good predictors of behavior, even at the single-trial level.

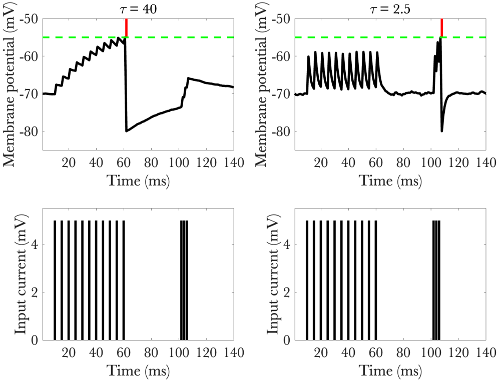
Integration vs. coincidence detection

- Integration when time constant is long; coincidence detection when the time constant is short.

Integration vs. coincidence detection

- Integration when time constant is long; coincidence detection when the time constant is short.
- Integration mode discards spike timing information, whereas coincidence detection mode relies on precisely-timed presynaptic spikes in order to produce a postsynaptic spike (multiple presynaptic spikes need to arrive near-simultaneously in order to push the membrane potential above threshold).

Integration vs. coincidence detection



The fluctuation-driven regime

- When excitation and inhibition of a neuron are approximately balanced (as observed in cortex), the membrane potential approaches a random walk, moving stochastically up or down until the threshold is reached.

The fluctuation-driven regime

- When excitation and inhibition of a neuron are approximately balanced (as observed in cortex), the membrane potential approaches a random walk, moving stochastically up or down until the threshold is reached.
- A short membrane time constant would be disastrous in this regime: spikes would essentially be propagating noise. Need long time constants to average out the noise.

The fluctuation-driven regime

- When excitation and inhibition of a neuron are approximately balanced (as observed in cortex), the membrane potential approaches a random walk, moving stochastically up or down until the threshold is reached.
- A short membrane time constant would be disastrous in this regime: spikes would essentially be propagating noise. Need long time constants to average out the noise.
- This implies that precise spike timing is implausible in the fluctuation-driven regime.

Study question

What are the computational advantages and disadvantages of using spike times vs. firing rates?

Tuning functions

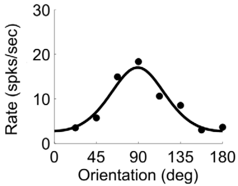
- In many neuroscience experiments, an animal is presented with a stimulus (e.g., an image, sound, etc.) while the firing rates of neurons are measured.

Tuning functions

- In many neuroscience experiments, an animal is presented with a stimulus (e.g., an image, sound, etc.) while the firing rates of neurons are measured.
- This allows the experimenter to plot the average firing rate as a function of some stimulus parameter—a *tuning function* (or *receptive field*).

Tuning functions

- In many neuroscience experiments, an animal is presented with a stimulus (e.g., an image, sound, etc.) while the firing rates of neurons are measured.
- This allows the experimenter to plot the average firing rate as a function of some stimulus parameter—a *tuning function* (or *receptive field*).
- When the stimulus parameter is one-dimensional, this is called a *tuning curve*. Example orientation tuning curve:



Tuning functions

- The stimulus parameter is a state variable (the cause of sensory input). We will use the notation $f_d(s)$ to denote the average firing rate of neuron d in response to state s.

Tuning functions

- The stimulus parameter is a state variable (the cause of sensory input). We will use the notation $f_d(s)$ to denote the average firing rate of neuron d in response to state s.
- The tuning curve is a useful abstraction because it tells us something about how the brain encodes state information.

Tuning functions

- The stimulus parameter is a state variable (the cause of sensory input). We will use the notation $f_d(s)$ to denote the average firing rate of neuron d in response to state s.
- The tuning curve is a useful abstraction because it tells us something about how the brain encodes state information.
- It is not a mechanistic description of the causal events that go from the state to the firing rate of a neuron. One of our goals will be to explain how particular tuning functions arise, both mechanistically and in terms of general design principles.

Tuning functions

- The tuning of single neurons can be highly misleading about the nature of neural computation.

Tuning functions

- The tuning of single neurons can be highly misleading about the nature of neural computation.
- Populations of neurons do much of the computational work in the brain; the relevant information is often distributed in complex ways across many neurons.

Tuning functions

- The tuning of single neurons can be highly misleading about the nature of neural computation.
- Populations of neurons do much of the computational work in the brain; the relevant information is often distributed in complex ways across many neurons.
- In other words, tuning functions are generally meaningful only in the context of the roles they play within a population.

Universality

- By asserting that these simple neuron models function as *neural primitives of thought*, we are making a promise that they can be used to construct computational systems capable of complex cognition.

Universality

- By asserting that these simple neuron models function as *neural primitives of thought*, we are making a promise that they can be used to construct computational systems capable of complex cognition.
- What is the class of computational systems that we can construct with these neural primitives?

Logical universality

- McCulloch & Pitts [1943] famously proposed a model in which spikes signal the truth value of a proposition represented by the neuron.

Logical universality

- McCulloch & Pitts [1943] famously proposed a model in which spikes signal the truth value of a proposition represented by the neuron.
- At each time step, a neuron receives a binary pattern that represents the truth values for a set of input propositions (represented by the presynaptic neurons).

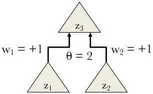
Logical universality

- McCulloch & Pitts [1943] famously proposed a model in which spikes signal the truth value of a proposition represented by the neuron.
- At each time step, a neuron receives a binary pattern that represents the truth values for a set of input propositions (represented by the presynaptic neurons).
- Inputs are linearly weighted by synaptic strengths, followed by a thresholding operation, $z(t) = \phi(I(t) - \theta)$, where θ is a threshold parameter and $\phi(\cdot)$ is a step function.

Logical universality

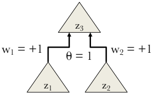
A variety of logical functions can be implemented with different choices of thresholds and weights.

AND



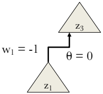
z_1	z_2	z_3
1	1	1
1	0	0
0	1	0
0	0	0

OR



z_1	z_2	z_3
1	1	1
1	0	1
0	1	1
0	0	0

NOT



z_1	z_3
1	0
1	0
0	1
0	1

Logical universality

- More complex functions can be built from these simple elements.

Logical universality

- More complex functions can be built from these simple elements.
- For example, a NAND ("not and") function can be built by composing the AND and NOT functions.

Logical universality

- More complex functions can be built from these simple elements.
- For example, a NAND ("not and") function can be built by composing the AND and NOT functions.
- This construction is significant because the NAND function is a universal element—all other Boolean functions can be constructed out of only NAND functions.

Computational universality

- Logical universality says that we can implement any logical function with a set of primitives, but it does not say that we can implement any computation.

Computational universality

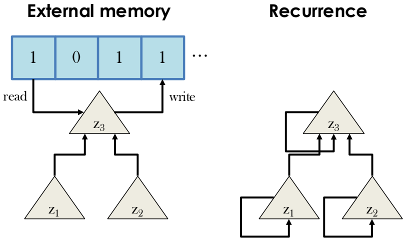
- Logical universality says that we can implement any logical function with a set of primitives, but it does not say that we can implement any computation.
- Consider the following problem: determine whether the first and last inputs in a sequence are the same. What happens when the sequence can be of indeterminate length?

Computational universality

- Logical universality says that we can implement any logical function with a set of primitives, but it does not say that we can implement any computation.
- Consider the following problem: determine whether the first and last inputs in a sequence are the same. What happens when the sequence can be of indeterminate length?
- Here the McCulloch-Pitts neural circuit runs into trouble. It could run out of neurons if the sequence is long enough—a finite network cannot handle arbitrarily long sequences without additional memory.

Computational universality

Two solutions: read-write memory (left) and recurrence (right). McCulloch-Pitts circuits with these augmentations can implement a universal computer.



Universal function approximation

- Instead of binary outputs, we can also build networks that output continuous values. What kinds of functions can these approximate?

Universal function approximation

- Instead of binary outputs, we can also build networks that output continuous values. What kinds of functions can these approximate?
- Under some assumptions, it can be shown that circuits with continuous outputs can approximate any continuous function (technically the input must be from a "compact" subset of $\mathbb{R}^N$).

Study question

Compare the three notions of universality (logical, computational, and function approximation). How do they differ in scope and implications?

Summary

- Networks of LIF neurons furnish our basic set of primitives for computation.

Summary

- Networks of LIF neurons furnish our basic set of primitives for computation.
- These primitives are (in principle) powerful enough to implement any logical function, digital computation, or smooth continuous function.