Gatsby Theoretical Neuroscience Notes

Will Dorrell, based on notes from: Jorge A. Menendez, & Ted Moskovitz

May 2021

Contents

1 Overview

These notes are an unofficial, incomplete, error-littered, guide to the Gatsby Theoretical Neuroscience course. None of this is original, it aims to cover the materials from the TN lectures given by Peter Latham & Maneesh Sahani [Sahani and Latham, 2021], with some added content from the Abbott & Dayan textbook [Dayan and Abbott, 2001], the Gerstner et al. book [Gerstner et al., 2014], and various other sources that help with understanding. The vast majority of the material has been pulled directly from the excellent notes of Ted Moskovitz [Moskovitz, 2020] and Jorge Menendez [Menendez, 2018], with sections added as the course has changed by Will Dorrell and Kira Düsterwald. The hope is that these notes can continue to be updated, and might provide a useful resource while taking the TN course (or in life in general!).

The course has changed over the years so some sections can be skipped if you ~~have no culture~~ just want to pass the exam:

- The plasticity section has been largely dropped from the course, but contains good stuff. In 2021, Peter and Andrew only covered Hebb's rule, BCM and Oja's rule, but none in great detail.
- The reinforcement learning section covers the basic application of RL ideas to biology, something not covered in the DeepMind RL course (though mentioned in [Sutton and Barto, 2018]). This is no longer part of the TN course.

1.1 Sections yet to be Written

- Bag of synapses model, from the Fusi & Abbott paper
- Line Attractors
- RNNs & BPTT
- Information theory: show proofs of the three easy statements about conditional entropy
- Information theory: add histogram equalisation and the section on retinal decoding, as well as the whole section on Gaussian channels up to retinal prediction.
- Neural encoding: that bit about one of Maneesh's papers where you get spurious receptive fields, and how you have to interpret multiple fits well
- Pop coding: add Bayesian decoding
- Uncertainty: needs a full re-write and a lot of additional content (Maneesh added in 2021).
- Dynamical systems in population codes (Maneesh added in 2022).
- Appendix, add the Lagrange method
- Whole of Andrew's section.

And anything else you think is missing! (Or just any improvements, typos, mismatches due to merging notes etc. that you want to correct – go for it!)

2 Biophysics

Here we seek to understand the behaviour of neurons using the physical laws directing ion flows. Much is known about this, and we spend the first few sections building up from ions and ion channels to the Hodgkin-Huxley equations which explains why neurons spike, with a brief diversion into the integrate and fire neuron. We then learn about dendrites and axons, before examining models of short-term synaptic plasticity.

2.1 Single-Compartment Models (Soma)

Setting the Scene Most of the time, there is an excess of negative charge in the interior of a neuron, which, because negative ions repel each other, builds up on the inside surface of the membrane. This in turn causes positive ions to accumulate on the outside surface of the membrane, which acts like a capacitor. The lipid-bilayer membrane generally has pretty high resistance, and would be essentially impermeable, except for the fact that it contains passive and active channels to facilitate movement of ions across it. The effective resistance of the membrane depends on the type and density of these ion channels, most of which are highly selective, only permitting a single type of ion to pass through them.

By convention, we define the extracellular fluid around the neuron to have a potential of 0. Under normal conditions, the internal membrane potential can vary from -90 to $+50$ mV, depending on the opening and closing of ion channels.

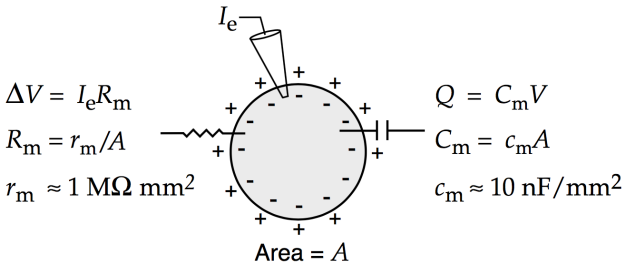


Figure 2.1: Basic set-up of a single-compartment neuron model.We treat the neuron as a circuit (see appendix B).

Membrane Capacitance and Resistance Intracellular resistance to current flow can cause significant differences in membrane potential in a neuron (especially those with long dendrites and/or axons), but for more compact neurons, we can approximate and say that the whole thing has a single membrane potential. This is called *electrotonic compactness*.

As mentioned above, an excess of negative charge Q typically builds up on the interior surface of the membrane, which can be computed via

$$Q = C_m V, \tag{1}$$

where C_m is the membrane capacitance and V is the voltage across the membrane. The membrane capacitance is proportional to the total area of the membrane A, so we can denote the *specific capacitance* by c_m, with

$$C_m = c_m A. \tag{2}$$

Similarly, the total membrane resistance R_m is inversely proportional to the area, so we have the *specific resistance* as

$$R_m = \frac{r_m}{A}. \tag{3}$$

Differentiating eq. 1 with respect to time gives the current required to change the membrane potential at a given rate:

$$C_m \frac{dV}{dt} = \frac{dQ}{dt} = I. \tag{4}$$

In other words, the rate of change of the membrane potential is proportional to the rate at which charge builds up inside the cell. Holding the membrane potential steady at a different voltage from its resting value also requires current, the amount of which is determined by Ohm's law:

$$\Delta V = I_e R_m, \tag{5}$$

where R_m is assumed to be constant over a range of ΔV. These relationships, along with example numbers, are summarized in Figure 2.1. The rate of change of the membrane potential is also governed by the *membrane time constant* τ_m, which is invariant to the surface area of the neuron:

$$\tau_m = R_m C_m = \left(\frac{r_m}{A}\right)(c_m A) = r_m c_m. \tag{6}$$

The value of the membrane time constant typically falls in the range of 10 and 100 ms.

Equilibrium and Reversal Potentials The voltage difference between the interior and exterior of the cell results in electrical forces that facilitate a diffusion of ions across the membrane. These forces are small enough that ion flow is affected both by diffusion and electrical forces (i.e. thermal energy $\approx$ potential energy, see D&A p. 155). Any model that describes the membrane potential of a neuron by a single quantity V is called a *single-compartment model*. When the membrane potential is negative, this drives positive ions into the cell and drives out negative ions.

Neurons actively maintain a concentration gradient with respect to the extracellular space, generating diffusion of particular ions. Two crucial ones are potassium (K^+) and sodium (Na^+) ions, which are respectively pumped in and out of the cell via the active sodium-pottasium pump on the membrane to maintain a relatively higher/lower concentration of K^+/Na^+ inside than outside the cell, by pumping out 3 Na^+ ions for every 2 K^+ pumped in. We define the *equilibrium potential* as the membrane potential at which flow of ions due to electrical forces is exactly canceled by the diffusion of ions due to concentration gradients. For channels that only admit one type of ions, this value is determined by the Nernst equation (see Dayan & Abbott p. 159), these can be seen in the table for various ions:

Ion	Concentration Gradient	Reversal Potential E_i
K^+	higher inside	~ -75mV
Cl^-	higher outside	~ -65mV
Na^+	higher outside	~ 50mV
Ca^{2+}	higher outside	~ 150mV

When a channel admits more than one type of ion, the equilibrium potential is usually a weighted averaged of the selected ions and is known as the *reversal potential*, denoted by $\mathcal{E}$. It's called the reversal potential because the flow of current through the channel switches direction when the membrane potential passes through $\mathcal{E}$. When $V > \mathcal{E}$, positive current flows out, bringing V back to $\mathcal{E}$, and when $V < \mathcal{E}$, there is an inflow of positive current. Therefore, because Na^+ and Ca^{2+} channels have positive reversal potentials, they tend to *depolarize* a neuron – make the membrane potential more positive (as the potential is drawn toward $\mathcal{E}$). Similarly, K^+ channels tend to *hyperpolarize* a neuron – push the membrane potential to be more negative – due to their negative reversal potentials. The reversal potential of Cl^- channels is around equilibrium for many neurons, so they doesn't really affect current flow, they just change the effective resistance of the cell – this is called *shunting*. Synapses with reversal potentials below the threshold needed for action potential generation are typically called *inhibitory*, while those with reversal potentials above the action potential threshold are typically known as *excitatory*.

It's also useful to consider what happens when, for example, the concentration of ions shifts either intracellularly or extracellularly. For instance, if the intracellular concentration of a negative ion such as Cl^- increases, there will be reduced diffusive force inwards, hence less electric force outwards will be needed to balance the flow. Therefore, the reversal/equilibrium potential must *increase*—becoming more positive will repel the negative ions less. This effect is seen in animals: intracellular Cl^- concentrations decrease during development so the reversal potential begins more positive but drops as we get older. This means Cl^- begins life as a depolarising ion, and causes developing nervous systems to be heavily driven by excitation, before becoming more inhibitory later in life.

2.2 Membrane Current and Passive Channels

The membrane current is the total current flowing through the ion channels of the membrane. By convention, it's defined to be positive when positive ions are leaving the cell, and negative when positive ions enter the cell. The total membrane current I_m is given by the product of the surface area A and the membrane current per unit area i_m:

$$i_m = \frac{I_m}{A}. \tag{7}$$

For many types of channels, the membrane current is approximately proportional to the difference between the current voltage V and the membrane potential. Ohm's law gives us

$$i_m = \sum_x \frac{1}{r_x} \Delta V_x = \sum_k g_x (V - \mathcal{E}_x), \tag{8}$$

where $g_x = 1/r_x$ is the channel conductance, and x is an index over channel types. In this section, we assume that the conductances g_x are constant, and thus the current flow is limited to *leakage* current, which includes the currents carried by ion pumps that are involved in maintaining concentration gradients at equilibrium. The

ions that we'll consider to be most involved in this process are Na^+, K^+, and Cl^-. We can expand out eq. 8 as

$$
\begin{aligned}
i_m &= \sum_x \frac{1}{r_x}\Delta V_x = \sum_k g_x(V - \mathcal{E}_x) && (9)\\
&= g_{\mathrm{Na}^+}(V - \mathcal{E}_{\mathrm{Na}^+}) + g_{\mathrm{K}^+}(V - \mathcal{E}_{\mathrm{K}^+}) + g_{\mathrm{Cl}^-}(V - \mathcal{E}_{\mathrm{Cl}^-}) && (10)\\
&= (g_{\mathrm{Na}^+} + g_{\mathrm{K}^+} + g_{\mathrm{Cl}^-})\left(V - \frac{g_{\mathrm{Na}^+}\mathcal{E}_{\mathrm{Na}^+} + g_{\mathrm{K}^+}\mathcal{E}_{\mathrm{K}^+} + g_{\mathrm{Cl}^-}\mathcal{E}_{\mathrm{Cl}^-}}{g_{\mathrm{Na}^+} + g_{\mathrm{K}^+} + g_{\mathrm{Cl}^-}}\right) && (11)\\
&= g_\ell(V - \mathcal{E}_\ell), && (12)
\end{aligned}
$$

where $g_\ell \coloneqq g_{\mathrm{Na}^+} + g_{\mathrm{K}^+} + g_{\mathrm{Cl}^-}$ is the leakage conductance. By convention, external current I_e entering the cell is considered positive, while membrane current leaving the cell is considered negative. From eq. 4, we can then write the dynamics of a passive channel as

$$
\begin{aligned}
c_m\frac{dV}{dt} &= -i_m + \frac{I_e}{A}\\
&= -g_\ell(V - \mathcal{E}_\ell) + \frac{I_e}{A},
\end{aligned}
\tag{13}
$$

where I_e is divided by A because we are considering the current flow per unit area.

2.3 Passive Integrate-and-Fire Models

Integrate-and-fire models stipulate that a neuron will usually fire an action potential when its membrane potential reaches a threshold value V_{th} of around -55 to -50 mV. It then rapidly depolarizes before return to a reset value V_{reset}. Ignoring the role of active conductances and relying solely on the leakage in action potential analyses results in the *passive integrate-and-fire model*. The model behaves like an electric circuit with a resistor and capacitor in parallel, the behavior of which is described by eq. 13. If we multiply both sides by $r_m = 1/g_\ell$, we get

$$
\begin{aligned}
r_m c_m\frac{dV}{dt} &= -(V - \mathcal{E}_\ell) + r_m\frac{I_e}{A}\\
\Rightarrow \tau_m\frac{dV}{dt} &= -(V - \mathcal{E}_\ell) + R_m I_e.
\end{aligned}
\tag{14}
$$

We can see that when $I_e = 0$, the neuron will relax exponentially with time constant τ_m to $\mathcal{E}_\ell$, its resting

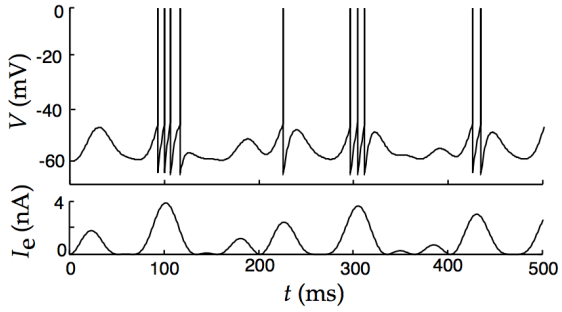


Figure 2.2: Leaky integrate-and-fire model with a time-varying input current.

potential. In other words, $\mathcal{E}_\ell = V_{reset}$. We can solve for the subthreshold potential $V(t)$:

$$
\begin{aligned}
\tau_m\frac{dV}{dt} &= -(V(t) - \mathcal{E}_\ell) + R_m I_e\\
\Rightarrow \tau_m\frac{dU}{dt} &= -U(t) \quad \text{(change of vars: } U = V - \mathcal{E}_\ell - R_m I_e)\\
\Rightarrow \int_0^t \frac{dU}{U} &= \int_0^t -\frac{1}{\tau_m}dt'\\
\Rightarrow \log\left(\frac{U(t)}{U(0)}\right) &= -\frac{t}{\tau_m}\\
\Rightarrow U(t) &= U(0)e^{-t/\tau_m}\\
\Rightarrow V(t) &= \mathcal{E}_\ell + R_m I_e + (V(0) - \mathcal{E}_\ell - R_m I_e)e^{-t/\tau_m}.
\end{aligned}
\tag{15}
$$

Note that this expression holds only for $V(t) < V_{th}$. Suppose that $V(0) = V_{reset}$. Then the time until the neuron spikes (the interspike interval) t_{isi} is the time at which the voltage reaches the threshold potential:

$$V(t_{isi}) = V_{th} = \mathcal{E}_\ell + R_m I_e + (V_{reset} - \mathcal{E}_\ell - R_m I_e) e^{-t_{isi}/\tau_m}. \tag{16}$$

Solving for the firing rate r_{isi} (the inverse of the inter-spike interval) gives

$$r_{isi} = \frac{1}{t_{isi}} = \left[\tau_m \log \left(\frac{V_{reset} - \mathcal{E}_\ell - R_m I_e}{V_{th} - \mathcal{E}_\ell - R_m I_e} \right) \right]^{-1}. \tag{17}$$

Note that this expression is valid when $V_{th} - \mathcal{E}_\ell > R_m I_e$. The firing pattern for a simulated neuron with time-varying input current is shown in Figure 2.2. We can use the approximation $\log(1+z) \approx z$ for small z to show that r_{isi} grows linearly with I_e for large I_e. It's also possible to consider alternative models for the dynamics, such as *quadratic* integrate-and-fire (QIF) and *exponential* integrate-and-fire (EIF) models, which take the following general forms:

$$\tau \frac{dV}{dt} \propto V^2 + \beta V + V_{ext}(t), \tag{18}$$

$$\tau \frac{dV}{dt} \propto \exp(V/\gamma V_0) - \alpha V + V_{ext}(t), \tag{19}$$

respectively. The neuron is said to fire when $V \to \infty$. The EIF is generally a better model for real neurons because the time it takes to fire can be tuned by its parameters, while for the QIF (and LIF) it's dependent solely on the membrane time constant. The EIF is also a better fit for experimental data.

2.4 Active Channels and Voltage-Dependent Conductance

Many important biophysical properties of neurons arise as a result of changing channel conductances. There are several factors that can lead to varying conductances, such as synaptic conductances that depend on the presence or absence of a neurotransmitter, or channels that depend on internal messenger molecules or the concentration of ions like Ca^{2+}. Here, however, we'll focus on *voltage-dependent* conductances, which depend on the membrane potential of the neuron. Assuming independence among channels, we can define the conductance per unit area of membrane for channel type i as follows:

$$g_i \coloneqq \rho_i g_i^{open} P_i = \bar{g}_i P_i, \tag{20}$$

where ρ_i is the density of channels of type i in the membrane, g_i^{open} is the conductance of an open channel of type i, and P_i is the probability that any given such channel is open at a given time. $\bar{g}_i$ is then the conductance per unit area if all such channels are open; units typically range from $\mu S/mm^2$ to mS/mm^2. Two important channels are the delayed-rectifier K^+ conductance and the fast Na^+ conductance.

Persistent Conductances The delayed-rectifier K^+ conductance that is responsible for repolarizing a neuron after it fires is an example of a *persistent conductance* channel. Channels with persistent conductance (depicted in figure 2.3A) behave as though they only carry one kind of *gate* that swings open in response to a voltage-dependent sensor. Opening of the gate(s) is termed *activation* and closing of the gate is referred to as *deactivation*. For this type of channel, the probability that it's open, P_{K^+}, increases when the neuron is depolarized and decreases when it is hyperpolarized.

The gating mechanism of the delayed-rectifier K^+ channel consists of four identical subunits, which appear to open independently. In general, if k independent events are required for a channel to open, P_{K^+} can be written as

$$P_{K^+} = n^k = n^4, \tag{21}$$

where n is the probability that any of the gating events has occurred (i.e., that a gate subunit is open; $1-n$ is the probability it is closed). The variable n is called a *gating variable*, and a description of its voltage and time dependence is sufficient for a description of the conductance. We model the transition probabilities over a time interval dt as follows (summarized in Figure 2.4):

$$\begin{cases} p(\text{closed} \to \text{open}) & = \alpha(V)dt \\ p(\text{open} \to \text{closed}) & = \beta(V)dt. \end{cases} \tag{22}$$

To obtain a differential equation governing these gating dynamics, we can write

$$n(t+dt) = p(\text{open at } t+dt) \tag{23}$$

$$= p(\text{open}(t))p(\text{open}(t+dt)|\text{open}(t)) + p(\text{closed}(t))p(\text{open}(t+dt)|\text{closed}(t)) \tag{24}$$

$$= n(t)(1-\beta dt) + (1-n(t))\alpha dt. \tag{25}$$

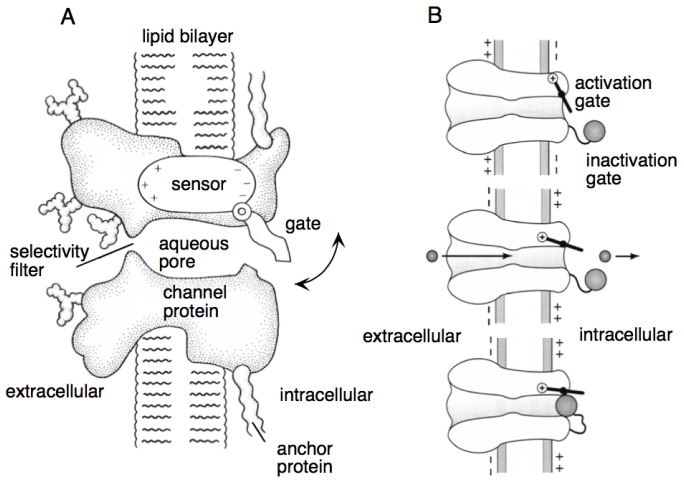


Figure 2.3: Simplified depictions of persistent (A) and transient (B) conductance channels. Descriptions in text.

We then use a linear Taylor approximation, $n(t+dt) \approx n(t) + dt\frac{dn}{dt}$. Applying this gets us

$$n(t) + dt\frac{dn}{dt} \approx n(t)(1-\beta dt) + (1-n(t))\alpha dt \tag{26}$$

$$= n(t) - n(t)\beta dt + \alpha dt - n(t)\alpha dt \tag{27}$$

$$\Rightarrow \frac{dn}{dt} \approx -n(t)\beta + \alpha - n(t)\alpha \tag{28}$$

$$= \alpha(1-n(t)) + \beta n(t) \tag{29}$$

$$= \alpha - (\alpha+\beta)n(t) \tag{30}$$

Dividing both sides of eq. 30 by $\alpha+\beta$ gives

$$\underbrace{\frac{1}{\alpha(V)+\beta(V)}}_{\tau_n(V)}\frac{dn}{dt} = \underbrace{\frac{\alpha(V)}{\alpha(V)+\beta(V)}}_{n_\infty(V)} - n(t), \tag{31}$$

$$\Rightarrow \tau_n(V)\frac{dn}{dt} = n_\infty(V) - n(t). \tag{32}$$

This indicates that for a fixed voltage, the opening probability approaches the limiting value $n_\infty(V)$ exponentially with time constant $\tau_n(V)$. Simple thermodynamic arguments (see Dayan & Abbott p. 170) can be made to show that $n_\infty(V)$ is sigmoidal–depolarization causes n to grow towards 1, and hyperpolarization causes it to shrink toward 0. Following this, the opening rate α is an increasing function of V, while β is decreasing. These functions are usually fitted using experimental data obtained from voltage clamping. Example traces of α, β, n_∞, and τ are plotted in Figure 2.5.

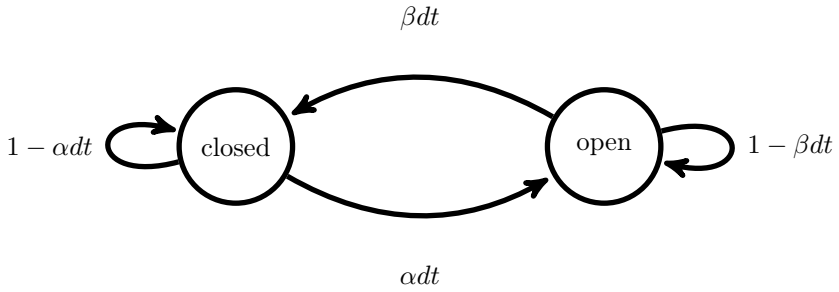


Figure 2.4: Markovian transition dynamics for active channel gates.

Transient Conductances Some channels only open transiently when the membrane potential depolarizes because they contain gates with opposite voltage dependences. The fast Na^+ conductance is an example of such

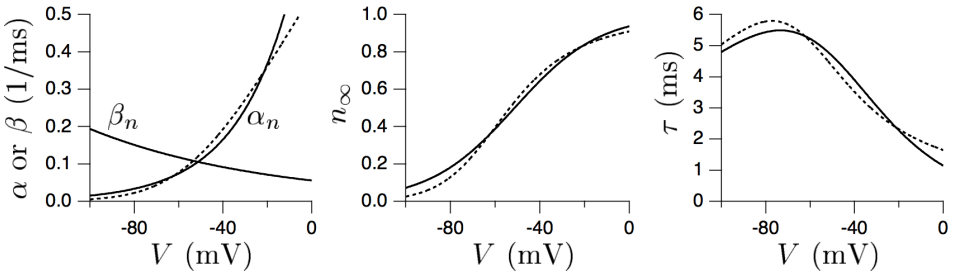


Figure 2.5: Example plots of channel opening and closing rates (left), limiting values for the opening probability (center), and the time constant (right) for the delayed-rectifier K^+ conductance.

a channel. Schematically, it can be thought of as having $k = 3$ swinging activation gates m who increase their probability of opening with increasing voltage, and an inactivation gate/ball h ($k = 1$) which closes with depolarization (Figure 2.3B). For the channel to conduct, both sets of gates must be open, which has probability

$$P_{\mathrm{Na}^+} = m^k h = m^3 h. \tag{33}$$

The probability variables m and h follow analogous equations to n, with similar forms for α and β. The steady state activation and inactivation functions $m_\infty(V)$ and $h_\infty(V)$, along with the associated time constants, are also similar to those for the K channel (although h_∞ is inverted, as it's an inactivation variable). These functions are visualized in Figure 2.6. To activate such a transient channel, it's required that both the m and h gates are nonzero–to do this maximally, it's best for the neuron to first hyperpolarize (activating h), and then quickly depolarize (activating n). The point of maximum activation is the intersection of the two curves–note that this is approximately the threshold voltage for spiking in a neuron.

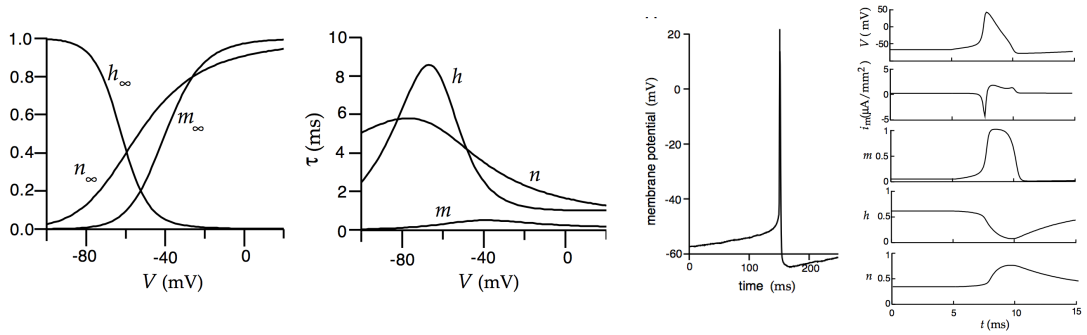


Figure 2.6: Example plots of steady-state values for the opening probabilities of the Na^+ and K^+ channels (left), along with associated time constants (middle-left), and an example action potential (middle-right) and the traces of each gate during it (right). The behavior of the gates during an action potential is as follows: From a hyperpolarized state, the m gates open quickly (see the time constant), allowing Na^+ to flood in. This rapidly depolarizes the neuron, causing the the slower h gates to shut, stopping the influx of Na^+ and re-hyperpolarizing the neuron. The persistent n gates then open, causing the slight re-depolarization to the steady-state at the end. This process is summarized by the rightmost panel. Note that if m and h had the same time constants, they would cancel each other's effects and nothing would happen. In general, the time constants determine the width of the action potential.

2.5 The Hodgkin-Huxley Equations

The Hodgkin-Huxley (HH) equations are simply a condensation of what we've derived so far, modeling the effects of passive and active channels on the membrane voltage dynamics. Combining equations 14, 20, 21, and 33, and ignoring external current injection, we get

$$C\frac{dV}{dt} = -i_m + \underbrace{\frac{I_{ext}}{A}}_{=0} = -\bar{g}_\ell(V - \mathcal{E}_\ell) - \bar{g}_{\mathrm{Na}^+} m^3 h(V - \mathcal{E}_{\mathrm{Na}^+}) - \bar{g}_{\mathrm{K}^+} n^4 (V - \mathcal{E}_{\mathrm{K}^+}). \tag{34}$$

Dividing both sides by $\bar{g}_\ell$ gives

$$\boxed{\tau\frac{dV}{dt} = -(V - \mathcal{E}_\ell) - \rho_{\mathrm{Na}^+} m^3 h(V - \mathcal{E}_{\mathrm{Na}^+}) - \rho_{\mathrm{K}^+} n^4 (V - \mathcal{E}_{\mathrm{K}^+}),} \tag{35}$$

where $\rho_{\text{Na}^+} = \bar{g}_{\text{Na}^+}/\bar{g}_\ell \approx 400$ and $\rho_{\text{K}^+} = \bar{g}_{\text{K}^+}/\bar{g}_\ell \approx 120$. We can also generalize eq. 32 for the dynamics of the opening probability of each gating variable, giving

$$\boxed{\tau_x(V)\frac{dx}{dt} = x_\infty(V) - x(t), \quad x \in \{m, n, h\}.} \tag{36}$$

Equations 35 and 36 are the **Hodgkin-Huxley** equations. Eq. 36 can be equivalently expressed as

$$\tau_x(V)\frac{dx}{dt} = \alpha_x(1 - x(t)) + \beta_x x(t) \quad \text{(see eq. 29)}. \tag{37}$$

As these are highly nonlinear equations in four variables, they can't be solved directly, and must be approximated. There are two standard ways to do this:

2.5.1 Simplified HH Neurons

In the first simplifying assumption, we assume that gating variables always hold their steady-state values–that is, $\tau_x = 0 \Rightarrow x = x_\infty(V)\,\forall x$. Then eq. 35 becomes

$$\tau\frac{dV}{dt} = -(V - \mathcal{E}_\ell) - \rho_{\text{Na}^+} m_\infty^3 h_\infty (V - \mathcal{E}_{\text{Na}^+}) - \rho_{\text{K}^+} n_\infty^4 (V - \mathcal{E}_{\text{K}^+}) + V_{ext}(t), \tag{38}$$

a one-dimensional system. This is equivalent to assuming that the membrane time constant τ is much larger than the gating time constants τ_m, τ_h, and τ_n. This yields a cubic function on the V-$\dot{V}$ phase plane with three roots—the two leftmost roots bound a local minimum (the left root is stable, and the center root is unstable), and the middle and the right root bound a stable local maximum (Figure 2.7). Changing the external current shifts the cubic function up and down. When it is sufficiently high, the left and center roots disappear, leaving only the right (stable) point. On the other hand, setting V_{ext} quite low shifts the function downwards, destroying the center and right roots and leaving only the left (stable) point. Thus, by modulating the external input, the neuron can effectively function as a switch between high (ON) and low (OFF) states. This could be a realistic model, except it results in dynamics that are very energy-intensive—ions pumps need to work incredibly hard to maintain the higher (ON) state. The shape of the resulting dynamics is also inconsistent with experimental evidence.

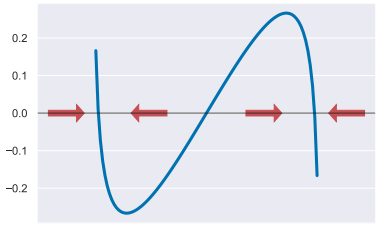


Figure 2.7: HH approximation #1: all gates set to their equilibrium values.

The second possible approximation (and a more biologically realistic one) is to let $m \to m_\infty(V)$, as the time constant for m is so much lower than for n and h, and to combine the slower n and h conductances—more precisely, we combine n and $1 - h$—into one dynamical variable $w(t)$ with its own reversal potential $\mathcal{E}_w$ and average conductance $\bar{\rho}_w$. This gives the simplified 2D *Morris-Lecar model* of action potential dynamics:

$$\tau\frac{dV}{dt} = -(V - \mathcal{E}_\ell) - \bar{\rho}_w w(t)(V - \mathcal{E}_w) - \bar{\rho}_m m_\infty(V)(V - \mathcal{E}_m) + V_{ext}(t) \tag{39}$$

$$\tau_w\frac{dw}{dt} = w_\infty(V) - w(t), \tag{40}$$

with $\tau_w(V) \approx \tau_n, h(V)$. Although a simplification, this system retains the qualitative behavior of the HH equations, as visualized in Figure 2.8.

Because the system is 2D, we can easily examine its behavior on the V-w plane. We can see that the nullclines (Figure 2.8B) imply three fixed points, and it turns out the leftmost is always stable, corresponding to the resting membrane potential. The right fixed point is typically unstable, and the center fixed point is a saddle point. Changing the input current via V_{ext} shifts the V-nullcline ($F = 0$) up and down in the plane. We can see that as the external input current increases and the V-nullcline shifts up, the left stable fixed point and the saddle point grow closer together and eventually disappear, leaving only the unstable fixed point at

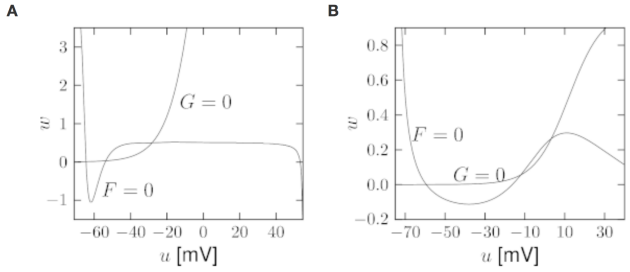


Figure 2.8: HH approximation #2: reduction to a 2D system. The left nullclines (A) are those for the HH model, rigorously reduced to 2D via a linear fitting for $w(t)$ and the right (B) are those for the Morris-Lecar approximation. Notationally, $u = V$, F is the V-nullcline and G is the w-nullcline.

high V. However, since the derivatives around it still point towards the fixed point, the **Poincaré-Bendixson theorem**[1] tells us that the system must form a limit cycle around it. In other words, if you increase the input current sufficiently—above some threshold I_θ—the neuron starts spiking repeatedly, and the change in the number of fixed points at $I_{ext} = I_\theta$ is called a *bifurcation*. The input current I is then called a *bifurcation parameter*. In neuroscience, I_θ, the threshold current required to induce spiking, is called the *rheobase*.

It's then natural to investigate the frequency of the resulting limit cycle oscillations, as it gives insight into the neuron's firing rate response to a given constant input I—its so-called *gain function*. Consider the behavior of the system when $I < I_\theta$ and the right fixed point is unstable. In this case, trajectories starting to the right of the saddle wrap around the unstable node counter-clockwise, eventually returning to the stable fixed point (Figure 2.9, left).

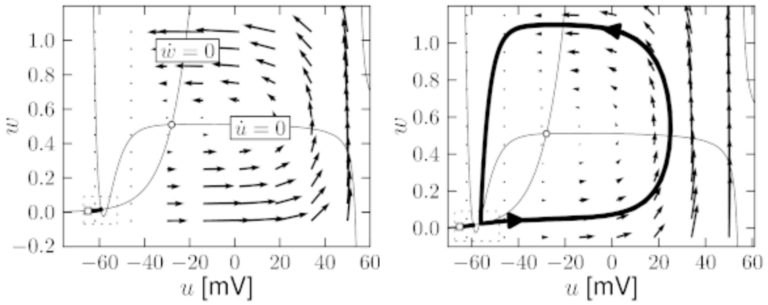


Figure 2.9: The phase plane trajectories for a Type I neuron (left) and a Type II neuron (right).

When I grows slightly larger than I_θ and the dynamics bifurcate, this behavior is maintained in the resulting limit cycle, such that the trajectories still pass through the area where the stable fixed point used to be. Moreover, when they pass through, the magnitude of the derivatives decreases, lowering the oscillation frequency and slowing the firing rate. When I grows even larger, this slowdown is alleviated and the spiking frequency increases. Neurons with this type of behavior are called *Type I*, and are characterized by a smooth, monotonic increase in firing rate as the input current increases. When two fixed points merge like this, it's called a *saddle node* bifurcation (for more on bifurcations, see 2.6). Intuitively, such dynamics are useful for encoding a continuous quantity, such as the overall strength of pre-synaptic input.

When the right fixed point is a limit cycle to begin with, however, different behavior occurs. In this case, the oscillatory trajectories pass by just to the right of the saddle, instead of the left stable region (Figure 2.9, right), and the dynamics are stuck at the low fixed point—there is no firing. Then, when I increases above I_θ and the left and center fixed points vanish, trajectories are pushed onto this limit cycle, without entering the region where the stable point used to be and slowing down. This type of transition, from a stable fixed point to a limit cycle, is called a *Hopf bifurcation*. Neurons whose gain function (and firing rate) jumps suddenly to a high value from zero when $I > I_\theta$ are termed *Type II*. This type of behavior is useful for encoding a binary variable, acting like a switch with ON/OFF settings.

[1] In a real, differentiable dynamical system defined on an open, simply connected subset C of $\mathbb{R}^2$, if the vector field (gradient) along the boundary points towards the interior of C and C contains no fixed points, then there exists a closed trajectory (or limit cycle) in C.

2.6 Bifurcation Theory & Neurons

This material is drawn from [Izhikevich, 2007], and is an attempt to explain what Andronov-Hopf and saddle point bifurcations are, and why they have relevance to the behaviour of neurons. It's very skimpy on details, so if you're looking for more I recommend the Izhikevich book – it's fun reading.

Neurons are excitable systems: depending on the input they receive, they can transition from resting to periodic firing. This observation alone is enough to begin an interesting conversation. Neurons are non-linear dyamical systems, and these two states correspond to different dynamical features: resting state = a stable equilibrium, and periodic firing = a limit cycle. There are only a few ways in which a system may transition from equilibrium to a stable limit cycle, especially in a two-dimensional system like the reduced Hodgkin-Huxley model. This means we can get instant insight into the behaviour of a neuron just by understanding the qualitative form of its transition from resting to periodic spiking.

In two dimensions, there are four ways a system can make this transition, known in the business as a bifurcation (Figure 2.10). Experimentally, all of these have been found in neurons, with the channel properties of the neuron determining which type is seen. Current can act as the bifurcation parameter driving the system left (increased current) and right (decreased) on these diagrams, transitioning from spiking to non-spiking, with potential for hysteresis depending on the type of bifurcation.

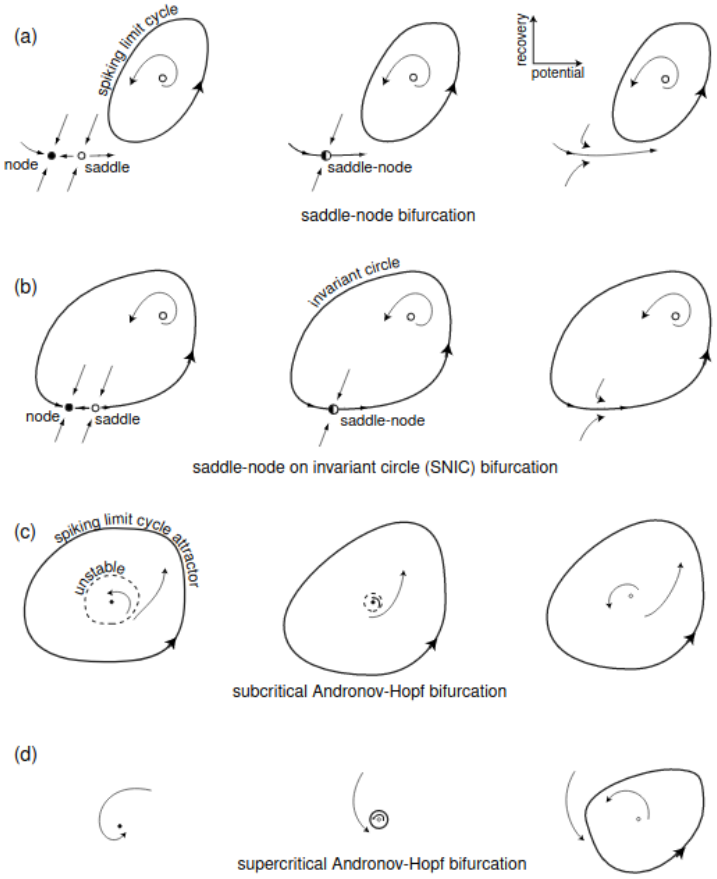


Figure 2.10: (Copy of Figure 1.12 from [Izhikevich, 2007]) The four ways a neuron can transition from stable to firing – that is, the four types of bifurcations.

A - **Saddle-Node Bifurcation**: a stable node and a limit cycle co-exist. As the bifurcation parameter is varied, the stable node collides with a saddle and annihilates it, leaving only the stable limit cycle. The stable node corresponds to the neuron at rest; once the saddle-node is annihilated, the neuron falls into the stable limit cycle and begins firing at a finite frequency.

B - **Saddle-Node on Invariant Circle**: The stable equilibrium is on a looping 1D trajectory together with a saddle. As the bifurcation parameter changes, these two fixed point collide and annihilate, leaving only the looping trajectory which is now a stable limit cycle. The neuron begins at the stable point, then begins firing following the annihilation event. Each loop around the limit cycle corresponds to one spike, and the spiking rate increases slowly, starting from zero at the transition. This is because the dynamics around the saddle-node are very slow even after annihilation. As the bifurcation parameter increases further, the speed in this region accelerates and hence the firing rate will pick up.

C - **Subcritical Andronov-Hopf**: A stable node and stable limit cycle coexist, separated by an unstable limit cycle. As the current is increased the unstable limit cycle approaches the stable fixed point, eventually merging to create an unstable fixed point. At this time the neuron, which had been happily at rest at the stable fixed point, falls into the limit cycle and begins firing. Since the limit cycle already exists the firing begins at a finite rate. Due to the swirling local dynamics around the stable equilibrium (especially when the neuron is close to firing, which it always is) these neurons show significant subthreshold membrane oscillations in response to noise (unlike both saddle-node type bifurcations where the local region shows more sedate, stable dynamics).

D - **Supercritical Andronov-Hopf**: A stable fixed point transitions into an unstable fixed point and births a stable limit cycle, onto which the neuron instantly falls. The size of the membrane fluctuations will, in this case, depend on the size of the limit cycle, which in turn is determined by the size of the current input. Again due to the swirling nature of the local dynamics these neurons will show subthreshold oscillations. Since there is only ever one stable setting this type of bifurcation shows no hysteresis.

All of these properties can be seen in 2.11, which shows the membrane voltage of neurons going through each of these bifurcations. This simple classification allows us to see some wood for the trees of biochemical details, and some people think these dynamics might be vital to the computational task the neurons are performing.

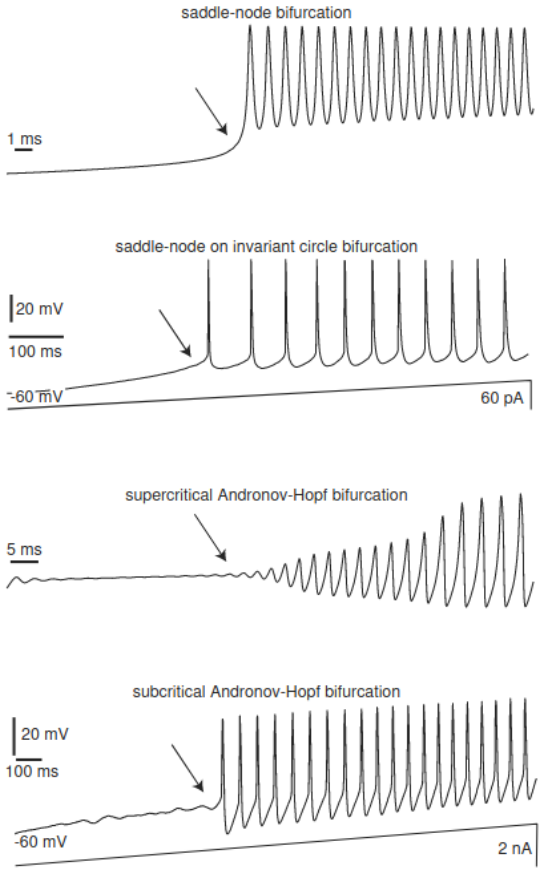


Figure 2.11: (Copy of Fig 6.1 from [Izhikevich, 2007]) Shows the voltage responses of neurons undergoing the four types of bifurcation as the current increases on a ramped schedule. Notice: the subthreshold oscillations in both Andronov-Hopf bifurcations, the increasing voltage in supercritical Andronov-Hopf, and the increasing rate of spiking in saddle-node on invariant circle bifurcations. These should all make sense to you!

In the 2D reduced Hodgkin-Huxley model Peter introduced we can see two different types of bifurcation happening, depending of the parameter setting. To derive his 2D version we make two approximations, the first (fairly reasonably) sets $\tau_m = 0$. The second, much more unreasonably, simply ignores the potassium conductance $\rho_K = 0$. This is a very bad model, especially since there are much better 2D models like the Morris-Lecar model, but it shows the details we are interested in. This gives us two equations:

$$\tau_v \frac{dV}{dt} = -(V - \epsilon_L) - \rho_{Na} m_\infty^3(V) h (V - \epsilon_{Na}) + R I_{Ext}$$

$$\tau_h \frac{dh}{dt} = h_\infty(V) - h$$

The first case we will examine shows a saddle-point bifurcation, figure 2.12. This is reliant on the third fixed point being unstable, which is not guaranteed but depends on the local linear dynamics. Neurons must keep this fixed point unstable else they'll end up at a tonic high firing rate and lose all their ions quicker than the pumps can restore them, causing death. There are apparently some disease where this happens, nasty...

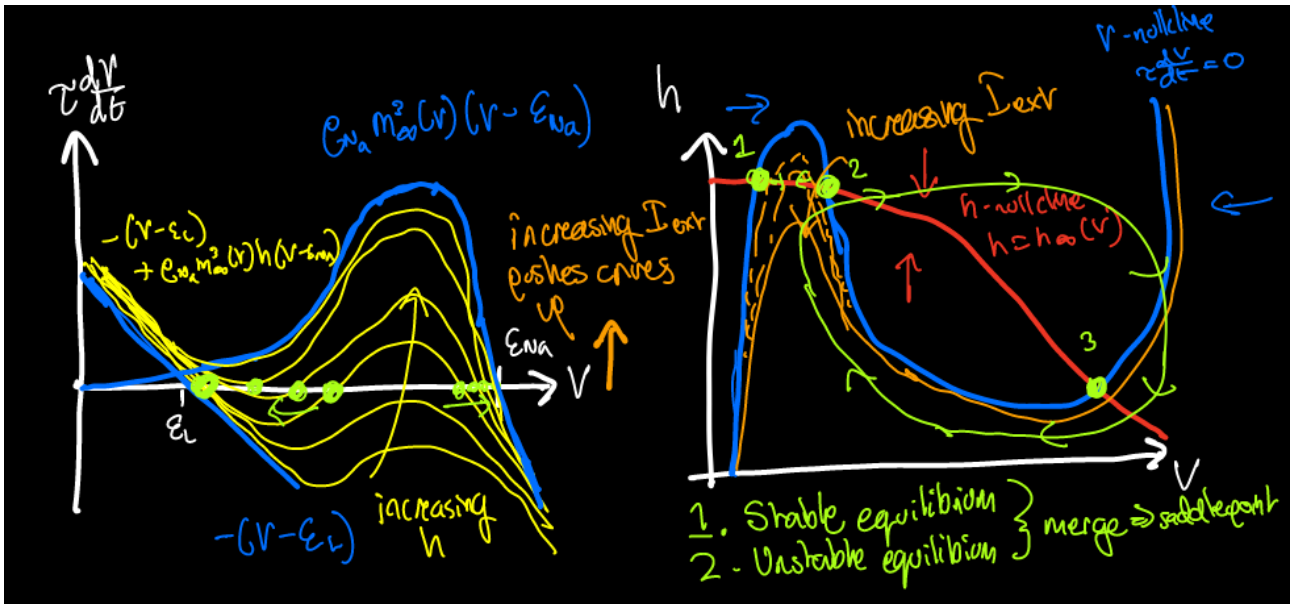


Figure 2.12: On the left we graphically calculate the shape of the the V nullcline for this reduced HH model. As h varies between 0 and 1 the dynamics go from purely leak current with one fixed point at ϵ_L to mostly sodium current with a fixed point at ϵ_{Na}. The behaviour of $m_\infty(V)$ makes this interesting. External current shifts the curves up, meaning the merging of fixed points happens at lower values of h. This is seen in the right plot of nullclines, as the current increases the V-nullcline transitions from the blue curve to the orange one. The fixed points 1 and 2 merge as the current increases, and the system undergoes a saddle-point on invariant circle bifurcation leading to the emergence of stable spiking (assuming fixed point 3 is unstable).

Alternatively the parameters might be such that the nullclines look like figure 2.13. Then there is only one fixed point and it is the changing nature of this point that causes the bifurcation. As the input current changes the nullclines will shift, causing the local environment of the fixed point to change. If it changes such that one of the eigenvalues begins having positive real part then it will flip to being unstable. By Poincaré-Bendixson (see section D), as long as the gradients far enough away are still pointing towards the fixed point there will be a stable limit cycle around it; the neuron will begin flowing around the limit cycle, representing firing. This is exactly the supercritical Andronov-Hopf bifurcation shown in Figure 2.10D.

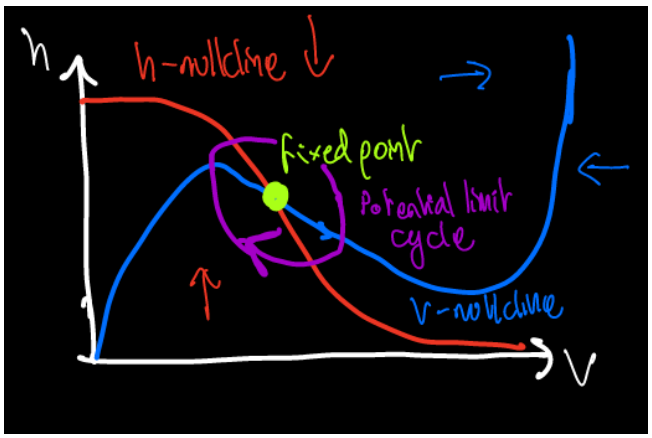


Figure 2.13: Nullclines for Peter's 2D HH model undergoing supercritical Andronov-Hopf Bifurcation into spiking.

2.7 Dendrites and Axons

In multi-compartmental models of neurons, we model the axons and dendrites of a neuron as *cables*. Since they are long and narrow, we assume uniformity in the radial dimension and model variations in membrane potential along the axial/longitudinal dimension:

$$V(x,t)$$

where x is the axial/longitudinal position along the cable.

As we did in the single compartment case, we want to derive the temporal dynamics of the membrane potential at a given position x, given by:

$$C\frac{\partial V}{\partial t} = -I + I_{ext}(x,t)$$

where, as above I_{ext} is an external injected current. We can think about the total current I by considering a small segment of the dendrite with width Δx centered at the longitudinal position x. Here, we have three sources of current, see 2.14:

- incoming axial current from the previous segment centered at $x - \Delta x$, given by the current at the border between the two segments: $I(x - \frac{\Delta x}{2})$
- outgoing axial current to the next segment centered at $x + \Delta x$, given by the current at that border: $I(x + \frac{\Delta x}{2})$
- membrane current generated by passive and active conductances via membrane ion channels: I_m

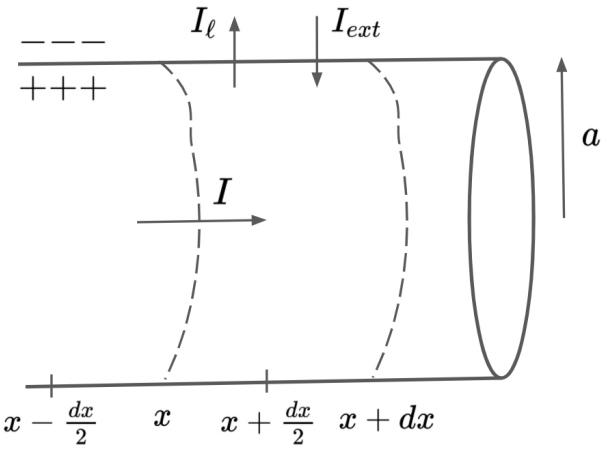


Figure 2.14: Current propagation in a dendrite.

Using Ohm's law to convert current to voltage potential $I = \frac{\Delta V}{R}$, we have:

$$\begin{aligned} C\frac{\partial V}{\partial t} &= I(x-\Delta x) - I(x+\Delta x) - I_m(x) + I_{ext}(x,t) \\ &= \frac{V(x-\Delta x) - V(x)}{R_L} - \frac{V(x) - V(x+\Delta x)}{R_L} - I_m(x) + I_{ext}(x,t) \end{aligned}$$

where we now consider the intracellular *axial resistance* acting on the axial current

$$R_L = \frac{r_L l}{A} = \frac{r_L \Delta x}{\pi a^2}$$

where l is the cable length (in this case equal to Δx), a is the radius of the cross-section of the cable ($A = \pi a^2$ is thus the cross-sectional area), and the constant $r_L \sim 10^3 \Omega$mm is an inherent property of the neurite cytoplasm. We now approximate $V(x \pm \Delta x)$ with a second-order Taylor expansion in space:

$$\begin{aligned} C\frac{\partial V}{\partial t} &\approx \frac{\left(\left(V(x) - \Delta x \partial_x V(x) + \frac{\Delta x^2}{2}\partial_x^2 V(x)\right) - V(x)\right) - \left(V(x) - \left(V(x) + \Delta x \partial_x V(x) + \frac{\Delta x^2}{2}\partial_x^2 V(x)\right)\right)}{R_L} - I_m(x) + I_{ext}(x,t) \\ &= \frac{\Delta x^2}{R_L}\frac{\partial^2 V}{\partial x^2} - I_m(x) + I_{ext}(x,t) \end{aligned}$$

Assuming the axial capacitance negligible, the capacitance term on the left-hand side becomes the membrane capacitance $C_m = c_m A = c_m 2\pi a \Delta x$ (where $A = membrane$ area = dendrite circumference $\times$ length), so we can divide both sides by the membrane area and multiply by the specific membrane resistance to get our dynamics in terms of our good old membrane time constant:

$$\begin{aligned}\tau_m \frac{\partial V}{\partial t} &= \frac{r_m}{2\pi a \Delta x R_L}\Delta x^2 \frac{\partial^2 V}{\partial x^2} - \frac{r_m}{2\pi a \Delta x} I_m(x) + \frac{r_m}{2\pi a \Delta x} I_{ext}(x,t) \\ &= \frac{r_m a}{2 r_L}\frac{\partial^2 V}{\partial x^2} - r_m i_m(x) + r_m i_{ext}(x,t) \\ &= \lambda^2 \frac{\partial^2 V}{\partial x^2} - r_m i_m(x) + r_m i_{ext}(x,t)\end{aligned}$$

where the membrane and external currents are now in units of current per unit membrane surface area (i.e. area of surrounding cell membrane). The constant

$$\lambda = \sqrt{\frac{r_m a}{2 r_L}}$$

(in mm^2) is called the *electrotonic length* which, as we will see below, sets the scale of spatial (i.e. longitudinal) variation in membrane potential along the given neurite. As before, τ_m sets the scale of temporal variation.

To be able to perform some analysis on this model, we ignore the non-linear action potential-generative active conductances contained in i_m, leaving only the leak current $i_m = (V - E_L)/r_m$ and thus giving us the *passive cable equation*:

$$\tau_m \frac{\partial V}{\partial t} = \lambda^2 \frac{\partial^2 V}{\partial x^2} - (V - E_L) + r_m i_{ext}(x,t)$$

This simplification linearizes the dynamics, thus making it amenable to analysis. Furthermore, it is not a bad approximation whenever the membrane potential is near the resting potential or the dendrites don't have any active channels. Note that we have entirely ignored synaptic conductances until now, so our analysis is restricted to the case of external (i.e. electrode) current injection (although D&A pg. 207, top, claim that current injection can mimic the effects of a synaptic conductance).

In solving the passive cable equation, it is necessary to assume boundary conditions at branching points and end points of the cable, where the dynamics will change. Different assumptions can be made here, and in the below we take the simplest scenario: an infinite cable. Our only constraint is then that the membrane potential remain bounded for all x, t. While no dendrite is infinite, this is still a good approximation for sites far away from branch- or end- points of the dendrite.

We begin by considering the case of constant current injection isolated in space, i.e.

$$i_{ext}(x,t) = i_{ext}\delta(x)$$

where $x = 0$ is the exact point at which the current is injected. This will push the membrane potential to an equilibrium state given by

$$0 = \lambda^2 \frac{\partial^2 V}{\partial x^2} - (V - E_L) + r_m i_{ext}\delta(x)$$

Letting $u(x,t) = V(x,t) - E_L$, we can solve the homogenous second-order ODE (see section C.2) for $x \neq 0$, where $\delta(x) = 0$:

$$\begin{aligned}\lambda^2 \frac{\partial^2 u}{\partial x^2} - u &= 0 \\ \Leftrightarrow u(x) = c_1 e^{\frac{x}{\lambda}} + c_2 e^{-\frac{x}{\lambda}}, &\quad x \neq 0\end{aligned}$$

Since $u(x)$ has to be bounded for $x \to \infty, -\infty$, we then have:

$$u(x) = \begin{cases} c_1 e^{-\frac{x}{\lambda}} & \text{if } x > 0 \\ c_2 e^{\frac{x}{\lambda}} & \text{if } x < 0 \end{cases} = \Theta(x) c_1 e^{-\frac{x}{\lambda}} + \Theta(-x) c_2 e^{\frac{x}{\lambda}}$$

where $\Theta(x)$ is the Heaviside function. We therefore have a discontinuity at $x = 0$, at which point we still don't know what the membrane potential is. To find this out, we solve for c_1, c_2 by computing the second derivative and plugging back into the original differential equation:

$$\begin{aligned}\lambda \frac{\partial u}{\partial x} &= -\Theta(x) c_1 e^{-\frac{x}{\lambda}} + c_1 \delta(x) + \Theta(-x) c_2 e^{\frac{x}{\lambda}} - c_2 \delta(x) \\ &= -\Theta(x) c_1 e^{-\frac{x}{\lambda}} + \Theta(-x) c_2 e^{\frac{x}{\lambda}} + \lambda(c_1 - c_2)\delta(x) \\ \lambda^2 \frac{\partial^2 u}{\partial x^2} &= \Theta(x) c_1 e^{-\frac{x}{\lambda}} + \Theta(-x) c_2 e^{\frac{x}{\lambda}} + \lambda(-c_1 - c_2)\delta(x) + \lambda^2(c_1 - c_2)\delta'(x) \\ &= u(x) - \lambda(c_1 + c_2)\delta(x) + \lambda^2(c_1 - c_2)\delta'(x)\end{aligned}$$

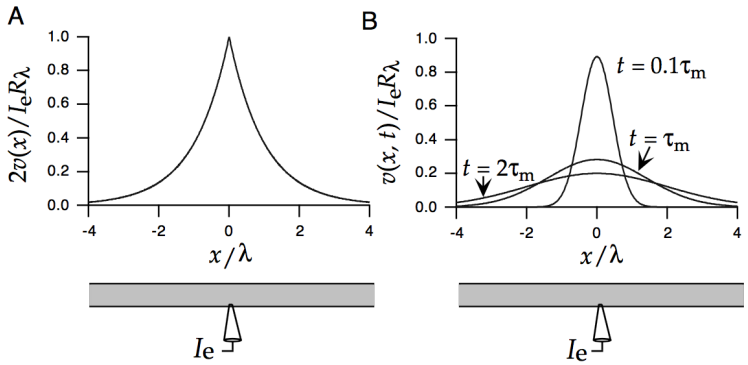


Figure 2.15: Voltage propagation in an infinite cable with injection at $x = 0$. (A) Solution for a constant electrode current. It decays exponentially from the injection point. (B) The solution for a (time-dependent) δ-pulse of current. Its described by a Gaussian centered at the injection point that broadens and shrinks in amplitude over time.

The δ-functions appear from the taking the derivative of the Heaviside functions. Plugging this back into the original differential equation at temporal equilibrium, we get

$$\begin{aligned} \lambda^2 \frac{\partial_x^2 u}{\partial x^2} &= u - r_m i_{ext} \delta(x) \\ \Leftrightarrow u - \lambda(c_1 + c_2)\delta(x) + \lambda^2(c_1 - c_2)\delta'(x) &= u - r_m i_{ext}\delta(x) \\ \Leftrightarrow -\lambda(c_1 + c_2)\delta(x) + \lambda^2(c_1 - c_2)\delta'(x) &= -r_m i_{ext}\delta(x) \end{aligned}$$

Since there is no term on the RHS with $\delta'(x)$, we conclude that $c_1 = c_2 = c$, giving us

$$\begin{aligned} -2c = -\frac{r_m}{\lambda} i_{ext} = -\frac{r_m}{\lambda 2\pi a} I_{ext} &= -R_\lambda I_{ext} \\ \Leftrightarrow c &= \frac{R_\lambda}{2} I_{ext} \\ \Rightarrow u(x) &= \frac{R_\lambda}{2} I_{ext} e^{-\frac{|x|}{\lambda}} \end{aligned}$$

at equilibrium (alternatively, we could have just assumed $c_1 = c_2$ on the grounds that the spatial gradient of the membrane potential should be continuous). The ratio of equilibrium potential at the injection site ($x = 0$) to the injected current I_{ext} is called the *input resistance* R_λ of the cable. This depends on a combination of the axial resistance and membrane resistance (in $R_\lambda \propto r_m, \sqrt{r_L}$). We have thus found that, when a constant current is injected into a dendrite at an infinitely small point $x = x_0$, at equilibrium the membrane potential will drop off exponentially to each side of x_0, with characteristic length scale given by the electrotonic length λ:

$$V(x) - E_L = \frac{R_\lambda}{2} I_{ext} e^{-\frac{|x - x_0|}{\lambda}}$$

see figure 2.15. Thus, to be able to propagate a signal all the way down to the soma, dendrites can't be much longer than λ or the current won't make it far enough before decaying to 0. This provides some insight into why real dendrites are relatively short (or have active ion channels).

Although the scenario of current being injected into an infinitely small point on the dendrite is completely unrealistic, the above analysis is still useful for understanding the membrane potential dynamics in a dendrite near resting potential E_L (recall we're ignoring all active conductances). Furthermore, the solution to the passive cable equation with $i_{ext}(x) = \delta(x)$ is in fact the Green's function (section C.3) for solving for the steady state of the more general case with an external current that varies smoothly over space and time. In other words, since our equation is linear, we can obtain the solution for a spatially smooth current injection by summing together the solutions to spatially isolated currents. Let L be the linear operator corresponding to our passive cable equation at equilibrium, i.e.

$$Ly = \lambda^2 \frac{\partial^2 y}{\partial x^2} - y$$

Letting $u_\delta(x)$ designate the solution found above, we then have that, at equilibrium,

$$Lu_\delta(x) = -r_m i_{ext} \delta(x)$$

Consider now the case of a constant current injection varying smoothly over space:

$$\tau_m \frac{\partial u}{\partial t} = \lambda^2 \frac{\partial^2 u}{\partial x^2} - u + r_m f(x)$$

(where $f(x)$ is in units of current per unit area). To solve for distribution of membrane potential over space at the temporal equilbrium, we set $\frac{\partial u}{\partial t} = 0$ and use the *Green's function* (section C.3) given by u_δ:

$$\begin{aligned}
\lambda^2 \frac{\partial_x^2 u}{\partial x^2} - u &= -r_m f(x) \\
\Leftrightarrow Lu &= -r_m f(x) \\
&= \int_{-\infty}^{\infty} -\delta(x - x') r_m f(x') \mathrm{d}x' \\
&= \int_{-\infty}^{\infty} \frac{L u_\delta(x - x')}{i_{ext}} f(x') \mathrm{d}x' \\
&= L \int_{-\infty}^{\infty} \frac{u_\delta(x - x')}{i_{ext}} f(x') \mathrm{d}x' \\
\Leftarrow u(x) &= \frac{1}{i_{ext}} \int_{-\infty}^{\infty} u_\delta(x - x') f(x') \mathrm{d}x'
\end{aligned}$$

where we were able to go from the fourth to the fifth line since integration and differentiation are both linear operators, and L consisted of differentiating with respect to x whereas the integral was over a different variable x'. The awkward left arrow on the last line makes the rather technical point that we have not shown that this is *the* unique solution to the ODE, only that it is *a* solution (I believe a boundary condition at $x = 0$ would suffice to get a unique solution). Recalling that $u_\delta(x) \propto e^{-\frac{|x|}{\lambda}}$ is just a rising and then decaying exponential, the resulting spatial distribution at equilibrium will simply look like a smoothed $f(x)$ as a result of the convolution with $u_\delta(x)$.

The next case to consider is a pulse of injected current isolated in space and time:

$$\tau_m \frac{\partial u}{\partial t} = \lambda^2 \frac{\partial^2 u}{\partial x^2} - u + r_m i_{ext} \delta(x) \delta(t)$$

We solve this by first taking the Fourier transform in space, which gives us a simple first-order ODE:

$$\begin{aligned}
\tau_m \frac{\partial}{\partial t} U(\omega) &= -\lambda^2 \omega^2 U(\omega) - U(\omega) + r_m i_{ext} \delta(t) \\
\Leftrightarrow \frac{\partial}{\partial t} U(\omega) + \frac{\lambda^2 \omega^2 + 1}{\tau_m} U(\omega) &= \frac{r_m i_{ext}}{\tau_m} \delta(t)
\end{aligned}$$

where we used the fact that the Fourier transform of a derivative $\frac{d^n}{dx^n} f(x)$ is equal to $(\omega i)^n F(\omega)$. Solving this, we get:

$$\begin{aligned}
U(\omega, t) &= U(\omega, 0) e^{-\frac{\lambda^2 \omega^2 + 1}{\tau_m} t} + \frac{r_m i_{ext}}{\tau_m} \Theta(t) e^{-\frac{\lambda^2 \omega^2 + 1}{\tau_m} t} \\
&\approx \frac{r_m i_{ext}}{\tau_m} \Theta(t) e^{-\frac{\lambda^2 \omega^2 + 1}{\tau_m} t}
\end{aligned}$$

by setting $u(x, 0) = 0 \Rightarrow U(\omega, 0) = 0$. Taking the inverse Fourier transform of both sides to return to the spatial domain, we note that the exponential term on the RHS is squared exponential in ω, meaning we can easily compute its inverse Fourier transform by putting it into Gaussian form:

$$\begin{aligned}
U(\omega, t) &= \frac{r_m i_{ext}}{\tau_m} \Theta(t) e^{-\frac{t}{\tau_m}} e^{-\frac{\lambda^2 t}{\tau_m} \omega^2} \\
&\overset{\omega \to 2\pi k}{=} \frac{r_m i_{ext}}{\tau_m} \Theta(t) e^{-\frac{t}{\tau_m}} \sqrt{\frac{\tau_m}{4\pi\lambda^2 t}} \sqrt{\frac{4\pi\lambda^2 t}{\tau_m}} e^{-\frac{4\lambda^2 t}{\tau_m} \pi^2 k^2} \\
&= \frac{r_m i_{ext}}{\tau_m \sqrt{\pi B t}} \Theta(t) e^{-\frac{t}{\tau_m}} \sqrt{\pi B t} e^{-B t \pi^2 k^2} \\
\Rightarrow u(x, t) &= \frac{r_m i_{ext}}{\tau_m \sqrt{\pi B t}} \Theta(t) e^{-\frac{t}{\tau_m}} e^{-\frac{x^2}{B t}} \\
&= \frac{R_\lambda I_{ext}}{\sqrt{4\pi\tau_m t}} \Theta(t) e^{-\frac{t}{\tau_m}} e^{-\frac{\tau_m}{4\lambda^2 t} x^2}
\end{aligned}$$

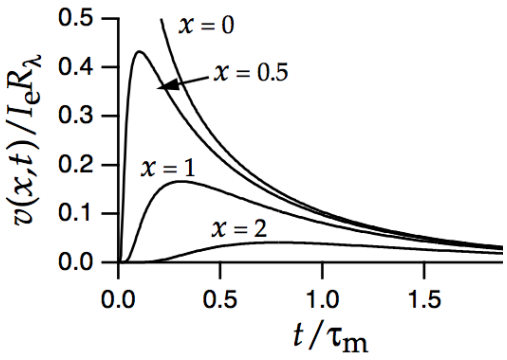


Figure 2.16: Voltage propagation across time for different fixed distances from the point of injection. Greater distances have greater delays in peak time.

where

$$B = \frac{4\lambda^2}{\tau_m}$$

and we took the change of variables $\omega \to 2\pi k$, such that k is in units of frequency (i.e. inverse units of x) and we could exploit our formula for the Fourier transform of a Gaussian function (section E). We thus see that the pulse of current injection decays with distance from the injection site as a Gaussian with width $\sqrt{Bt} \propto \lambda$, which expands over time as $\sqrt{t/\tau_m}$ and the peak decays as $e^{-\frac{t}{\tau_m}}/\sqrt{t}$. In other words, the dynamics of the current injection are a spreading Gaussian in space with integral decaying exponentially in time (see D&A fig 6.7B for a picture). This solution in turn provides the Green's function to solve for the propagation of a current injection that varies over space and time.

If we look at a site x away from the current injection site, the current as a function of time looks like a difference of exponentials, see figure 2.16. with the peak at some $t^* > 0$, later for sites further away. We can thus try to compute a kind of "velocity" of current propagation by computing the amount of time it will take for the current at site x to peak, and dividing the distance x from the current injection site (i.e. the distance travelled) by the time to peak. The time of the peak t^* is easily computed by setting the time derivative of the logarithm of $u(x,t)$ to 0 (assuming $t > 0$):

$$\begin{aligned}
0 &= \frac{\mathrm{d}}{\mathrm{d}t}\bigg|_{t^*}\left[-\frac{t}{\tau_m} - \frac{\tau_m x^2}{4\lambda^2 t} - \frac{1}{2}\log t + \text{const. w.r.t. } t\right] \\
&= -\frac{1}{\tau_m} + \frac{\tau_m x^2}{4\lambda^2 t^{*2}} - \frac{1}{2t^*} \\
&= \frac{4\lambda^2}{\tau_m}t^{*2} + 2\lambda^2 t^* - \tau_m x^2 \\
\Leftrightarrow t^* &= \frac{-\lambda \pm \sqrt{\lambda^2 + 4x^2}}{4\lambda}\tau_m \\
&= \frac{\tau_m}{4}\left(\sqrt{1 + 4x^2/\lambda^2} - 1\right)
\end{aligned}$$

(ignoring the negative root of the quadratic, since $t^* > 0$) which, in the limit of large x is:

$$t^* \approx \frac{\tau_m}{4}\left(\frac{2x}{\lambda}\right) = \frac{\tau_m x}{2\lambda}$$

In this limit, the velocity of current propagation is then:

$$v_{dendrite} = \frac{x}{t^*} \approx \frac{2\lambda}{\tau_m}$$

where the approximation is good for sites far away from the injection site, e.g. the soma when current is injected at a distal dendrite.

Axons, however, often need to propagate signals over long distances and therefore require higher speeds of propagation. Given that r_L and c_m are intrinsic properties of the cell cytoplasm and phospholipid bilayer, the two parameters we can manipulate to achieve higher speeds are a (axon radius) and r_m (membrane resistance). It turns out the mammalian brain does both. To change r_m, long-range projecting axons are often *myelinated*: they are wrapped with layers of cell membrane (*myelin*) that effectively increase the membrane resistance. We

model this by taking $r_m \to \infty$. Rearranging the passive cable equation to take this limit and then using the same strategy as above to solve for the propagation of a pulse of injected current (Fourier transform in space $\to$ solve differential equation in time $\to$ inverse Fourier transform of a Gaussian), we get:

$$\begin{aligned} c_m \frac{\partial V}{\partial t} &= \frac{\lambda^2}{r_m}\frac{\partial^2 V}{\partial x^2} - \frac{V - E_L}{r_m} + i_{ext}\delta(x)\delta(t) \\ &= \frac{a}{2r_L}\frac{\partial^2 V}{\partial x^2} - \frac{V - E_L}{r_m} + i_{ext}\delta(x)\delta(t) \\ \Rightarrow \lim_{r_m \to \infty} \frac{\partial V}{\partial t} &= \frac{a}{2r_L c_m}\frac{\partial^2 V}{\partial x^2} + i_{ext}\delta(x)\delta(t) \\ \Rightarrow V(x,t) &= \frac{i_{ext}}{\sqrt{\pi D t}}\Theta(t)e^{-\frac{x^2}{Dt}} \\ D &= \frac{2a}{r_L c_m} \end{aligned}$$

Note the lack of a term decaying exponentially with time, meaning that in this setting the signal propagates as a Gaussian spreading in time, with constant integral (an intuitive result from the fact that myelination effectively eliminates the leak current). This slowing down of the signal decay results in faster "velocity" of the propagating signal in the axon, which we can compute as above:

$$\begin{aligned} 0 &= \frac{\mathrm{d}}{\mathrm{d}t}\bigg|_{t^*}\left[-\frac{r_L c_m x^2}{2at} - \frac{1}{2}\log t + \text{const. w.r.t. } t\right] \\ &= \frac{r_L c_m x^2}{2at^{*2}} - \frac{1}{2t^*} \\ \Leftrightarrow t^* &= \frac{r_L c_m x^2}{a} \\ \Rightarrow v_{axon} &= \frac{a}{r_L c_m x} \end{aligned}$$

This looks like bad news: $v_{axon} \propto 1/x$ so long axons will have very slow signal propagation to their terminals. To deal with this, it turns out that in mammalian neural systems $a \propto L$. This means that for long and (therefore) thick myelinated axons,

PEL said this in lecture - reference??

$$v_{axon} = \frac{1}{r_L c_m} = \frac{2\pi a}{r_L C_m}$$

Thus, we have that (approximately)

$$\begin{aligned} v_{dendrite} &\propto \sqrt{a} \\ v_{axon} &\propto a \end{aligned}$$

For further discussion of such wiring principles at play in the mammalian brain, see Chklovskii et al., 2002.

Note, however, that the spatial decay of the signal remains the same in axons as in dendrites, since the Gaussians have the same width:

$$B = \frac{4\lambda^2}{\tau_m} = \frac{4r_m a}{2r_L r_m c_m} = \frac{2a}{r_L c_m} = D$$

So, although a signal originating from the soma may propagate faster down an axon, it will still decay to 0 for any distances much further than about $2\sqrt{Dt}$. Since axons need to be long to project to different brain areas, they deal with this problem by separating segments of myelination with so-called *nodes of Ranvier* where there is a high concentration of active Na^+ channels that can initiate an action potential if the membrane potential gets high enough. This is called *saltatory conductance*, since the action potential "jumps" (*salta*, in Spanish) from one node to the next.

2.8 Synaptic Transmission

Synaptic transmission at a spike-mediated chemical synapse begins when an action potential invades the presynaptic terminal and activates voltage- dependent Ca^{2+} channels, leading to a rise in the concentration of Ca^{2+} within the terminal. This causes vesicles containing transmitter molecules to fuse with the cell membrane and release their contents into the synaptic cleft between the pre- and postsynaptic sides of the synapse. The transmitter molecules then diffuse across the cleft and bind to receptors on the postsynaptic neuron. Binding of transmitter molecules leads to the opening of ion channels that modify the conductance of the postsynaptic neuron, completing the transmission of the signal from one neuron to the other. Postsynaptic ion channels can be activated directly by binding to the transmitter, or indirectly when the transmitter binds to a distinct

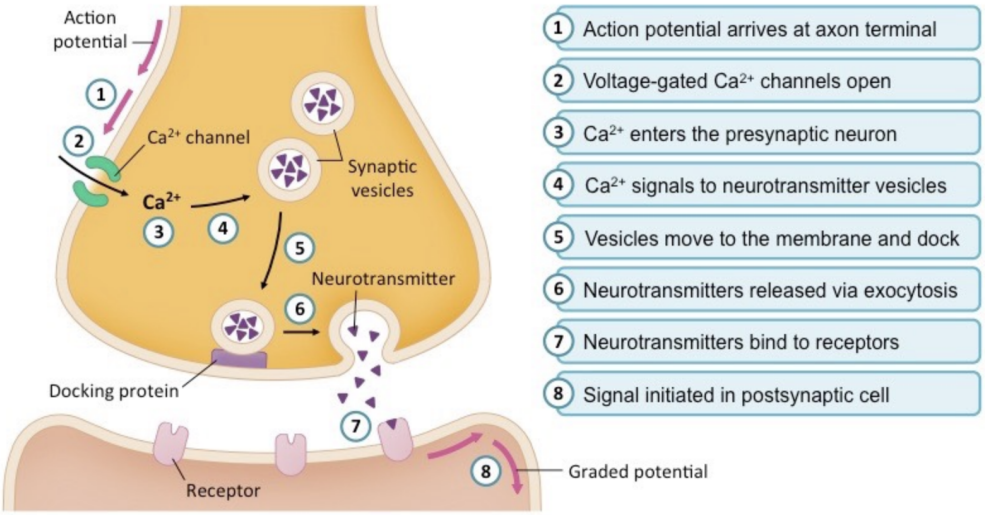


Figure 2.17: Visualization of synaptic transmission.

Neurotransmitter	Receptor	Time constant	Ions	Type
Glutamate	AMPA	fast ($\sim$ 1 ms)	cations	ionotropic
	NMDA	slow	cations, incl. Ca^{2+}	ionotropic
GABA	$GABA_A$	fast	Cl^- conductance	ionotropic
	$GABA_B$	flow	K^+ conductance	metabotropic

Table 1: Common neurotransmitters and receptor types.

receptor that affects ion channels through an intracellular second-messenger signaling pathway (direct quote from Dayan & Abbott). This process is visualized in Figure 2.17.

As with standard channel conductances, synaptic channel conductances can be modeled as the product of an average conductance term and an opening probability: $g_s = \bar{g}_s P$, where in this case

$$P = P_{rel} P_s. \tag{41}$$

Here, P_{rel} is the probability that a vesicle successfully releases neurotransmitter from the presynaptic terminal into the synaptic cleft (given the arrival of an action potential), and P_s is the probability that the postsynaptic receptor opens to receive the neurotransmitter. Release probability is governed by two quantities. One is the amount of calcium in the presynaptic terminal, with higher calcium implying higher release probability. The other is release itself: every time a vesicle is released, the probability of subsequent release drops; then it decays exponentially back to baseline (which is calcium dependent). The concentration of calcium in the presynaptic terminal is largely independent of the amount of neurotransmitter.

There are two broad classes of synaptic conductances. In *Ionotropic* receptors, the neurotransmitter binds to the channel directly and activates it, while in *metabotropic* receptors, the neurotransmitter binds to a separate receptor and activates the conductance through an intracellular signaling pathway. Ionotropic conductances activate and deactivate more rapidly than metabotropic receptors, while in addition to opening ion channels, metabotropic receptors can induce long-term changes within the post-synaptic neuron via mechanisms like G-protein-mediated receptors and second-messengers. Serotonin, dopamine, norepinephrine, and acetylcholine all act via metabotropic receptors.

Glutamate and GABA are the major excitatory and inhibitory transmitters in the brain, and both can act ionotropically and metabotropically. The main ionotropic receptor types for glutamate are called AMPA and NMDA. Both AMPA and NMDA receptors use mixtures of cations (positive ions, such as Ca^{2+}) and have reversal potentials around 0 mV. AMPA receptors activate and deactivate rapidly, while NMDA is slower, more permeable to Ca^{2+}, and has an usual voltage dependence. GABA activated two major inhibitory conductances in the brain. $GABA_A$ receptors produce fast ionotropic Cl^- conductance, while $GABA_B$ receptors are metabotropic and slower, producing a longer-lasting K^+ conductance.

In addition to chemical synapses, neurons can communicate via *gap junctions*, which produce a synaptic current proportional to the voltage difference at the two terminals.

2.9 Models of Synaptic Conductance

We have two important probabilities

1. With probability P_{rel}, the presynaptic action potential triggers the fusion of vesicles containing neurotransmitter to the cell membrane, leading to the release of the neurotransmitter into the synaptic cleft.
2. With probability p_j, neurotransmitter binds to type-j receptors on the post-synaptic cell membrane, causing type-j ion channels to open. If the resulting post-synaptic membrane current is strong enough (summing over all j at the synapse), the membrane potential may rise above threshold and trigger an action potential in the post-synaptic cell.

Our synaptic conductance-based model for the post-synaptic membrane current at synapse s then becomes:

$$i_s = \xi_s \sum_j \bar{g}_j p_j (V - E_j)$$

$$\xi_s = \begin{cases} 1 & \text{with probability } P_{rel} \\ 0 & \text{with probability } 1 - P_{rel} \end{cases}$$

where j indexes different neurotransmitter-dependent ion channel types on the post-synaptic membrane at the given synaptic cleft. Our notation follows Hodgkin-Huxley model conventions, with $\bar{g}_j$ (= conductance of open channel j × density of channel j) and E_j (= reversal potential of ions channel j is permeable to) as constants and $p_j(t)$ (= probability of a j-channel being open) modelled as a gating variable with (two-state Markov model) dynamics

$$\frac{\mathrm{d}p_j}{\mathrm{d}t} = \alpha_j(C_j)(1 - p_j) - \beta_j(C_j)p_j$$

$$\Leftrightarrow \tau_j(C_j)\frac{\mathrm{d}p_j}{\mathrm{d}t} = p_\infty^{(j)}(C_j) - p_j$$

$$\tau_j(C_j) = \frac{1}{\alpha_j(C_j) + \beta_j(C_j)}, \quad p_\infty^{(j)}(C_j) = \frac{\alpha_j(C_j)}{\alpha_j(C_j) + \beta_j(C_j)}$$

where C_j is the concentration (in the synaptic cleft) of the neurotransmitter that activates channel j and E_j designates the reversal potential for the ions that channel j is permeable to. α_j and β_j respectively refer to the rate of binding and unbinding of neurotransmitter to the given receptor type. Typically, at a given synapse there will only be one type of neurotransmitter being released by the pre-synaptic cell (the strongest version of *Dale's Law*: every neuron releases only one type of neurotransmitter at all its synaptic terminals), but I will continue with the general case.

Solving the equation above gives us

$$p_j(t) = p_\infty^{(j)}(C_j) + (p_j(0) - p_\infty^{(j)}(C_j))e^{-\frac{t}{\tau_j(C_j)}}$$

A useful simplification here is to assume β_j to be a small constant, and to set $\alpha_j(C_j) \propto C_j^k$ with some exponent k such that neurotransmitter binding rate is highly dependent on the concentration of neurotransmitter in the synaptic cleft, with $\alpha_j(0) = 0$. We then model the concentration of neurotransmitter $C_j(t)$ as a square wave

$$C_j(t) = \bar{C}_j \Theta(t)\Theta(T - t)$$

with large $\bar{C}_j$ so that $p_\infty^{(j)}(C_j) \approx 1$ at times $t \in [0, T]$. In reality, after neurotransmitter is released into the synaptic cleft, it is quickly removed via enzyme-mediated degradation as well as through diffusion, making the square wave a reasonable approximation. This results in the following solution:

$$p_j(t) = \begin{cases} 1 - (1 - p_j(0))e^{-(\alpha_j(\bar{C}_j) + \beta_j)t} & \text{if } 0 \le t \le T \\ p_j(T)e^{-\beta_j t} & \text{if } t > T \end{cases}$$

which consists of a saturating (to 1) rising exponential with time constant $\tau_{rise} = \frac{1}{\alpha_j(\bar{C}_j) + \beta_j}$ at times $t \in [0, T]$ followed by an exponential decay with time constant $\tau_{decay} = \frac{1}{\beta_j}$, where time $t = 0$ indicates the moment at which neurotransmitter is released into synaptic cleft.

A common further simplification is to assume instantaneous neurotransmitter release and removal by letting $T \to 0$ so that $C_j(t) \to \bar{C}_j \delta(t)$, and setting

$$p_j(0^+) = p_j(T) = p_j(0^-) + (1 - p_j(0^-))p_j^{max}$$

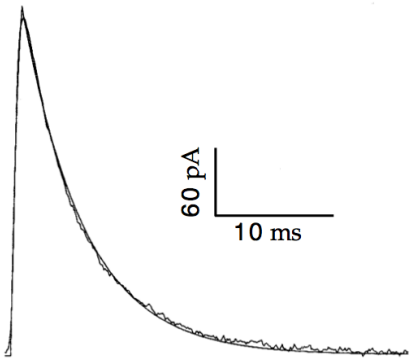


Figure 2.18: A pulse of neurotransmitter following by an exponential decay in channel opening probability.

, figure 2.18, where $p_j(0^+), p_j(0^-)$ are the probability of channel j being open at the exact moment of and just prior to neurotransmitter release, respectively. Here, we've set $p_j(0^+)$ to its maximum in the previous more realistic model, given by $p_j(T)$. In this case, $p_j^{max} = (1 - e^{-(\alpha_j(\bar{C}_j)+\beta_j)T})$. Generalizing this model to arbitrary pre-synaptic spike times $\{t_k\}$ gives us the following synaptic conductance dynamics[2]:

$$\frac{\mathrm{d}p_j}{\mathrm{d}t} = -\beta_j p_j + (1 - p_j)p_j^{max} \sum_k \xi_k \delta(t - t_k)$$

$$\xi_k = \begin{cases} 1 & \text{with probability } P_{rel} \\ 0 & \text{with probability } 1 - P_{rel} \end{cases}$$

Note that ξ_k, t_k are not indexed by j (in fact they should be indexed by the particular synapse s) - all receptors j at this synapse share the same pre-synaptic spike times and neurotransmitter release probability. In this case, we drop the ξ_s term in our equation for the post-synaptic membrane current i_s, since the stochastic vesicle release component is now implicit in the channel opening probabilities.

A more phenomenological model of the synaptic conductance is the difference-of-exponentials

$$p_j(t) = p_j^{max} B \left(e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_2}} \right)$$

$$B = \left(\left(\frac{\tau_2}{\tau_1} \right)^{\frac{\tau_{rise}}{\tau_1}} - \left(\frac{\tau_2}{\tau_1} \right)^{\frac{\tau_{rise}}{\tau_2}} \right)^{-1}, \quad \tau_1 > \tau_2$$

with rise time $\tau_{rise} = \frac{\tau_1 \tau_2}{\tau_1 - \tau_2}$ and decay time $\tau_{decay} = \tau_1$. The normalizer B simply enforces that the peak of the conductance be p_j^{max}, which occurs at $t^* = \tau_{rise} \log \frac{\tau_1}{\tau_2}$. Another more simplified phenomenlogical model is the α-function

$$p_j(t) = \frac{p_j^{max} t}{\tau_j} e^{1 - t/\tau_j}$$

which reaches its maximum p_j^{max} at $t^* = \tau_j$, with decay time $\tau_{decay} = \tau_j$. These two models are useful for neurotransmitter receptors with slower rise times (e.g. GABA_B, NMDA).

Note that under certain simplifications, we now have a full model of brain activity. Specifically, if we assume only one type of post-synaptic channel at each synapse and ignore the dynamics of dendrites and axons, the following equation gives us the membrane potential dynamics of any given neuron i:

$$\tau_m \frac{\mathrm{d}V_i}{\mathrm{d}t} = -(V_i - E_L) - [\text{HH active currents}] - \sum_j \bar{g}_{ij} \xi_{ij} p_{ij} (V_i - E_j)$$

$$\tau_{ij}(C_{ij}) \frac{\mathrm{d}p_{ij}}{\mathrm{d}t} = p_\infty^{(ij)}(C_{ij}) - p_{ij} \quad \text{[insert favorite synaptic conductance model here]}$$

$$\xi_{ij} = \begin{cases} 1 & \text{with probability } P_{rel}^{(ij)} \\ 0 & \text{with probability } 1 - P_{rel}^{(ij)} \end{cases}$$

[2] As hinted just above, this simple looking differential equation hides an analytical obstacle in that it requires evaluating $p_j(t_k)$, which is impossible to evaluate since p_j is in the middle of an infinite slope jump at this point because of the contribution of the δ-function $\delta(t - t_k)$. My derivation implies circumventing this problem by simply evaluating $p_j(t_k)$ at time t_k^-, just before the jump. This is called an Itô integral. It is worth noting, however, that an alternative approach would be evaluating $p_j(t_k)$ in the middle of the jump, called a Stratonovich integral.

where j indexes all synapses ij onto neuron i. This is a conductance-based model of the brain, since we are explicitly modelling the conductances in $\bar{g}_{ij}\xi_{ij}p_{ij}$. Alternatively (as we will do below in section 3), we could simplify this to a current-based model by absorbing $\bar{g}_{ij}\xi_{ij}(V_i - E_j)$ into one term W_{ij} such that the synaptic inputs to model i (i.e. the last term in the equation for $\frac{\mathrm{d}V_i}{\mathrm{d}t}$) are modelled as input currents.

2.9.1 Short-term Synaptic Plasticity: Synaptic Depression and Facilitation

The current outlook on synaptic plasticity is that, at a synapse ij, the maximal conductance term $\bar{g}_{ij}$ changes on long timescales (e.g. by increasing/decreasing density of receptors) whereas the neurotransmitter release probability $P_{rel}^{(ij)}$ change on both short and long timescales. On short timescales one of two things can happen:

- *Synaptic depression:* post-synaptic potential temporarily decreases with repeated high frequency pre-synaptic spikes, since the stock of readily available neurotransmitter in the pre-synaptic axon terminal has been depleted, thus lowering the probability of vesicle release on the next spike.
- *Synaptic facilitation:* post-synaptic potential temporarily increases with repeated high frequency pre-synaptic spikes, since this leads to a high influx of calcium Ca^{2+} ions into the pre-synaptic axon terminal, thus increasing the probability of vesicle release on the next spike.

We can thus model both synaptic depression and facilitation with a two-dimensional system of ODEs explicitly modelling the dynamics of calcium ion concentration $[\mathrm{Ca}^{2+}]$ in the pre-synaptic terminal and the number of vesicles M ready for release:

$$\begin{aligned} P_{rel} &= \mathrm{f}([\mathrm{Ca}^{2+}], M) \\ \tau_{Ca}\frac{\mathrm{d}[\mathrm{Ca}^{2+}]}{\mathrm{d}t} &= -[\mathrm{Ca}^{2+}] + \alpha\sum_k \delta(t - t_k) \\ \tau_M\frac{\mathrm{d}M}{\mathrm{d}t} &= M_0 - M - \sum_k \xi_k\delta(t - t_k) \end{aligned}$$

with some complicated $\mathrm{f}([\mathrm{Ca}^{2+}], M)$ that is presumably monotonically increasing in $[\mathrm{Ca}^{2+}], M$.

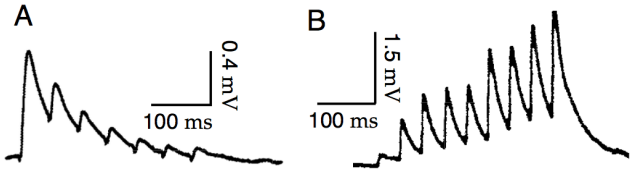


Figure 2.19: Post-synaptic voltage traces for short-term depression (A) and facilitation (B).

An alternative is to abstract away and forget the calcium and vesicle number dynamics by directly modelling the temporal dynamics of P_{rel}:

$$\tau_{rel}\frac{\mathrm{d}P_{rel}}{\mathrm{d}t} = P_0 - P_{rel} + \tau_{rel}\sum_k \delta(t - t_k) \times \begin{cases} -\xi_k(1 - f_D)P_{rel} \\ f_F(1 - P_{rel}) \end{cases}$$

where $f_F, f_D \in [0, 1]$ (larger f_F for stronger facilitation, smaller f_D for stronger depression), k indexes pre-synaptic spikes, and ξ_k is our usual stochastic variable taking on 1 with probability $P_{rel}(t)$ and 0 otherwise (representing whether or not vesicles were released upon pre-synaptic spike k). In other words, the release probability decays exponentially to its equilibrium value P_0, updating itself every time a pre-synaptic action potential arrives at the axon terminal according to either a synaptic depression or facilitation update rule. Because there is a P_{rel} term multiplied by δ-functions on the right hand side, analyzing this equation becomes a mess (see footnote 2), so it is actually easier work with the explicit update rules:

$$\begin{aligned} \tau_{rel}\frac{\mathrm{d}P_{rel}}{\mathrm{d}t} &= P_0 - P_{rel} \\ P_{rel} &\to \xi_k f_D P_{rel} + (1 - \xi_k)P_{rel} && \text{[synaptic depression]} \\ P_{rel} &\to P_{rel} + f_F(1 - P_{rel}) && \text{[synaptic facilitation]} \end{aligned}$$

where the updates occur upon a pre-synaptic spike arriving at the axon terminal.

Synaptic depression can be particularly useful for normalizing synaptic inputs and for detecting changes in firing rate. We can see this by setting $\xi_k = 1$ to allow perfectly reliable vesicle fusion and neurotransmitter

release on every pre-synaptic spike and then computing the steady state $\langle P_{rel} \rangle$ averaged over pre-synaptic spikes drawn from some homogenous Poisson process with rate r. Suppose the release probability is at this average steady state, $P_{rel} = \langle P_{rel} \rangle$, when a single pre-synaptic spike occurs at time t_k so that:

$$P_{rel} \to f_D \langle P_{rel} \rangle$$

Solving our ODE, the release probability at the time of the next spike t_{k+1} is then given by

$$P_{rel}(t_{k+1}) = P_0 + (f_D \langle P_{rel} \rangle - P_0) e^{-\frac{t_{k+1} - t_k}{\tau_{rel}}}$$

Having defined $\langle P_{rel} \rangle$ as the average steady state, averaging over spike times t_{k+1} should give us $\langle P_{rel}(t_{k+1}) \rangle = \langle P_{rel} \rangle$. In other words, the release probability should on average decay to $\langle P_{rel} \rangle$ by the time the next spike arrives (by definition). Given that the pre-synaptic spike times are drawn from a homogenous Poisson process, we can directly compute this average by averaging over the exponentially distributed inter-spike intervals:

$$\begin{aligned}
\langle P_{rel}(t_{k+1}) \rangle &= P_0 + (f_D \langle P_{rel} \rangle - P_0) \int_{t_k}^{\infty} P(t_{k+1} - t_k) e^{-\frac{t_{k+1} - t_k}{\tau_{rel}}} \mathrm{d}t_{k+1} \\
&= P_0 + (f_D \langle P_{rel} \rangle - P_0) \int_0^{\infty} r e^{-r\tau} e^{-\frac{\tau}{\tau_{rel}}} \mathrm{d}\tau \\
&= P_0 + (f_D \langle P_{rel} \rangle - P_0) r \int_0^{\infty} e^{-\tau \left(\frac{r\tau_{rel} + 1}{\tau_{rel}} \right)} \mathrm{d}\tau \\
&= P_0 + (f_D \langle P_{rel} \rangle - P_0) \frac{r\tau_{rel}}{r\tau_{rel} + 1}
\end{aligned}$$

Setting $\langle P_{rel}(t_{k+1}) \rangle = \langle P_{rel} \rangle$, we then solve for the average steady state $\langle P_{rel} \rangle$:

$$\langle P_{rel} \rangle = \frac{P_0 \left(1 - \frac{r\tau_{rel}}{r\tau_{rel}+1} \right)}{1 - f_D \frac{r\tau_{rel}}{r\tau_{rel}+1}} = \frac{P_0}{(1 - f_D) r\tau_{rel} + 1}$$

We thus see that, at high pre-synaptic firing rates r, the release probability scales with $\frac{1}{r}$. This means that the rate of arriving post-synaptic potentials, given by rP_{rel}, remains approximately constant with respect to the pre-synaptic firing rate r (at steady state). Synaptic depression thus serves as a mechanism for normalizing pre-synaptic inputs (with potentially different firing rates) on different synapses to the same synaptic transmission rate (and hence the same time-averaged post-synaptic potential amplitude, assuming individual post-synaptic potential amplitudes to be the same across synapses), at least in the regime of high pre-synaptic firing.

I have never like this argument... Couldn't the prob decrease on average from the average value in this setting, but increase in others in a way that balances out?

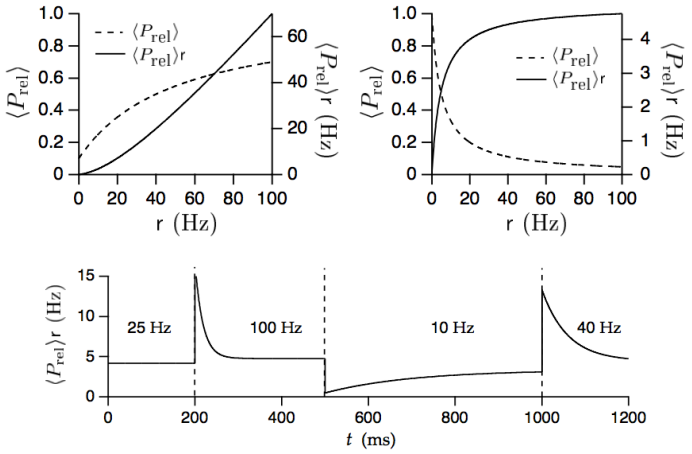


Figure 2.20: (Top) Visualization of the normalization effect of synaptic depression—as the firing rate r increases, $\langle P_{rel} \rangle$ drops proportionally. (Bottom) Depiction of the effect of transient increases in firing rate.

Of course, this also means that the synapse cannot convey any information about smooth changes in pre-synaptic firing rate, on the timescale of τ_{rel}. Faster changes $r \to r + \Delta r$, however, will be detected since it takes $\mathcal{O}(\tau_{rel})$ time for P_{rel} to reach its new steady state. Before reaching it, the synaptic transmission rate will thus transiently rise to

$$(r + \Delta r) \langle P_{rel} \rangle = \frac{(r + \Delta r) P_0}{(1 - f_D) r\tau_{rel} + 1}$$

which, in the limit of high pre-synaptic firing rates, is $\mathcal{O}(\frac{r + \Delta r}{r})$, figure 2.20. In other words, the resulting increase in synaptic transmission rate is proportional to the relative, rather than absolute, increase in pre-synaptic firing rate. A synapse can therefore use synaptic depression to encode the relative magnitude of transient changes in the pre-synaptic firing rate.

2.9.2 NMDA-mediated plasticity

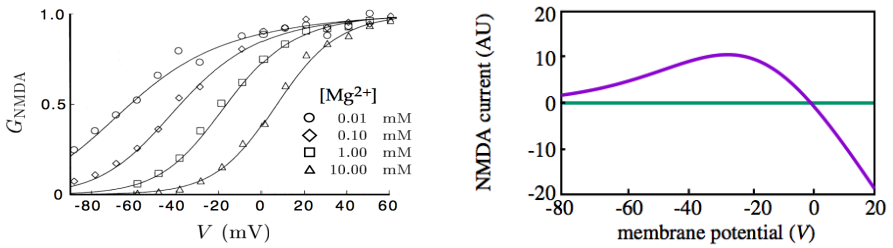


Figure 2.21: NMDA channel conductance (left) and current (right) as a function of voltage.

One of the few postulated mechanisms for long-term plasticity (section 5) is the unblocking of NMDA receptors via back-propagating action potentials. NMDA receptors are unique in that they have sites that magnesium Mg^{2+} ions bind to, thus blocking the receptor in a voltage-dependent fashion. Namely, since Mg^{2+} ions have a positive charge, a high membrane potential at the post-synaptic cell/dendrite will repel them, thus unblocking the NMDA receptors at the synapse. We can thus model NMDA receptor conductance by incorporating a scaling factor that is sigmoidal in the membrane potential V, this sigmoidal relationship mediated by concentration of magnesium ions in the synaptic cleft $[Mg^{2+}]$:

$$i_{\text{NMDA}} = -\frac{\bar{g}_{\text{NMDA}} p_{\text{NMDA}}}{1 + \frac{[Mg^{2+}]}{3.57} e^{-\frac{V}{16.8}}} (V - E_{\text{NMDA}})$$

(where the vesicle release probability is implicit in p_{NMDA}). Visualised in 2.21.

Importantly, it turns out that when a neuron spikes, action potentials often "back-propagate" up the dendrites, thus increasing the membrane potential near the synapses and unblocking NMDA receptors. Furthermore, NMDA receptors are permeable to Ca^{2+} ions, which trigger long-term changes at the synapse by signalling the cell to (1) open more NMDA channels and (2) produce and insert new AMPA channels. This process is called NMDA-mediated plasticity. Note that NMDA-mediated plasticity at a synpase can also be triggered by increases in the membrane potential resulting from the depolarization of neighboring synapses, leading to what is called *heterosynaptic plasticity* (more on the functional implications of this in section 5).

Because it requires both pre- and post-synaptic activity, NMDA-dependent plasticity is thought to play a major role in Hebbian learning. Indeed, it has been experimentally observed that in many cases synaptic plasticity stops when NMDA channels are blocked. From a neurocomputational point of view, NMDA receptors can act as coincidence detectors, since they are most active under simultaneous post- and pre- synaptic depolarization.

2.10 Spike-Timing Dependent Plasticity (STDP)

Another classical experimental finding is *spike-timing-dependent plasticity (STDP)* (Markram et al, 1997; Bi & Poo, 1998), whereby synaptic plasticity is only induced when action potentials in the pre- and post- synaptic cells occur within $\sim$ 50ms of each other, the magnitude of the plasticity decaying with increasing latency. If the post-synaptic spike occurs after the pre-synaptic spike, LTP occurs; if, on the other hand, the post-synaptic spike precedes the pre-synaptic spike, then LTD occurs, figure 2.22.

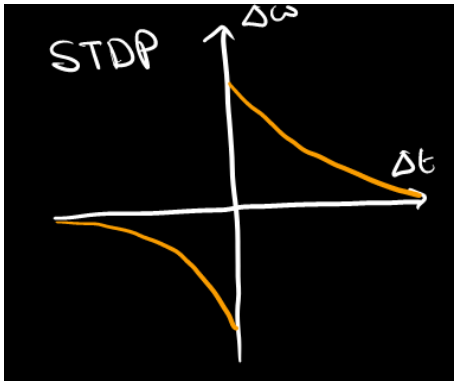


Figure 2.22: Classic shape of STDP curve, shows the change in synaptic weight after the occurence of a pair of spikes, one presynaptic and the other postsynaptic separated by Δt.

This can be easily implemented in spiking networks. In a continuous activity network like those we have been considering in this section, we can approximate it using a function $H(\tau)$ that gives the weight change when $t_{post} - t_{pre} = \tau$, t_{post}, t_{pre} being adjacent post- and pre- synaptic spike times (e.g. the line in fig. 8.2B):

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \int_0^\infty \mathrm{d}\tau H(\tau) v(t) \mathbf{u}(t-\tau) + H(-\tau) v(t-\tau) \mathbf{u}(t)$$

where $sign(H(\tau)) = sign(\tau)$ such that the first and second terms inside the integral induce LTP and LTD respectively.

Like the basic Hebb rule, the STDP learning rule is unstable. Even balancing the positive and negative effects isn't a good permanent solution, it might balance temporarily but a slight increase in input firing rate will unbalance it, leading to exploding or 0 weights. A fix is to make potentiation constant but to change the depression to being multiplicative. This leads to a roughly log-normal distribution of weights that matches biology, and the multiplicative depression has some experimental backing [Van Rossum et al., 2000].

Vanilla STDP implements competition between weights: an increase in w_i makes it easier for an increase in u_i to lead to higher v regardless of the state of the other inputs $u_{j\neq i}$, thus possibly increasing $v(t-\tau)u_{j\neq i}(t)$ and inducing LTD at those synapses. This tends to lead to a highly bimodal distribution of feed-forward weights.

Interestingly, the STDP rule can lead to invariant responses. We can approximately solve the above differential equation by ignoring the changes in $\mathbf{v}$ over time caused by the changes in $\mathbf{w}$:

$$\mathbf{w} = \frac{1}{\tau_w} \int_0^T \mathrm{d}t v(t) \int_{-\infty}^\infty \mathrm{d}\tau H(\tau) \mathbf{u}(t-\tau)$$

where we have also assumed $\mathbf{w}(0) = 0$ and ignored small contributions from the end points of the integral. In this approximation, our final learned $\mathbf{w}$ depends on the temporal correlation between the post-synaptic activity $v(t)$ and the pre-synaptic activity $\mathbf{u}(t)$ temporally filtered by the STDP kernel $H(\tau)$. Consider now the scenario of $\mathbf{u}(t)$ arising from an object moving across the visual field. If the filter $H(\tau)$ filters the resulting sequence of inputs over the amount of time the object is present, then it will strengthen the synapses from all pre-synaptic cells responding to the object while it moves, regardless of its position. In the long run, the resulting weights will thus lead to post-synaptic responses independent of the position of the object, producing position-invariant responses to the object such as those seen in inferotemporal cortex (IT).

STDP can also produce predictive coding responses in a recurrent network with fixed feedforward weights. Consider a set of post-synaptic neurons with equally spaced homogenous tuning curves (e.g. for orientation) and an input stimulus that traverses the stimulus space in the same direction on each presentation (e.g. a clockwise rotating bar). As the stimulus is repeated, the tuning curves will then gradually shift in the opposite direction, since each neuron's recurrent input from neurons selective for the previous stimulus state will be strengthened. On each subsequent presentation, then, a given post-synaptic neuron is more and more likely of firing earlier and earlier. In the long run, this will produce predictive responses anticipating subsequent input according to the input stimulus it was trained on. Such behavior is observed in hippocampal place cells (D&A pgs 312-3).

2.11 Models of Synaptic Plasticity

2.11.1 Models of NMDA-mediated plasticity

In this model the weight change is some function of the calcium concentration inside the postsynaptic terminal; and calcium only enters when there is coincident presynaptic spiking and postsynaptic depolarisation. We don't know what this function looks like but we do know it must have both positive and negative areas, else all weights will tend to either zero of infinity. To achieve this we set a threshold calcium concentration θ below which the weight decreases, figure 2.23. As can be seen in the figure, this simple setup creates some nice properties, weight increases in response to nearby spikes and decreases else. But there is no setting of the threshold which performs anything like STDP with its precise dependence on the ordering of spikes. For that we will need a more complex model.

2.11.2 Model of Graupner & Brunel

This model makes a some different assumptions, outlined below and in figure 2.24:

- Each postsynaptic spike leads to a constant amount of calcium influx, and each presynaptic spike does the same but with a delay. Calcium concentration is then the sum of all these spike induced calcium influxes. This at odds with our biophysical understanding (the calcium influx shouldn't be concentration independent...), but ah well.

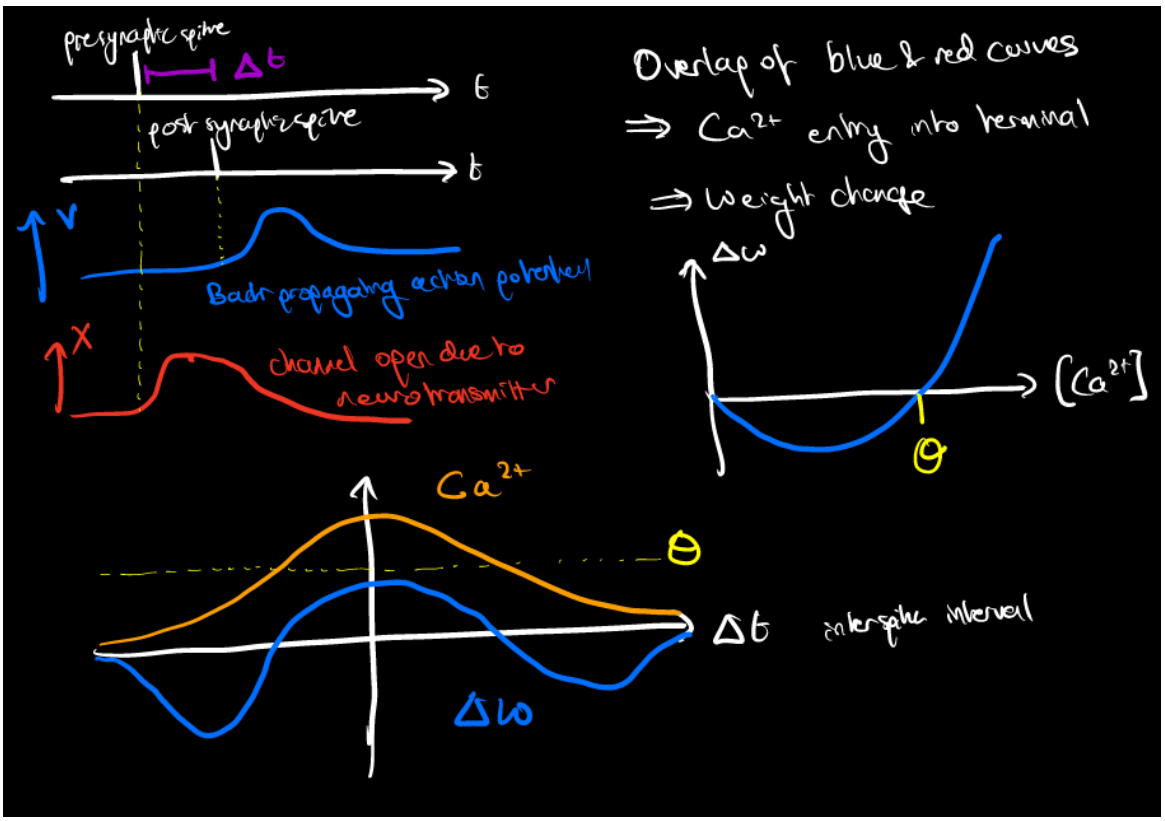


Figure 2.23: Top Left: A presynaptic and postsynaptic spike occur separated by time Δt. The presynaptic spike causes neurotransmitter release that opens the postsynaptic channels, X, and the postsynaptic spike causes a backpropagating action potential that increases the membrane voltage, removing the magnesium blocker. When the two of these happen together calcium enters the cell. This caues AMPA channels to be inserted, hence changing the weight according to some curve. The peak calcium concentration and resulting weight change are shown as a function of time between spikes in the bottom plot. There is no way in this model to produce something like STDP.

- The synapse is considered as a binary system. This is inspired by a chemical called CaM Kinase II that undergoes conformational changes based on the calcium concentration. In its high state AMPA channels are inserted, and in the low state they are removed.
- The transition rates between conformational states are determined by the amount of time the synapse has spent above two thresholds, rather than by the calcium concentration directly.

This gives huge flexibility to the model allowing it to capture many of the behaviours observed in real neurons [Graupner and Brunel, 2012].

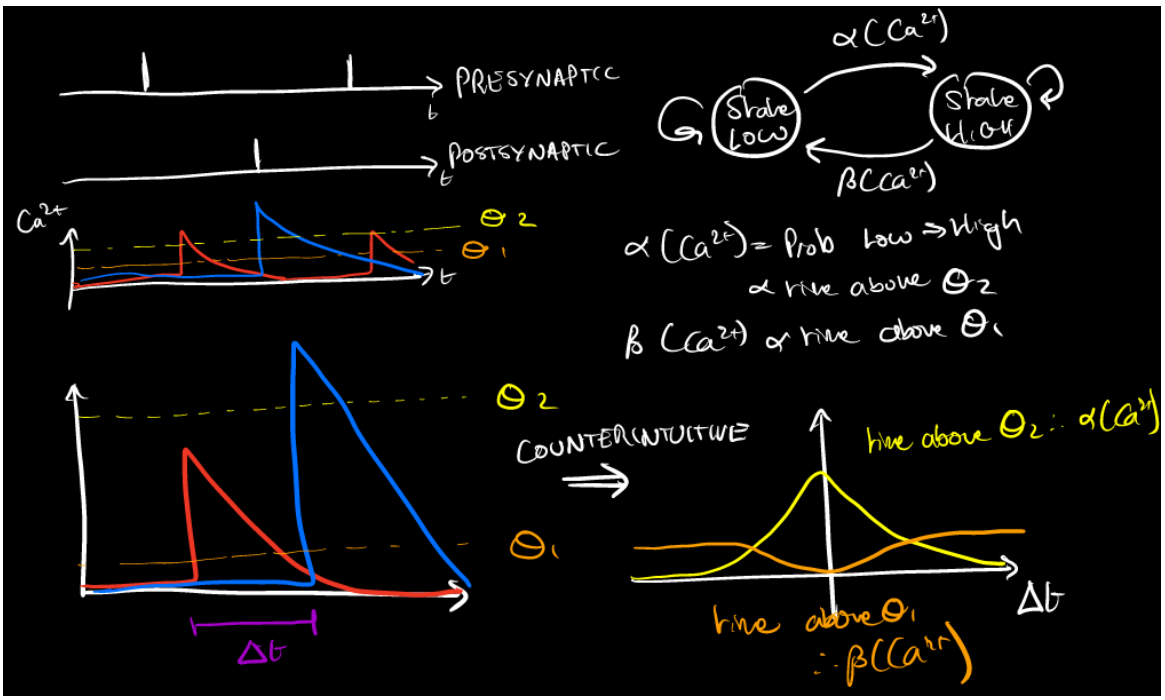


Figure 2.24: Top Left: strings of pre and pos synaptic spikes cause stereotyped patterns of caclium concentration change that sum. Top Right: the synapse is a binary model, and the transition probabilities are controlled by the amount of time the calcium concentration spends above the two thresholds. This can lead to some counterintutive effects, for example if θ_2 is high and θ_1 low then one transition probability can increase

3 Random Networks

In this section we analyse a very rough model of neuronal networks and show that it can provide excellent intuition. In fact, it will tie together the following 8 observations:

1. There is a broad distribution of firing rates, somewhat log-normal.
2. Most neurons have low firing rates, relative to their maximum. ($2Hz$).
3. You see irregular activity (Poisson-like spiking).

   We don't actually link this one here, though see section 3.3, but I believe irregular firing is intimately linked to low firing rates and $\frac{1}{\sqrt{K}}$ weight scaling in some non-trivial way I don't understand. Something to add in the future...
4. Often see oscillations in mean activity, fig 3.1A.
5. During oscillations excitatory neurons leads inhibitory, fig 3.1A.
6. Some areas show up-down states, average firing rates are either very high or low, fig 3.1B. Particularly common under anaesthesia and last on the order of seconds.
7. There are bumps in the average activity, fig 3.1C.
8. A section of brain can switch between switch, bump, irregular, and oscillatory behaviour

3.1 Mean-Field Analysis of Spiking Networks

Consider the following simplified network model:

$$\tau_m \frac{\mathrm{d}V_i}{\mathrm{d}t} = f_i(V_i, t) - \sum_{j \neq i} m_{ij} g_j(t)(V_i - E_j) \tag{42}$$

$$\tau_s \frac{\mathrm{d}g_j}{\mathrm{d}t} = -g_j + \sum_k \delta(t - t_k^{(j)}) \tag{43}$$

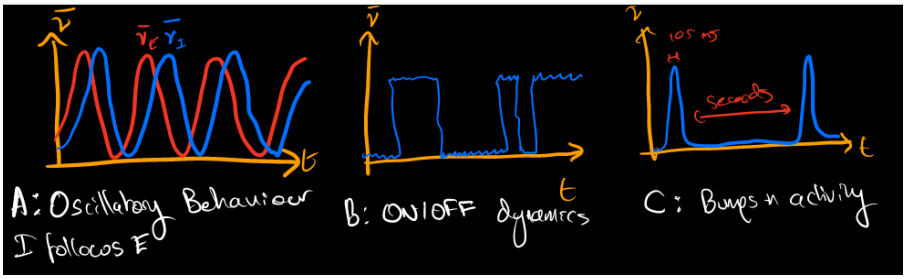


Figure 3.1: Cartoon showing qualitative behaviour of average network activity in different regimes.

where $f_i(V_i, t)$ represents the single neuron dynamics (e.g. leak current, Hodgkin-Huxley currents) and m_{ij} is an abstract variable including contributions from neurotransmitter release probability, the ion channel density, and the open ion channel conductance (i.e. combination of $\xi_j \sim P_{rel}$ and $\bar{g}_i$) across all synapses between pre-synaptic neuron j and post-synaptic neuron i. The pre-synaptic neuron j is assumed to release the same neurotransmitter at all its axon terminals, with associated reversal potential E_j (+ve for excitatory, -ve for inhibitory). The dynamics of g_j are meant to emulate the dynamics of a synaptic conductance, under the simplification that each pre-synaptic spike at time $t_k^{(j)}$ instantly triggers neurotransmitter release at all its axon terminals, leading to an instant rise in the neurotransmitter concentration at the synaptic cleft, modelled by a δ-function in equation 43.

This is a set of $2N$ dimensional dynamical system that we have reason to believe could be a good vanilla description of N neurons in the brain. We then ask: what kinds of dynamics emerge from such a system? Unfortunately, the non-linear terms $f_i(V_i, t)$ and $g_j(t)V_i(t)$ in equation 42 make this system very hard to analyze, so we approximate, and hope we're not throwing out the baby with the bathwater.

First, we get rid of the difficult non-linear interaction $g_j(t)V_i(t)$ between the membrane potential and synaptic conductance by moving from a *conductance-based model* to a *current-based model*. We approximate the post-synaptic membrane potential in this interaction term by its temporal mean: $g_j(t)V_i(t) \to g_j(t)\bar{V}_i$, allowing us to rewrite equation 42 as

$$\tau_m \frac{\mathrm{d}V_i}{\mathrm{d}t} = f_i(V_i, t) + \sum_{j \neq i} w_{ij} g_j(t)$$

where $w_{ij} = -m_{ij}(\bar{V}_i - E_j)$ is an approximate "synaptic weight".

This gives us the *synaptic drive*:

$$h_i(t) = \sum_j w_{ij} g_j(t)$$

(henceforth all sums over pre-synaptic neurons $j \neq i$ will be written shorthand as sums over j), which is a function of time by virtue of the temporal dynamics of $g_j(t)$ (equation 43). Note that we've defined these dynamics such that the total current contribution of a single pre-synaptic spike k from neuron j integrates to 1:

$$g_j(t) = \Theta(t - t_k^{(j)}) \frac{1}{\tau_s} e^{-\frac{-(t-t_k)}{\tau_s}}$$

$$\Rightarrow \int_0^\infty g_j(t)\mathrm{d}t = \int_{t_k^{(j)}}^\infty \frac{1}{\tau_s} e^{-\frac{-(t-t_k)}{\tau_s}} \mathrm{d}t = \left[-e^{-\frac{-(t-t_k)}{\tau_s}}\right]_{t_k^{(j)}}^\infty = 1$$

Thus, its temporal mean $\bar{g}_j = \langle g_j(t) \rangle_t$ is in fact the mean firing rate ν_j of neuron j:

$$\begin{aligned}
\langle g_j(t) \rangle_t &= \lim_{T \to \infty} \frac{1}{T} \int_0^T g_j(t)\mathrm{d}t \\
&= \lim_{T \to \infty} \frac{1}{T} \sum_k \int_{t_k^{(j)}}^\infty \frac{1}{\tau_s} e^{-\frac{-(t-t_k)}{\tau_s}} \mathrm{d}t \\
&= \lim_{T \to \infty} \frac{1}{T} n_j(T) \equiv \nu_j
\end{aligned}$$

where $n_j(T)$ designates the number of emitted spikes up until time T. We can thus rewrite the synaptic drive

to neuron i in terms of mean firing rates:

$$\begin{aligned} h_i(t) &= \sum_j w_{ij} g_j(t) \\ &= \sum_j w_{ij}(\bar{g}_j + \delta g_j(t)) \\ &= \underbrace{\sum_j w_{ij}\nu_j}_{\text{“quenched noise”}} + \underbrace{\sum_j w_{ij}\delta g_j(t)}_{\text{“dynamic noise”}} \\ &= \quad \bar{h}_i \quad + \quad \delta h_i(t) \end{aligned}$$

where $\delta g_j(t) \equiv g_j(t) - \bar{g}_j$ are 0-mean fluctuations of each $g_j(t)$ around its temporal mean $\bar{g}_j$, such that

$$\langle \delta h_i(t) \rangle_t = \sum_j w_{ij} \langle \delta g_j(t) \rangle_t = 0$$

We can thus express the synaptic drive $h_i(t)$ as 0-mean temporal fluctuations $\delta h_i(t)$ (the “dynamic noise”) around a temporal mean $\bar{h}_i$ (the “quenched noise” - noisy over neurons rather than over time).

This inspires the following idea: rather than analyzing actual trajectories of our high-dimensional non-linear system, let's try to solve the following two possibly easier *statistical* problems:

1. characterise the distribution of *time-averaged* activity over all neurons in the network (arising from the quenched noise component of the synaptic drive)
2. characterise temporal correlations in activity (arising from the dynamic noise component of the synaptic drive)

Solving these problems naturally won't give us temporal trajectories or anything like what we'd get from solving the ODEs, but it will still prove useful to gain some insight into the different dynamical regimes such a system can exhibit.

Here, we'll focus mainly on problem #1 by simply computing the moments of the distribution: $\langle \nu^\ell \rangle$. Our approach to doing this stems from the following two observations:

I. the time-averaged synaptic drive $\bar{h}_i = \sum_j w_{ij}\nu_j$ depends on the distribution of time-averaged firing rates

II. the time-averaged firing rate ν_i of a neuron should depend on its time-averaged synaptic drive $\bar{h}_i$

These two observations respectively suggest that we should be able to write down a set of *self-consistent* equations expressing

i. the moments of the synaptic drive $\langle \bar{h}^\ell \rangle$ in terms of the moments of the firing rate $\langle \nu^\ell \rangle$

ii. the moments of the firing rate $\langle \nu^\ell \rangle$ in terms of the moments of the synaptic drive $\langle \bar{h}^\ell \rangle$

Thus, for computing L moments, we should be able to write down $2L$ equations with $2L$ unknowns - a system of equations we can (at least in principle) solve. This will require a few more assumptions along the way (namely, $N \to \infty$, i.i.d. weights, and a saturating gain function), but it will turn out to yield some interesting results.

We start with writing down equations expressing the moments of the distribution over time-averaged synaptic drives in terms of the moments of the distribution over time-averaged firing rates. We first note that the synaptic drive is a big sum of $N-1$ terms:

$$\bar{h}_i = \sum_j w_{ij}\nu_j$$

Thus, if each of the terms inside the sum are independently distributed over index i, as $N \to \infty$ (the regime we're interested in, since there's lots of neurons in the brain) the Central Limit Theorem tells us that the distribution of $\bar{h}_i$ over index i (i.e. over the population of N neurons) becomes Gaussian. Noting that the only terms on the right-hand-side that vary over index i are the synaptic weights, we conclude that if the synaptic weights are independently sampled then the distribution over time-averaged synaptic drives in the population will approach a Gaussian in the large N limit. For analytical convenience, we therefore incorporate the following structural constraint into our model:

the synaptic weights w_{ij} are independent and identically distributed (i.i.d.) with mean $\bar{w}$ and variance σ_w^2

While the assumption of i.i.d. weights may not hold in the brain, it might still allow our model to provide useful insights into how actual neural circuits operate[3]. following We can now easily take the limit of large N and invoke the CLT to write:

$$\bar{h}_i = \mu_h + \sqrt{N}\sigma_h \xi_i \quad \xi_i \sim \mathcal{N}(0,1)$$

where

$$\mu_h = \langle \bar{h}_i \rangle_i = \sum_j \langle w_{ij} \rangle_i \nu_j = \bar{w} \sum_j \nu_j = N\bar{w}\bar{\nu} \tag{44a}$$

$$\mathrm{Var}_i[\bar{h}_i] = \sum_j \mathrm{Var}_i[w_{ij}]\nu_j^2 = \sigma_w^2 \sum_j \nu_j^2 = N\sigma_w^2 \overline{\nu^2} \equiv N\sigma_h^2 \tag{44b}$$

where I have defined the ℓth moment of the firing rates as

$$\overline{\nu^\ell} \equiv \frac{1}{N}\sum_j \nu_j^\ell$$

and the i subscript on the expectation and variance operators indicates an expectation over the random variation along index i. We also used the fact that, because the w_{ij}'s are identically distributed, $\forall j, k \ \langle w_{ij} \rangle_i = \langle w_{ik} \rangle_i = \bar{w}$ (and similarly for the second moment and therefore the variance).

Having written down a whole distribution over time-averaged synaptic drives $\bar{h}_i$ in terms of the first and second moments of the time-averaged firing rates ν_i, we now turn to the problem of expressing the latter in terms of the former. This becomes easy once we have a way of transforming the time-averaged synaptic drive to a neuron into its time-averaged firing rate:

$$\nu_i = \phi_i\left(\bar{h}_i\right)$$

where the so-called "gain function" $\phi_i(\cdot)$ is some non-linear function depending on the single neuron dynamics $f_i(V_i, t)$ of neuron i. Generally, we'll take $\phi_i(\cdot) = \phi(\cdot)$ to simply be a scaled sigmoid saturating at $\nu_{max} \sim 100$Hz, reflecting the fact that firing rates cannot be negative and neurons can only fire at a finite rate. Further justification for the sigmoid shape is provided by [Wilson and Cowan, 1972]: the time-averaged firing rate should be directly proportional to the probability of having suprathreshold input per unit time, approximately equal to the cumulative probability density of $\bar{h}_i$ from spiking threshold to infinity. If $\bar{h}_i$ has a unimodal distribution (which, as we just showed, it does under the assumption of i.i.d. weights), then this cumulative distribution will be a sigmoid[4]. Moreover, the sigmoid captures the main ingredients of what a neuronal gain function should look like: positive, monotonically increasing, and saturating. That said, most of the subsequent analysis is agnostic as to what the form of the gain function is, so any reasonable saturating function should do.

With this gain function in hand, we can easily express any moment of ν as a function of the mean μ_h and variance $N\sigma_h^2$ of the time-averaged synaptic drives $\bar{h}_i$. In the limit of $N \to \infty$, the law of large numbers give us:

$$\overline{\nu^\ell} = \lim_{N\to\infty} \frac{1}{N}\sum_i \phi\left(\bar{h}_i\right)^\ell = \int \phi(\bar{h})^\ell P(\bar{h})\, \mathrm{d}\bar{h} = \int \phi(\mu_h + \sqrt{N}\sigma_h \xi)^\ell \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}}\, \mathrm{d}\xi \tag{45}$$

Putting equations 44 and 45 together gives us the following set of four self-consistent equations describing the mean and variance of $\bar{h}_i$ and ν_i over the population:

$$\begin{aligned}
\mu_h &= N\bar{w}\bar{\nu} \\
\sigma_h^2 &= \sigma_w^2 \overline{\nu^2} \\
\bar{\nu} &= \int \phi(\mu_h + \sqrt{N}\sigma_h \xi) \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}}\, \mathrm{d}\xi \\
\overline{\nu^2} &= \int \phi(\mu_h + \sqrt{N}\sigma_h \xi)^2 \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}}\, \mathrm{d}\xi
\end{aligned}$$

3.1.1 Observation 1: Firing Rate Distribution

We are now in a position to discuss our first neural observation: that there is a broad distribution of firing rates. First, consider the case of $\mathrm{Var}[\bar{h}_i] \to 0$. In this limit, $\overline{\nu^\ell} \to \phi^\ell(\mu_h)$, such that $\overline{\nu^2} = \bar{\nu}^2$ and $\mathrm{Var}[\nu] = 0$, meaning that

$$P(\nu) \to \delta(\nu - \bar{\nu})$$

[3]Recall the famous UCL graduate statistician George Box: "all models are wrong; some models are useful"

[4]If $\bar{h}_i$ had a multimodal distribution, then $\phi(\cdot)$ would look like a stack of sigmoids, with an inflection point at each mode.

On the other hand, as $\text{Var}[\bar{h}_i] \to \infty$, $\nu_i = \phi(\bar{h}_i) \to \{0, \nu_{max}\}$, so that

$$P(\nu) \to \frac{1}{2}\delta(\nu) + \frac{1}{2}\delta(\nu - \nu_{max})$$

where ν_{max} is the maximum of the saturating gain function $\phi(\cdot)$ (it is easy to verify that this distribution has the exact moments given by equation 45, namely $\overline{\nu^\ell} = \frac{1}{2}\nu_{max}^\ell$). Given that actual observed firing rates are variable across neurons and generally lie somewhere between 0 and their maximum, these two results tell us that $\text{Var}[\bar{h}_i]$ should be greater than 0 and finite. We thus require that $\sigma_w^2 \sim 1/N$, so that in the limit of $N \to \infty$, $\text{Var}[\bar{h}_i] = N\sigma_w^2\overline{\nu^2} \sim \mathcal{O}(1)$. Furthermore, the mean time-averaged synaptic drive μ_h scales with N, meaning that in the large N limit $\nu_i \to 0$ or ν_{max} (depending on whether $\bar{w}$ is negative or positive, respectively). We should thus also enforce that the mean weight $\bar{w} \sim 1/N$. This is our first result: in a randomly fully connected current-based network with sigmoidal gain functions, the mean and variance of the weights need to scale with $1/N$ in order for the neurons to have a spread in firing weights.

Another possibility could be to set $w_{ij} = \frac{\tilde{w}_{ij}}{N}$ with $\tilde{w}_{ij}$ i.i.d. with mean and variance $\tilde{w}, \sigma_{\tilde{w}^2} \sim \mathcal{O}(1)$, so that $\bar{w} = \frac{\tilde{w}}{N} \sim \mathcal{O}(1/N)$. However, this would entail $\sigma_w^2 = \frac{\sigma_{\tilde{w}}}{N^2} \Rightarrow \text{Var}[\bar{h}_i] \sim \mathcal{O}(\frac{1}{N})$, quickly leading to constant firing rates across the network (i.e. $P(\nu) = \delta(\nu - \bar{\nu})$) as $N \to \infty$. This problem illustrates the fine-tuning required to obtain realistic dynamics in a randomly fully connected network: it is hard to keep μ_h low while also ensuring $\text{Var}[\bar{h}_i] \sim \mathcal{O}(1)$. A viable alternative along these lines would be $w_{ij} = \frac{\tilde{w}_{ij}}{\sqrt{N}}$, in which case $\text{Var}[\bar{h}_i] \sim \mathcal{O}(1)$. But in this case $\mu_h \sim \mathcal{O}(\sqrt{N})$, thus only partially solving the problem since the mean synaptic drive will still (albeit slowly) tend towards the saturated regime as $N \to \infty$. This $\sqrt{N}$ solution will, however, become important when we have both excitation and inhibition!

3.1.2 Sparse Connectivity

Such *dense* connectivity structure - in which every neuron is connected to every other neuron - evidently requires strong restrictions on the weights for the network to be able to generate realistic dynamics. This might also explain why it is rarely found in nature, where cortical connectivity rates, for example, are on the order of 10%. We can incorporate such *sparse* connectivity structure into our mean field equations by adding a parameter $K << N$ that controls the mean number of outgoing connections per neuron, so that the probability that any any two neurons are connected - called the *connectivity rate* - is equal to K/N (because the network is randomly connected). We can then write

$$\begin{aligned}
w_{ij} &= \zeta_{ij}\tilde{w}_{ij} \\
\zeta_{ij} \in \{0,1\} &\sim \text{Bernoulli}\,(K/N) \\
\tilde{w}_{ij} &\text{ i.i.d. with mean } \tilde{w} \text{ and variance } \sigma_{\tilde{w}}^2 \\
\Rightarrow \mu_h &= N\bar{w}\bar{\nu} = N\frac{K}{N}\tilde{w}\bar{\nu} = K\tilde{w}\bar{\nu} \\
\Rightarrow \sigma_h^2 &= \sigma_w^2\overline{\nu^2} \\
&= \left(\langle\zeta^2\rangle\langle\tilde{w}^2\rangle - \langle\zeta\rangle^2\langle\tilde{w}\rangle^2\right)\overline{\nu^2} \\
&= \left(\frac{K}{N}(\sigma_{\tilde{w}}^2 + \tilde{w}^2) - \frac{K^2}{N^2}\tilde{w}^2\right)\overline{\nu^2} \\
&= \frac{K}{N}\left(\sigma_{\tilde{w}}^2 + \left(1 - \frac{K}{N}\right)\tilde{w}^2\right)\overline{\nu^2}
\end{aligned}$$

since $\zeta_{ij}, \tilde{w}_{ij}$ are independent. Writing it in this form makes it easy to interpret: in the large N limit, the variance of the synaptic drive $\text{Var}[\bar{h}_i] = N\sigma_h^2$ is proportional to the variance of the (on average) K non-zero input connections plus a correction for the remaining absent connections with weight 0. Thus, $\text{Var}[\bar{h}_i] \sim \mathcal{O}(K)$ is independent of N regardless of $\sigma_{\tilde{w}}^2$. If we want to maintain the connectivity rate $p = K/N$ constant, however, $K \propto N$ so we will need $\tilde{w}, \sigma_{\tilde{w}} \sim 1/\sqrt{K}$ for the firing rates to maintain variable non-saturating firing rates. However, it is still the case that $\mu_h \sim \mathcal{O}(\sqrt{K})$ so we will need either a really small p or a really small constant of proportionality $k_h = \mu_h/\sqrt{(K)}$.

3.2 Excitatory and Inhibitory

We now make one vital step towards realism by including both excitatory and inhibitory neurons, creating four types of synaptic weights $w_{ij}^{\text{QR}} > 0$ from R neurons onto Q neurons, R,Q $\in$ {E,I} standing for excitatory (E) or inhibitory (I). The synaptic drive to a Q neuron is then given by

$$h_i^{\text{Q}}(t) = \sum_{j\in E} w_{ij}^{\text{QE}} g_j^{\text{E}}(t) - \sum_{j\in I} w_{ij}^{\text{QI}} g_j^{\text{I}}(t) + I_{\text{Q}}$$

where, for generality, I have also included an external constant input current I_{Q} injected equally into all Q neurons. Assuming a fixed connectivity rate $p = K/N$, and setting

$$w_{ij}^{\mathrm{QR}} = \zeta_{ij}^{\mathrm{QR}} \frac{\tilde{w}_{ij}^{\mathrm{QR}}}{\sqrt{K}}$$
$$\zeta_{ij}^{\mathrm{QR}} \sim \text{Bernoulli}(p)$$
$$\tilde{w}_{ij}^{\mathrm{QR}} \text{ i.i.d. with mean } \bar{w}_{\mathrm{QR}} \sim \mathcal{O}(1) \text{ and variance } \sigma^2_{w_{\mathrm{QR}}} \sim \mathcal{O}(1)$$
$$I_{\mathrm{Q}} = \frac{\bar{I}_{\mathrm{Q}}}{\sqrt{K}}$$

we can exactly repeat the above derivation, giving us, in the large N limit (for both subpopulations):

$$\overline{\nu_{\mathrm{Q}}^{\ell}} = \int \phi^{\ell} \left(\sqrt{K} \left(\bar{w}_{\mathrm{QE}} \overline{\nu}_{\mathrm{E}} - \bar{w}_{\mathrm{QI}} \overline{\nu}_{\mathrm{I}} + \bar{I}_{\mathrm{Q}} \right) + \sigma \xi \right) \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}} \, \mathrm{d}\xi \tag{46a}$$
$$\sigma^2 = \sigma^2_{\mathrm{QE}} + \sigma^2_{\mathrm{QI}} \tag{46b}$$
$$\sigma^2_{\mathrm{QR}} = \left(\sigma^2_{w_{\mathrm{QR}}} + (1-p)\bar{w}^2_{\mathrm{QR}} \right) \overline{\nu_{\mathrm{R}}^2} \tag{46c}$$

In this case, we can hope for realistic dynamics even for unbounded N: if $\tilde{w}_{QE}, \tilde{w}_{QI}$ are picked carefully enough so that $\tilde{w}_{\mathrm{QE}}\overline{\nu}_{\mathrm{E}} - \tilde{w}_{\mathrm{QI}}\overline{\nu}_{\mathrm{I}} \approx 0$ (the so-called balanced regime), then, under infinitesimally small external input $\bar{I}_{\mathrm{Q}}$, $\sigma \sim \mathcal{O}(1)$ and we are guaranteed to stay within a brain-like regime of time-averaged firing rates.

In sum, for a current-based network model with

- saturating gain function $\phi(\cdot)$
- synaptic drive

$$h_i(t) = \sum_j w_{ij} g_j(t) + I_i^{ext}(t) = \underbrace{\sum_j w_{ij}\nu_j + I_i}_{\bar{h}_i} + \underbrace{\sum_j w_{ij}\delta g_j(t) + \delta I_i(t)}_{\delta h_i(t)}$$

- connectivity rate $p = K/N$
- random i.i.d. non-zero weights w_{ij} with mean $\bar{w}$ and variance σ_w^2
- random i.i.d. time-averaged inputs $I_i \sim \mathcal{O}(\sum_j w_{ij}\nu_j)$ with mean $\bar{I}$ and variance σ_I^2

the following holds in the large $N \to \infty$ limit:

$$\boxed{\overline{\nu^{\ell}} = \int \phi \left(K\bar{w}\bar{\nu} + \bar{I} + \sqrt{K((\sigma_w^2 + (1-p)\bar{w}^2)\overline{\nu^2} + \sigma_I^2)}\xi \right)^{\ell} \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}} \, \mathrm{d}\xi}$$

For such a network to have realistic dynamics (i.e. different firing rates across the population), it is therefore necessary that

$$\sigma_w^2 \sim \frac{1}{K}$$

3.2.1 Wilson-Cowan Equations

We're now going to make up some reasonable dynamical equations, and use them to analyse the behaviour of the network:

$$\tau_{\mathrm{E}}\dot{\nu}_{\mathrm{E}} = \psi_{\mathrm{E}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}}) - \nu_{\mathrm{E}}$$
$$\tau_{\mathrm{I}}\dot{\nu}_{\mathrm{I}} = \psi_{\mathrm{I}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}}) - \nu_{\mathrm{I}}$$

where, as per equation 46,

$$\psi_{\mathrm{Q}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}}) = \int \phi_{\mathrm{Q}} \left(\sqrt{K} \left(\bar{w}_{\mathrm{QE}}\overline{\nu}_{\mathrm{E}} - \bar{w}_{\mathrm{QI}}\overline{\nu}_{\mathrm{I}} \right) + I_{\mathrm{Q}} + \sigma\xi_i^Q \right) \frac{e^{-\frac{\xi^2}{2}}}{\sqrt{2\pi}} \, \mathrm{d}\xi$$

Thus, $\psi_{\mathrm{Q}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}})$ is just a Gaussian-smoothed version of the corresponding individual neuron gain function $\phi_{\mathrm{Q}}(\cdot)$, i.e. some smoother sigmoid thing. These are called the Wilson-Cowan equations, and can be alternatively

derived from very general assumptions [Wilson and Cowan, 1972]. It is possible to make headway understanding this 2D system, in a way that wasn't with N dimensions, using the tools of nullcline analysis. The nullclines are given by:

$$\nu_{\mathrm{E}} = \psi_{\mathrm{E}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}})$$
$$\nu_{\mathrm{I}} = \psi_{\mathrm{I}}(\nu_{\mathrm{E}}, \nu_{\mathrm{I}})$$

We can find these graphically by plotting the left and right hand side of these equations and finding the points of intersection for different firing rates, figure 3.2.

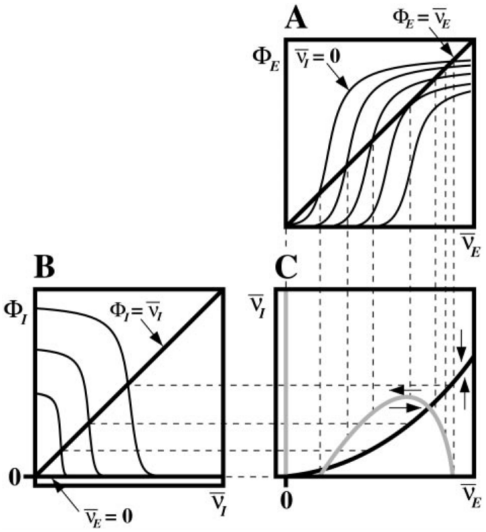


Figure 3.2: From [Latham et al., 2000]. $\phi_{\mathrm{Q}} = \psi_{\mathrm{Q}}$ on the labels. A gives the graphical method for finding the excitatory nullcline, B gives the inhibitory version, and the nullclines are plotted in C.

The intersections of the nullclines in figure 3.2C give the fixed points, and we will try and analyse their nature. Let $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*)$ designate the location of one such fixed point, such that

$$\nu_{\mathrm{E}}^* = \psi_{\mathrm{E}}(\nu_{\mathrm{E}}^*, \nu_{\mathrm{E}}^*)$$
$$\nu_{\mathrm{I}}^* = \psi_{\mathrm{I}}(\nu_{\mathrm{I}}^*, \nu_{\mathrm{E}}^*)$$

As per standard stability analysis (appendix D), we can find out if this fixed point is stable by looking at the Jacobian matrix of the dynamical system evaluated at the fixed point, here given by

$$\mathbf{J} = \begin{bmatrix} \tau_{\mathrm{E}}^{-1}(\psi_{\mathrm{E,E}} - 1) & \tau_{\mathrm{E}}^{-1}\psi_{\mathrm{E,I}} \\ \tau_{\mathrm{I}}^{-1}\psi_{\mathrm{I,E}} & \tau_{\mathrm{I}}^{-1}(\psi_{\mathrm{I,I}} - 1) \end{bmatrix}$$

where we define

$$\psi_{\mathrm{Q,R}} \equiv \left.\frac{\partial \psi_{\mathrm{Q}}}{\partial \nu_{\mathrm{R}}}\right|_{\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*}$$

We then know that the fixed point will be stable iff the trace and determinant of $\mathbf{J}$ are negative and positive, respectively, giving us the following conditions for the stability of $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*)$:

$$\frac{1 - \psi_{\mathrm{I,I}}}{\tau_{\mathrm{I}}} > \frac{\psi_{\mathrm{E,E}} - 1}{\tau_{\mathrm{E}}}$$
$$(\psi_{\mathrm{E,E}} - 1)(\psi_{\mathrm{I,I}} - 1) > \psi_{\mathrm{E,I}}\psi_{\mathrm{I,E}}$$

It turns out we can verify these conditions geometrically by expressing the slopes of the nullclines in terms of the partial derivatives $\psi_{\mathrm{Q,R}}$. To do so, we consider a perturbation from the fixed point $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*) \to (\nu_{\mathrm{E}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{E}}, \nu_{\mathrm{I}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{I}})$ *along the* ν_E*-nullcline* - hence the E subscript on the δ's. Since we know the equation for the

nullcline, we can use this to compute its slope at the fixed point via a first-order Taylor approximation:

$$\begin{aligned}
\nu_{\mathrm{E}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{E}} &= \psi_{\mathrm{E}}(\nu_{\mathrm{E}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{E}},\, \nu_{\mathrm{I}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{I}}) \\
&\approx \psi_{\mathrm{E}}(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*) + \delta_{\mathrm{E}}\nu_{\mathrm{E}} \frac{\partial \psi_{\mathrm{E}}}{\partial \nu_{\mathrm{E}}}\bigg|_{\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*} + \delta_{\mathrm{E}}\nu_{\mathrm{I}} \frac{\partial \psi_{\mathrm{E}}}{\partial \nu_{\mathrm{I}}}\bigg|_{\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*} \\
&= \nu_{\mathrm{E}}^* + \delta_{\mathrm{E}}\nu_{\mathrm{E}}\psi_{\mathrm{E,E}} + \delta_{\mathrm{E}}\nu_{\mathrm{I}}\psi_{\mathrm{E,I}} \\
\Leftrightarrow \delta_{\mathrm{E}}\nu_{\mathrm{E}} &= \delta_{\mathrm{E}}\nu_{\mathrm{E}}\psi_{\mathrm{E,E}} + \delta_{\mathrm{E}}\nu_{\mathrm{I}}\psi_{\mathrm{E,I}} \\
\Leftrightarrow \frac{\delta_{\mathrm{E}}\nu_{\mathrm{I}}}{\delta_{\mathrm{E}}\nu_{\mathrm{E}}} &= \frac{1 - \psi_{\mathrm{E,E}}}{\psi_{\mathrm{E,I}}}
\end{aligned}$$

Performing the same calculation for the ν_{I}-nullcline, we have that the slopes of the excitatory and inhibitory nullclines at the fixed point are given by, respectively

$$\begin{aligned}
m_{\mathrm{E}} &= \frac{\delta_{\mathrm{E}}\nu_{\mathrm{I}}}{\delta_{\mathrm{E}}\nu_{\mathrm{E}}} = \frac{1 - \psi_{\mathrm{E,E}}}{\psi_{\mathrm{E,I}}} \\
m_{\mathrm{I}} &= \frac{\delta_{\mathrm{I}}\nu_{\mathrm{I}}}{\delta_{\mathrm{I}}\nu_{\mathrm{E}}} = \frac{\psi_{\mathrm{I,E}}}{1 - \psi_{\mathrm{I,I}}}
\end{aligned}$$

Because $\psi_{\mathrm{Q,E}} > 0, \psi_{\mathrm{Q,I}} < 0$ we then know that

$$\begin{aligned}
&m_{\mathrm{I}} > 0 \text{ always} \\
&m_{\mathrm{E}} \begin{cases} < 0 \Leftrightarrow 0 < \psi_{\mathrm{E,E}} < 1 \\ > 0 \Leftrightarrow \psi_{\mathrm{E,E}} > 1 \end{cases}
\end{aligned}$$

We can now relate our stability conditions on the partial derivatives $\psi_{\mathrm{Q,R}}$ to statements about the nullcline slopes at the fixed point:

$$\begin{aligned}
m_{\mathrm{E}} < m_{\mathrm{I}} &\Leftrightarrow (\psi_{\mathrm{E,E}} - 1)(\psi_{\mathrm{I,I}} - 1) > \psi_{\mathrm{E,I}}\psi_{\mathrm{I,E}} \\
m_{\mathrm{E}} < 0 &\Rightarrow \frac{1 - \psi_{\mathrm{I,I}}}{\tau_{\mathrm{I}}} > \frac{\psi_{\mathrm{E,E}} - 1}{\tau_{\mathrm{E}}}
\end{aligned}$$

The first statement tells us that for the fixed point $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*)$ to be stable, the slope of the inhibitory nullcline at this point has to be steeper than that of the excitatory nullcline. This is intuitive: the inhibitory population mean firing rate ν_{I} has to be more sensitive to changes in the excitatory population mean firing rate than the excitatory population itself for the negative feedback to be strong enough to generate a stable state. Graphically, this translates to the inhibitory nullcline intersecting the excitatory nullcline at $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*)$ from below.

The second condition tells us that if the excitatory nullcline has negative slope at $(\nu_{\mathrm{E}}^*, \nu_{\mathrm{I}}^*)$ (i.e. the fixed point lies on a *stable branch* of the ν_{E}-nullcline), we know that the fixed point is stable, otherwise (the fixed point lies on an *unstable branch* of the ν_{E}-nullcline) we don't know - it depends on the values of $\psi_{\mathrm{E,E}}, \psi_{\mathrm{I,I}}, \tau_{\mathrm{E}}, \tau_{\mathrm{I}}$. Particularly, given similar $\tau_{\mathrm{E}} \approx \tau_{\mathrm{I}}$, we can see from our original stability conditions that the fixed point will be stable for highly negative $\psi_{\mathrm{I,I}}$ and small $\psi_{\mathrm{E,E}}$. In other words, if $m_{\mathrm{E}} > 0$ at the fixed point, then the fixed point will only be stabile if there is weak coupling between excitatory firing rates - i.e. weaker positive feedback - and strong coupling between inhibitory neurons - i.e. stronger disinhibiton. This latter requirement might seem counterintuitive, but it makes sense mathematically when you consider that it is the only coupling able to pull local perturbations from an equilibrium back to it: *E-E* coupling repels them further away and *E-I*,*I-E* couplings rotates them around the equilibrium, but *I-I* coupling brings them back. Thus, a strong *I-I* coupling - i.e. a highly negative $\psi_{\mathrm{I,I}}$ - is necessary for a fixed point on an unstable branch of the ν_{E}-nullcline to be stabile. Note that in the case of $m_{\mathrm{E}} > 0$, a small $\psi_{\mathrm{E,E}}$ implies a small m_{E}, i.e. a more shallow positive slope (recall that $\psi_{\mathrm{E,I}} < 0$ so in this case $\psi_{\mathrm{E,E}} \geq 1$). So as the excitatory nullcline slope gets larger at the fixed point, it becomes more unstable (eventually leading to a Hopf bifurcation).

Using these conditions, we can now go back to our standard *E-I* nullclines in figure 3.2C and conclude that the left and right intersections correspond to stable fixed points (with $m_{\mathrm{E}} < m_{\mathrm{I}}, m_{\mathrm{E}} < 0$) and the fixed point in between is therefore unstable (thus separating their basins of attraction). Thus, this set of nullclines is inconsistent with the observation of low average firing rates in cortex. While the rightmost stable state is at a high firing rate, the leftmost one is at (0,0), corresponding to the state where no neurons are firing and thus no neurons can or will fire (thus making it stable).

3.2.2 Simplified Version of Above Analysis

Some of the same conclusions can be found by a much simpler analysis. Take the following dynamical equations for the mean excitatory and inhibitory activity:

$$\tau_E \frac{d\nu_E}{dt} = \tilde{\phi}_E(\sqrt{K}(w_{EE}\nu_E - w_{EI}\nu_I + h_E)) - \nu_E$$
$$\tau_I \frac{d\nu_I}{dt} = \tilde{\phi}_I(\sqrt{K}(w_{IE}\nu_E - w_{II}\nu_I + h_I)) - \nu_I$$

We assume K is massive the only way the output of the saturating functions $\tilde{\phi}_Q$ are not either minimum or maximum is for the term multiplying $\sqrt{K}$ to be $\mathcal{O}(\frac{1}{\sqrt{K}}$. Let's say we're willing to make errors to $\mathcal{O}(\frac{1}{\sqrt{K}}$, then for firing rates to be stably non-saturating we require:

$$w_{EE}\nu_E - w_{EI}\nu_I + h_E = 0$$
$$w_{IE}\nu_E - w_{II}\nu_I + h_I = 0$$

So we've got some simplified, linear, first-order nullcines. Plotting these we can see many of the same conclusions, endogenously active neurons are needed for there to be stable non-zero firing rates, and these are only zero when inhibition stabilised.

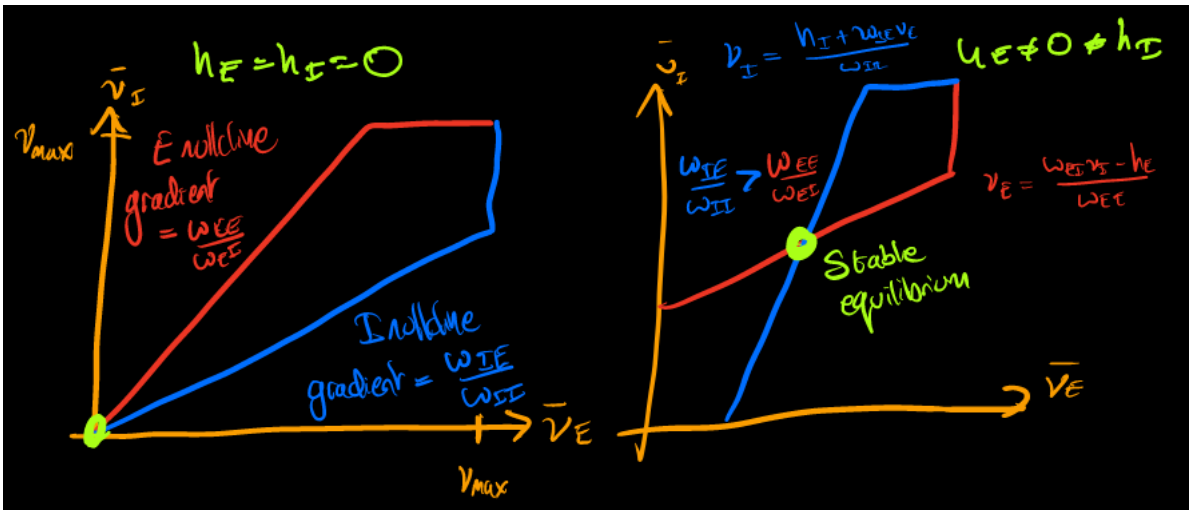


Figure 3.3: Linear nullclines plotted, there's a stable equilibrium only when non-zero inputs, and when inhibition stablilises.

3.2.3 Observation 2: How to get stable low firing rates

To get a stable state at a low firing rate, one can see graphically that we need the excitatory and inhibitory nullclines to both shift upwards such that the leftmost intersection is pushed out of the origin. Recalling that $\psi_E(\nu_E, \nu_I), \psi_I(\nu_E, \nu_I)$ are monotonically increasing functions of I_E, I_I, we can shift the nullclines up by simply increasing the external current inputs I_E, I_I, which until now were set to 0. This corresponds to shifting the ψ_E, ψ_I curves in figure 3.2A,B to the left and right, respectively. It turns out that if we increase I_E, I_I enough so that $\psi_E(0,0), \psi_I(0,0) > 0$ (i.e. shift the ψ_E, ψ_I curves until the y-intercept is above 0, see regime 3 in figure 2 of Latham et al., 2000), we end up with nullclines roughly looking like those in figure 3.4, with an equilibrium at low mean firing rates. Biologically, this shift in the equilibrium gain functions such that their y-intercepts are above 0 can be interpreted as there being a number of *endogenously active* cells in the population, which can have non-zero firing rates in the abscence of any recurrent input from the rest of the population. Our analysis thus suggests that an external current input strong enough to endow each subpopulation with endogenously active cells is necessary for networks to have a stable equilibrium at low average firing rates, like what is observed in the mammalian neocortex ([Latham et al., 2000]). In a real brain, this external current could come from upstream inputs, membrane potential noise, or some single-cell intrinsic membrane currents.

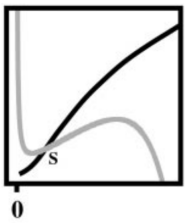


Figure 3.4: From [Latham et al., 2000]. Axes as in figure 3.2C.

But we still have to confirm that the new equilibrium we have found is stable. As drawn in the figure, the equilibrium lies on the unstable branch of the excitatory nullcline (i.e. $m_{\text{E}} > 0$), meaning that its stability depends on the strength of the *E-E* and *I-I* coupling. We could instead push the inhibitory nullcline further up so that it intersects the excitatory one on its leftmost stable branch, but this would create a stable equilibrium where the mean inhibitory firing rate is greater than the mean excitatory firing rate. Such a regime is rarely seen in real brains, and it can be shown that the minimum of the excitatory nullcline is so small ($\sim .01$Hz) that this equilibrium would correspond to unrealistically low excitatory firing rates [Latham et al., 2000]. We thus conclude that the equilibrium has to be on the unstable branch of the excitatory nullcline as in figure 3.4, so its stability depends on the *E-E* and *I-I* coupling.

We can regulate this coupling by changing the mean synaptic weight strengths $\bar{w}_{\text{EE}}, \bar{w}_{\text{II}}$. But note that this also results in shifts of the nullclines: namely, increasing excitatory synaptic weights $\bar{w}_{\text{EE}}(\bar{w}_{\text{IE}})$ leads to upward shifts of the excitatory(inhibitory) nullclines while increasing inhibitory synaptic weights $\bar{w}_{\text{EI}}(\bar{w}_{\text{II}})$ lowers them. Since a rise in the excitatory/inhibitory nullcline pushes the equilibrium to higher/lower mean firing rates, this translates to high $\bar{w}_{\text{EE}}, \bar{w}_{\text{II}}$ pushing the equilibrium mean firing rates up and high $W_{\text{EI}}, W_{\text{IE}}$ pushing them down.

Summing this all up, a randomly connected network with endogenously active cells will have a stable equilibrium with low mean firing rates (observation #2) if the mean *E-E* connection strenghts are weak (to stabilize and lower the equilibrium), the mean *E-I* and *I-E* connection strenghts are strong (to lower the equilibrium), and the mean *I-I* connection strengths are strong (to stabilize) but not too strong (to lower).

3.2.4 Observations 4 & 5: Oscillations in Cortex

On the other hand, if the connectivity is such that the *E-E*/*I-I* connections are too strong/weak, then the mean firing rate equilibrium becomes unstable. But we now note that the derivatives on the boundaries of figure 3.4 all point inwards: anything above(below) the ν_{I}-nullcline has a downward(upward) facing derivative (since $\nu_{\text{I}} > (<)\, \psi_{\text{I}}(\nu_{\text{E}}, \nu_{\text{I}})$), and anything left(right) of the ν_{E}-nullcline has a rightward(leftward) facing derivative (since $\nu_{\text{E}} < (>)\, \psi_{\text{I}}(\nu_{\text{E}}, \nu_{\text{I}})$). By the Poincarè-Bendixson Theorem (see appendix D), then, there must be a limit cycle around this unstable equilibrium. In other words, a randomly connected network with strong *E-E* connections and weak *I-I* conenctions can exhibit oscillations, like those observed in the cortex (observation #4). This transition from stable to unstable + oscillations is called a *Hopf bifurcation* (see section 2.6 for more on bifurcations). The shape of the limit cycle also explains observation #5: excitatory leads inhibitory, because it is the inhibitory neurons that are stabilising the cycle, and they kick in only after the excitatory firing has risen.

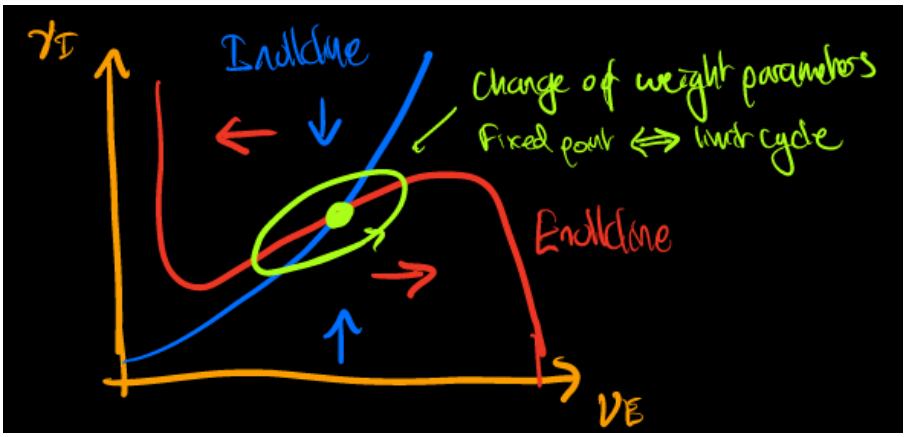


Figure 3.5: Dynamics around fixed point all point inwards, so if the nature of the fix point changes you get a stable limit cycle.

3.2.5 Observation 6: Spike-Frequency Adaption and ON/OFF states

Finally, we consider the more biologically realistic case of dynamics under spike-frequency adaptation. We consider a parameter regime where there are few endogenously active cells, such that the nullclines look like those in figure 3.6. We incorporate spike-frequency adaptation into our model simply by expressing the external current input I_{Q} as a dynamical variable:

$$I_{\text{Q}} = \theta_{\text{Q}} - g_{\text{Q}}$$
$$\tau_{\text{SFA}} \frac{\mathrm{d}g_{\text{Q}}}{\mathrm{d}t} = G_{\text{Q}}\nu_{\text{Q}} - g_{\text{Q}}$$

where G_Q is a constant, θ_Q reflects the number of endogenously active cells under no adaptation (in the form of some kind of external current), and $Q \in \{E, I\}$. Thus, as the mean firing rate of subpopulation Q increases, g_Q increases, pushing I_Q below 0 as the endogenously and non-endogenously active cells are silenced via spike-frequency adaptation. This results in a downward shift of the nullclines, as illustrated in figures 3.6 A→B→C. In this parameter regime where θ_Q is not too big, this results in a series of bifurcations: as the nullclines shift from A to B, a new 0-firing rate equilibrium is created, to which the network is forced to in the shift from B to C as the original non-zero firing rate equilibrium is destroyed. Once the firing rates drop to this new equilibrium, g_Q goes to 0 and $I_Q \to \theta_Q$, pushing the equilibrium back up to its non-adapted state. Crucially, τ_{SFA}, on the order of seconds, is much larger than τ_E, τ_I, so the network settles to equilibrium between the bifurcations. This leads to bursting: transitions between states of high mean firing rates (UP states) and states of very low mean firing rates (DOWN states), which last on the order of seconds (namely, on the order of spike-frequency adaptiation time τ_{SFA}). We thus see that, in this regime, randomly connected networks can replicate observation #6 - the on off states [Latham et al., 2000].

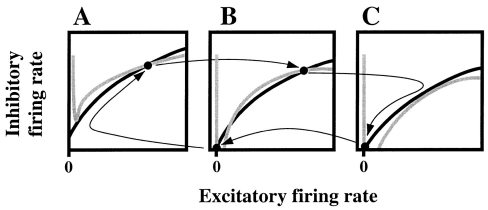


Figure 3.6: From [Latham et al., 2000]

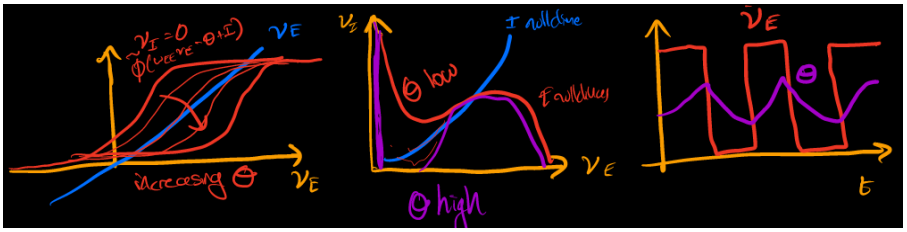


Figure 3.7: In the first figure we graphically calculate the shape of the E nullcline, as the threshold increases these change from the red line in the central plot to the purple. This changes the nature of the stable equilbrium from 0 firing to low firing, giving the ON/OFF behaviour.

3.2.6 Observation 7: Effect of Perturbations and Bumps of Activity

Finally, we note that the nullclines in figure 3.6B can also generate observation #7, in the abscence of spike-frequency adaptation. In this case, we have a stable equilibrium at 0 mean firing rate and a stable or unstable equilibrium at higher firing rate separated by a saddle. If this higher equilibrium is unstable, then any perturbation to the lower stable equilibrium that pushes it rightward beyond the saddle will lead to a trajectory that loops around the high firing rate equilibrium before returning to the 0 mean firing rate equilibrium. This is evocative of bump responses, where perturbation or stimulus-evoked responses lead to a short period of high membrane potential (and thus high firing rates) before quickly returning to a DOWN state (observation #7), fig

3.2.7 Observation 8: Switching

We've seen how many types of qualitatively different behaviour can emerge from the nullclines in different parameters settings: bumps, ON/OFF switching, oscillations, or stable firing. Any effect (neuromodulator, external input, internal neuronal excitabtility changes) that changes the parameter effects could switch the circuit from one behaviour to another, as seen in cortex. All that remains to be explained is the irregular firing and for that we will have to consider more than just a mean field approach.

3.3 Temporal Effects

We now briefly turn to problem #2 outlined above: characterizing temporal correlations in network activity. For this, we turn to the dynamic noise component of the synaptic drive, given by

$$\delta h_i(t) = \sum_j w_{ij} \delta g_j(t)$$

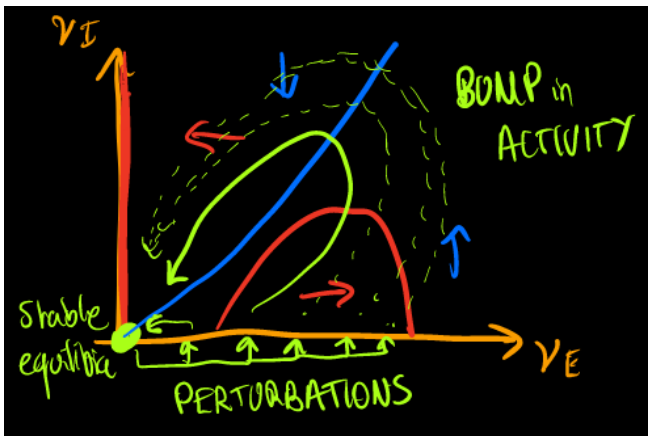


Figure 3.8: In a certain setting of the parameters perturbations of beyond a certain size lead to large deviations in firing rates. These are like the bumps seen in average firing rates.

where $\delta g_j(t) \equiv g_j(t) - \bar{g}_j$ are 0-mean fluctuations of the input synaptic conductances. It turns out that with i.i.d. weights, in the limit of large N two things happen: (i) the weights and fluctuations decouple, and (ii) the fluctuations across pairs of different neurons become uncorrelated. Intuitively, this is can be shown to be true when the connectivity is very sparse, when $K << N$ (i.e. $p << 1$, [Vreeswijk and Sompolinsky, 1998]). Somewhat less intuitively, it turns this is also true in E/I networks in the so-called "balanced state", where the total excitatory and inhibitory input to a given neuron are correlated and cancel each other out ([Renart et al., 2010, Rosenbaum et al., 2017])[5]. Thus, we again find ourselves with a big sum of independent random variables. In this case, however, we have a dynamical variable that is a function of time, so the CLT tells us that in the large N limit $\delta h_i(t)$ becomes a draw from a Gaussian *process*.

Thus, all we need to fully characterize the statistics of the synaptic drive fluctuations $\delta h_i(t)$ is their mean and covariance. By construction, their mean is 0, and their covariance is captured by the cross-correlation function

$$C_{ij}(\tau) = \langle \delta h_i(t) \delta h_j(t+\tau) \rangle_t$$

where, following the above notational convention with expectations, the expectation is over time t. Given that in the large N limit neurons are uncorrelated, $C_{ij} = 0$ whenever $i \neq j$. Furthermore, in our homogenous randomly connected network model, there is nothing to distinguish one neuron from another, so the only thing we really care about is the population mean autocorrelation function:

$$\begin{aligned}
C(\tau) &= \frac{1}{N} \sum_i C_{ii}(\tau) \\
&= \frac{1}{N} \sum_i \langle \delta h_i(t) \delta h_i(t+\tau) \rangle_t \\
&= \frac{1}{N} \sum_i \sum_{j,j'} w_{ij} w_{ij'} \langle \delta g_j(t) \delta g_{j'}(t+\tau) \rangle_t \\
&= \frac{1}{N} \sum_{i,j} w_{ij}^2 \langle \delta g_j(t) \delta g_j(t+\tau) \rangle_t \\
&= \sum_j \left(\frac{1}{N} \sum_i w_{ij}^2 \right) \langle \delta g_j(t) \delta g_j(t-\tau) \rangle_t \\
&\simeq N \overline{w^2} \Delta(\tau), \quad \Delta(\tau) = \frac{1}{N} \sum_j \langle \delta g_j(t) \delta g_j(t-\tau) \rangle_t
\end{aligned}$$

where the fourth equality follows from our assumption of no correlations, and the last approximation follows from the weights being i.i.d. and therefore self-averaging in the $N \to \infty$ limit.

We now have a full statistical characterization of the synaptic drives in a spiking network with i.i.d. weights (exact in the $N \to \infty$ limit under a few extra assumptions - namely, no correlations), as a function of the statistics of the spiking output (namely, the first and second moments of the time-averaged firing rates $\bar{\nu}, \overline{\nu^2}$ and the population mean autocorrelation of synaptic conductances $\Delta(\tau)$). However, unlike in the case of the quenched noise $\bar{h}_i$, it is not clear how to relate correlations in synaptic drives $C(\tau)$ to correlations in

[5] For a brief overview of the 20+ years it took for theorists to figure out why this was the case, see [Latham, 2017]

synaptic conductances $\Delta(\tau)$ to obtain a set of self-consistent equations we can solve for $C(\tau), \Delta(\tau)$. To my knowledge this has never been done analytically for a particular spiking neuron model $f_i(V_i, t)$ (except under the assumption of Poisson firing, which makes everything pretty straightforward given you know the f-I curve: [Grabska-Barwińska and Latham, 2014]).

However, in simplified rate-based models, the output autocorrelation $\Delta(\tau)$ can be solved analytically to provide substantial insight into the possible dynamical regimes of the network ([Sompolinsky et al., 1988, Mastrogiuseppe and Ostojic, 2017]). For example, consider a randomly connected *tanh* network with dynamics

$$\dot{x}_i = -x_i + \sum_{j=1}^{N} w_{ij}\phi(\gamma x_j)$$

with i.i.d. 0-mean weights $w_{ij} \sim \mathcal{N}(0, 1/N)$. In this case, the "synaptic conductances" are given by $g_j(t) = \phi(\gamma x_j(t))$, where γ parametrizes the steepness of the non-linearity $\phi(\cdot) = \tanh(\cdot)$. Given this simple relationship between the synaptic conductances and the neurons' "potentials" x_i, one can show through some tedious but straightforward algebra that the autocorrelation satisfies the following differential equation:

$$\Delta(\tau) - \frac{\mathrm{d}^2\Delta}{\mathrm{d}\tau^2} = C(\tau)$$

with the following intuitive boundary conditions:

- $\Delta(\tau) \leq \Delta(0)$ since the autocorrelation will always be highest at 0 time lag
- $\Rightarrow \frac{\mathrm{d}\Delta}{\mathrm{d}\tau}\Big|_{\tau=0} = 0$ since $\tau = 0$ is an extremum of the function
- $\Rightarrow \frac{\mathrm{d}^2\Delta}{\mathrm{d}\tau^2}\Big|_{\tau=0} < 0$ since $\tau = 0$ is a maximum

Along with the above ODE, these boundary conditions allow you to make general statements about the shape of $\Delta(\tau)$, which gives qualitative insights into the trajectories of the system. Particularly, for certain values of γ, $\Delta(\tau) \to 0$ as $\tau \to \infty$, indicating that trajectories diverge over time and the system is chaotic ([Sompolinsky et al., 1988]).

4 Structured Networks

Our previous analysis relied on assuming the synaptic weights were drawn iid from some distribution. While this was sufficient to get a lot of insight into the behaviour of networks, it is obviously not correct! Your synaptic weights (assuming that is a useful term) are tuned over both evolutionary and lifetime timescales to respond to inputs appropriately. In this section we consider a few simple models of networks with structured connectivity.

As a first discussion we might wonder about the correct scaling of the weights, which after much discussion we concluded should be $\frac{1}{\sqrt{K}}$ for random weights. If we imagine the weights have both a random ($\mathbf{w}$) and a structured ($\mathbf{J}$) component:

$$\tilde{w}_{ij} = \frac{1}{\sqrt{K}} w_{ij} + \alpha J_{ij}$$

we wonder what should be the scaling of α with K in order to maintain the broad spread in firing rates, if both w_{ij} and J_{ij} are $\mathcal{O}(1)$. A common choice for the form of the structured connectivity is some low rank matrix that picks out prefered directions in activity space:

$$\mathbf{J} = \alpha \sum_{\mathbf{l}} \mathbf{u_l} \mathbf{u_l^T}$$

If the activity, $\mathbf{v}$, aligns with one of the prefered directions then $\mathbf{Jv} \sim \alpha \cdot \mathcal{O}(\mathbf{K}) \cdot \mathbf{u_l}$, so in order to ensure the contribution of this term is $\mathcal{O}(1)$, α must scale as $\frac{1}{K}$. i.e. the structured weights have to be a factor of $\frac{1}{\sqrt{K}}$ smaller than the random ones. This means very small changes to the weights can create the kinds of structure we are considering.

4.1 Hopfield Network

Associative memory is, I believe, the word psychology types use for that experience when just smelling the rain on a hot road brings back memories of childhood summers (I'm sure you all have your own examples). Hopfield networks provide a basic model of associative memory, and provoked a lot of interest for turning such a recognisable part of our experiences into analysable mathematics. Here we go through the basics of Hopfield networks, attempts to improve on them, before following following a paper of Peter's as it struggles to turn the Hopfield network into reasonable dynamics [Roudi and Latham, 2007].

4.1.1 Basics

We consider a fully-connected network of $N+1$ neurons with discrete states given by $s_i(t) \in \{1, -1\}$. We model their dynamics in discrete time, using the update

$$s_i(t+1) = \text{sign}\left[\sum_{j=1}^{N} J_{ij} s_j(t)\right]$$

where

$$\text{sign}[x] = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{else} \end{cases}$$

The crucial property of the Hopfield network is its connectivity matrix, given by

$$J_{ij} = \frac{1}{N} \sum_{m=1}^{M} \xi_i^{(m)} \xi_j^{(m)}, \quad J_{ii} = 0$$

$$\xi_i^{(m)} = \begin{cases} 1 & \text{with probability } \frac{1}{2} \\ -1 & \text{with probability } \frac{1}{2} \end{cases}$$

All connections are thus symmetric $J_{ij} = J_{ji}$, with no autapses (synapse from one neuron back to itself).

Suppose now that $\forall i\ s_i(t) = \xi_i^{(m')}$. We then have

$$\begin{aligned}
s_i(t+1) &= \text{sign}\left[\sum_{j\neq i} J_{ij}s_j(t)\right] \\
&= \text{sign}\left[\sum_{j\neq i}\frac{1}{N}\sum_{m=1}^{M}\xi_i^{(m)}\xi_j^{(m)}\xi_j^{(m')}\right] \\
&= \text{sign}\left[\frac{1}{N}\sum_{j\neq i}\left(\xi_i^{(m')}\xi_j^{(m')}\xi_j^{(m')} + \sum_{m\neq m'}\xi_i^{(m)}\xi_j^{(m)}\xi_j^{(m')}\right)\right] \\
&= \text{sign}\left[\frac{1}{N}\sum_{j\neq i}\left(\xi_i^{(m')} + \sum_{m\neq m'}\xi_i^{(m)}\xi_j^{(m)}\xi_j^{(m')}\right)\right] \\
&= \text{sign}\left[\xi_i^{(m')} + \frac{1}{N}\sum_{j\neq i}\sum_{m\neq m'}\xi_i^{(m)}\xi_j^{(m)}\xi_j^{(m')}\right] \\
&= \text{sign}\left[\xi_i^{(m')} + \eta_i\right]
\end{aligned}$$

Since each of the terms inside the sum are independently distributed, in the limit of large N we can use the CLT to approximate η_i with a Gaussian random variable:

$$\eta_i \to \mu + \frac{\sigma}{\sqrt{N}}\zeta_i, \quad \zeta_i \sim \mathcal{N}(0,1)$$

$$\mu = \left\langle \sum_{m\neq m'}\xi_i^{(m)}\xi_j^{(m)}\xi_j^{(m')}\right\rangle = \sum_{m\neq m'}\langle\xi_i^{(m)}\rangle\langle\xi_j^{(m)}\rangle\langle\xi_j^{(m')}\rangle = 0$$

$$\sigma^2 = \sum_{m\neq m'}\left\langle\xi_i^{(m)^2}\right\rangle\left\langle\xi_j^{(m)^2}\right\rangle\left\langle\xi_j^{(m')^2}\right\rangle = \sum_{m\neq m'}(1)(1)(1) = M-1$$

Thus, we have:

$$s_i(t+1) = \text{sign}\left[\xi_i^{(m')} + \sqrt{\frac{M-1}{N}}\zeta_i\right], \quad \zeta_i \sim \mathcal{N}(0,1)$$

In other words, as long as $M << N$, $s_i(t) = \xi_i^{(m')}$ is an equilibrium of the system, since $s_i(t) \approx s_i(t+1)$. But as you increase M, the number of memories stored, this noise term will begin to dominate and your network will begin to fail.

Generalizing the above for all possible m', the system has M such equilibria, each being a local minimum of the network's *energy* given by

$$E = -\frac{1}{2}\sum_{i,j}J_{ij}s_is_j$$

The Hopfield network thus implements a kind of associative memory, whereby feeding it some input leads it to converge to the equilibrium state - or "memory" - most similar to the input. The Hopfield network can store M such memories as long as it has many more neurons than memories $N >> M$, namely at least 1000 neurons for every 138 memories (i.e. $\frac{M}{N} \leq 0.138$; Amit, Gutfreund & Sompolinsky, 1987).

4.1.2 Challenges & Improvements to Hopfield Networks

The big problem with Hopfield networks as a model of biological memory is the all-to-all connectivity, which is very unreaslistic. Most cortical neurons connect to around 1000 other neurons. As such, an appropriate model might include the Bernoulli variables, like we did for sparse random networks (section 3.1.2):

$$J_{ij} = \frac{1}{K}\sum_{\mu}c_{ij}\xi_i^\mu\xi_j^\mu \qquad c_{ij} = \begin{cases}1 & \text{with probability } \frac{K}{N} \\ 0 & \text{with probability } 1-\frac{K}{N}\end{cases}$$

You can do exactly the same analysis as before, but now the noise variance becomes $\approx \frac{P}{K}$, i.e. much larger! It means no matter how many neurons you can only store 138 memories. This either requires many different memory modules with substantial overhead invested in co-ordinating them, or some way of improving the storage efficiency.

Fortunately there is a solution, we just need to change the sparsity of the memory patterns:

$$\xi_i^\mu = \begin{cases} 1 & \text{with probability } f \\ -1 & \text{with probability } 1-f \end{cases}$$

It turns out [Tsodyks and Feigel'man, 1988] that this fixes the problem, and the capacity becomes $\frac{K}{f \log \frac{1}{f}}$. This is great news, since sparse firing patterns are observed in the brain, so its a win for hopfield on two counts, the more sparse the more capacity even without all-to-all connectivity.

4.1.3 Dynamical Model of Hopfield Networks

There are however a few problems remaining that we would like to fix in a real model of biological memory. Namely:

1. Neurons have continuous, not binary, firing rates
2. There is likely to be random background activity, not purely structured
3. There should be a 'no memory active' state. Hopfield networks are always in some attractor.
4. It shouldn't really be discrete dynamics

Motivated by our previous random network analysis we then introduce the following equations:

$$\tau_E \frac{d\nu_{E,i}}{dt} = \tilde{\phi}_E(\sqrt{K}(w_{EE}\nu_E - w_{EI}\nu_I) + \frac{\beta}{Nf(N-f)}\sum_j \xi_i(\xi_j - f)\nu_{E,j}) - \nu_{E,i}$$

$$\tau_I \frac{d\nu_{I,i}}{dt} = \tilde{\phi}_I(\sqrt{K}(w_{IE}\nu_E - w_{II}\nu_I)) - \nu_{I,i}$$

Here, ν_E without the neuron index represents the average excitatory firing rates, all the w_{QR} are random weights, β is a parameter that adjusts the relative contribution of structured vs random, and $\tilde{\phi}_E$ represents the activation function of the neuron smoothed over by the gaussian noise. We've also assumed there is only one memory pattern, additional ones can be thought of as contributing to the random weights. We will assume the inhibitory dynamics are much quicker than the excitatory (PEL says inhibitory neurons really are quicker than excitatory, but not by much, and certainly not by enough to make this approximation very valid). In the balanced state this means we'll just set: $\nu_I = \frac{w_{IE}\nu_E}{w_{II}}$. This corresponds to ignoring the spread of firing rates.

This might seem like quite a shoddy model if we're trying to address the shortcomings of Hopfield networks, but it includes dynamics and that turns out to be the key detail for the point being made here. More realism can be added but it just adds to the maths without adding insight, [Roudi and Latham, 2007].

Looking at the random contribution to the excitatory dynamics:

$$\sqrt{K}(w_{EE}\nu_E - w_{EI}\nu_I) = \sqrt{K}(w_{EE} - \frac{w_{IE}w_{EI}}{w_{II}})\nu_E = \frac{\sqrt{K}\text{Det}(w)}{w_I I}\nu_E$$

Which shows that if the determinant of the weight matrix is negative, these will help to stabilise the fixed point. We'll just replace this term with $-\gamma\nu_E$, a generic stabilising factor, to get:

$$\tau_E \frac{d\nu_{E,i}}{dt} = \tilde{\phi}_E(-\gamma\nu_E + \frac{\beta}{Nf(N-f)}\sum_j \xi_i(\xi_j - f)\nu_{E,j}) - \nu_{E,i}$$

Now we're going to turn these N equations describing the motion of the excitatory activities into 2 decsribing the mean excitatory activity and the memory overlap. We define the memory overlap as:

$$m = \frac{1}{Nf(1-f)}\sum_j (\xi_j - f)\nu_{E,j}$$

So if $\nu_E = \xi$ this is 1, if it is a random f-sparse pattern $m = 0$, hence it is a good measure of memory overlap.

$$\tau_E \frac{dm}{dt} = \frac{\tau_E}{Nf(1-f)}\sum_j (\xi_j - f)\frac{d\nu_{E,j}}{dt} = \frac{1}{Nf(1-f)}\sum_j (\xi_j - f)\tilde{\phi}_E(-\gamma\nu_E + \xi_j\beta m) - m$$

Similarly:

$$\tau_E \frac{d\nu_E}{dt} = \frac{\tau_E}{N}\sum_j \frac{d\nu_{E,j}}{dt} = \frac{1}{N}\sum_j \tilde{\phi}_E(-\gamma\nu_E + \xi_j\beta m) - \nu_E$$

The sums over neuron index can be taken as averages over the random f-sparse pattern ξ, giving:

$$\tau_E \frac{dm}{dt} = \tilde{\phi}_E(-\gamma\nu_E + \beta m) - \tilde{\phi}_E(-\gamma\nu_E) - m$$

$$\tau_E \frac{d\nu_E}{dt} = \tilde{\phi}_E(-\gamma\nu_E) + f[\tilde{\phi}_E(-\gamma\nu_E + \beta m) - \tilde{\phi}_E(-\gamma\nu_E)] - \nu_E$$

We'll do one final relabelling:

$$\Delta\tilde{\phi}_E(\nu_E, m) = \tilde{\phi}_E(-\gamma\nu_E + \beta m) - \tilde{\phi}_E(-\gamma\nu_E)$$

Giving our final version of the equations with which we'll do nullcline analysis:

$$\tau_E \frac{dm}{dt} = \Delta\tilde{\phi}_E(\nu_E, m) - m$$

$$\tau_E \frac{d\nu_E}{dt} = \tilde{\phi}_E(-\gamma\nu_E) + f\Delta\tilde{\phi}_E(\nu_E, m) - \nu_E$$

We can then plot these nullclines:

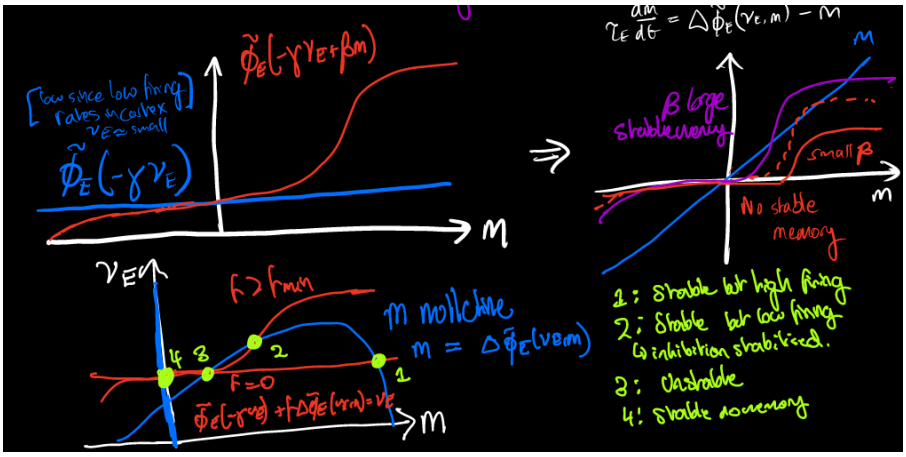


Figure 4.1: Top Left: We begin by working out the shape of the function $\Delta\tilde{\phi}_E$ as a function of m, which is a little weird. Top Right: using $\Delta\tilde{\phi}_E$ we graphically calculate the shape of the m nullcline for a couple of different β values. Increasing ν_E is kind of similar to decreasing β. Bottom: shows the nullcline plot, including the E-nullcline for two values of f - the sparsity. If $f \approx 0$ there is no dependence on memory overlap, and the stable fixed point with positive memory overlap is at very high m. This corresponds to very high firing rates in the subset of memory neurons, which is not observed experimentally. Only above a certain value of sparseness does a qualitatively different stable, memory-forming, solution appear on the unstable branch of the nullcline, this has reasonable firing rates.

As discussed in figure 4.1 hopfield style networks have to have a sparsity above a certain threshold in order to have a memory state with firing that matches cortical observations. This state has empirically much lower memory capacity (50 memories for 2400 neurons apparently). Peter concludes that, with this, Hopfield networks are dead and we have no good models of memory. This conclusion seems a little strong (after all, they've managed to overcome a wide variety of problems before), but its a nice conclusion.

5 Functional Models of Synaptic Plasticity

Aside from NMDA-mediated plasticity, long-term plasticity is generally not well understood and thus generally modelled more phenomenologically. Rather than thinking about neurotransmitter release probabilities and neurotransmitter receptor conductances, here we directly model changes to the "weight" $W_{ij} = \bar{g}_{ij} P_{rel}^{(ij)}$:

$$\tau_w \frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = f_{ij}(r_j, r_i)$$

with r_j, r_i for the pre- and post- synaptic cell activity (e.g. firing rate), respectively. τ_w sets the timescale of synaptic plasticity, analagous to a learning rate (large $\tau_w \rightarrow$ small learning rate).

Experimentally, a proxy for measuring W_{ij} is the change in membrane potential in a post-synaptic neuron induced by triggering an action potential in a pre-synaptic neuron, called the *post-synaptic potential (PSP) amplitude*. A classic experimental setup consists of inducing high-frequency ($\sim$ 100Hz) bursts of action potentials simultaneously in both the pre- and post- synaptic neurons and then measuring the PSP amplitude. Such a stimulation protocol will usually lead to a jump in the PSP amplitude that can last on the order of hours. However, if you block protein synthesis, the PSP amplitude decays back to its pre-protocol level soon afterwards. We thus distinguish between "early" and "late" *long-term potentiation (LTP)*, where late LTP requires protein synthesis and early LTP does not[6]. Similarly, *long-term depression (LTD)* can be triggered via a low-frequency ($\sim$ 2Hz) bursting protocol. D&A fig. 8.1 shows the classic picture of this, where LFP (local field potential) amplitude in a hippocampal slice was used as a proxy for PSP amplitude (note the initial transient increase after the high frequency protocol, reflecting early LTP).

We now turn to phenomenological models of synaptic plasticity that can produce both LTD and LTP, as well as an array of other experimental observations. To facilitate analysis, we begin by considering a single post-synaptic neuron with linear dynamics:

$$\tau \frac{\mathrm{d}v}{\mathrm{d}t} = -v + \mathbf{w}^{\mathrm{T}}\mathbf{u}$$

For the rest of this section I adopt the convention of v and u_i for pre- and post- synaptic neuron activity. Importantly, we additionally assume that synaptic plasticity occurs on a much slower timescale than the neural dynamics (i.e. $\tau << \tau_w$), such that we can assume that the post-synaptic activity has converged to its equilibrium

$$v = \mathbf{w}^{\mathrm{T}}\mathbf{u}$$

Later we consider feed-forward and recurrent networks of neurons.

5.1 Hebb Rule

One interpretation of the above observation is the occurence of Hebbian plasticity: "neurons that fire together wire together." Since the high-frequency stimulation is likely to induce post-synaptic firing, synapses should strengthen because of co-occurence of high pre- and post- synaptic activity. Conversely, low-frequency stimulation is less likely to induce post-synaptic spikes, leading to weakening of the synapses and thus LTD. This gives us the basic *Hebb learning rule*:

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = v\mathbf{u}$$

Averaging over pre-synaptic inputs from trial to trial (e.g. over a lifetime) and taking the equilibrium post-synaptic activity $v = \mathbf{w}^{\mathrm{T}}\mathbf{u} = \mathbf{u}^{\mathrm{T}}\mathbf{w}$, we have:

$$\tau_w \left\langle \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} \right\rangle = \langle \mathbf{u}v \rangle = \langle \mathbf{u}\mathbf{u}^{\mathrm{T}} \rangle \mathbf{w} = \mathbf{Q}\mathbf{w}$$

where $\mathbf{Q}$ is the correlation matrix over pre-synaptic activity input patterns. We thus call the Hebb learning rule a *correlation-based plasticity rule*.

It is easy to see that the Hebb rule will lead to LTP whenever pre- and post- synaptic activity is correlated, but it can't lead to LTD if the activity vectors are positive. For this, we introduce an activity threshold:

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = (v - \theta_v)\mathbf{u}$$

or

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = v(\mathbf{u} - \theta_{\mathbf{u}})$$

[6] This has lead to the *synaptic tagging hypothesis*, which postulates that simultaneous pre- and post- synaptic activity leads to the "tagging" of synapses, signalling the cell to insert more receptors there.

If the pre- or post- synaptic activity is now below its corresponding threshold, the derivative of $\mathbf{w}$ becomes negative and LTD occurs. A natural setting for this threshold is the mean background firing activity $\theta_v = \langle v \rangle, \theta_{\mathbf{u}} = \langle \mathbf{u} \rangle$. Again averaging over all inputs over time, the result of adding a threshold gives us a *covariance-based plasticity rule*:

$$\tau_w \left\langle \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} \right\rangle = \langle \mathbf{u}(v - \langle v \rangle) \rangle = \langle \mathbf{u} \left(\mathbf{u}^{\mathrm{T}}\mathbf{w} - \langle \mathbf{u}^{\mathrm{T}} \rangle \mathbf{w} \right) \rangle = \left(\langle \mathbf{u}\mathbf{u}^{\mathrm{T}} \rangle - \langle \mathbf{u} \rangle \langle \mathbf{u} \rangle^{\mathrm{T}} \right) \mathbf{w} = \mathbf{C}\mathbf{w}$$

where $\mathbf{C}$ is the covariance matrix of the input patterns. One can easily verify that using a pre-synaptic activity threshold $\theta_{\mathbf{u}} = \langle \mathbf{u} \rangle$ instead of a post-synaptic activity threshold leads to the same average dynamics. However, having one or the other threshold leads to entirely different biological predictions:

- If only a post-synaptic activity threshold θ_v is included, then plasticity is induced at the ith synapse (i.e. $\mathrm{d}w_i/\mathrm{d}t \neq 0$) iff there is pre-synaptic activity at that synapse (i.e. $u_i > 0$). This is called *homosynaptic plasticity*.
- If only a pre-synaptic activity threshold $\theta_{\mathbf{u}}$ is included, then plasticity is induced at the ith synapse whenever there is post-synaptic activity (i.e. $v > 0$), even if there is no pre-synaptic activity at that synapse (i.e. $u_i = 0$). This leads to *heterosynaptic plasticity*.
- If both thresholds are included, then the model counterintuitively predicts that there should be LTP when both pre- and post- synaptic activity are below threshold

Because the weight dynamics are linear, we can easily analyze the result of applying these simple Hebbian synaptic learning rules. We first note that because $\mathbf{C}$ is symmetric, its eigenvectors $\mathbf{e}_i$ are orthogonal and form a complete basis for the space of $\mathbf{w}$. We can thus write

$$\mathbf{w}(t) = \sum_i c_i(t)\mathbf{e}_i$$

where the coefficients are simply $c_i(t) = \mathbf{w}^{\mathrm{T}}\mathbf{e}_i$ when we assume the eigenvectors to be normalized ($\|\mathbf{e}_i\| = 1$). Solving the above differential equation for the covariance-based Hebb learning rule then gives us:

$$\mathbf{w}(t) = \sum_i c_i(0) e^{\frac{\lambda_i t}{\tau_w}} \mathbf{e}_i$$

where λ_i is the eigenvalue of $\mathbf{C}$ corresponding to eigenvector $\mathbf{e}_i$. As $t \to \infty$, the eigenvector with the largest eigenvalue λ_1 dominates, giving

$$\lim_{t \to \infty} \mathbf{w}(t) \propto \mathbf{e}_1$$

as long as $\mathbf{w}(0)$ is not perpendicular to $\mathbf{e}_1$ (such that $c_1(0) = 0$). Thus, this learning rule leads to post-synaptic activity v proportional to the projection of pre-synaptic input $\mathbf{u}$ onto the direction of maximum variance of the input patterns observed during learning, i.e. the principal eigenvector of the covariance matrix $\mathbf{C}$:

$$v \propto \mathbf{e}_1^{\mathrm{T}}\mathbf{u}$$

A similar analysis holds for the basic correlation-based Hebb rule with no thresholds, with the exact same result whenever the inputs have mean 0 such that $\mathbf{Q} = \mathbf{C}$ (see D&A fig. 8.4 for the difference whenever $\mathbf{Q} \neq \mathbf{C}$).

This analysis only holds, however, if the weights are allowed to change unboundedly, which does not occur with real synapses. Not only can they not grow unboundedly, but they cannot switch from excitatory to inhibitory. Constraining all synapses to be excitatory, we should thus impose a lower bound at 0 along with a reasonable upper bound. In this case, the above theoretical result will still hold as long as the initial condition $\mathbf{w}(0)$ is far enough away from any of the boundaries such that there is enough time for $\mathbf{e}_1$ to dominate the dynamics.

In fact, the consideration of bounding synaptic weight changes brings up two general problems with the basic Hebb rule:

1. Without any bounds on the size of the w_i, the Hebb rule is unstable:

$$\tau_w \frac{\mathrm{d}\|\mathbf{w}\|}{\mathrm{d}t} = 2\mathbf{w}^{\mathrm{T}}\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = 2\mathbf{w}^{\mathrm{T}}\mathbf{C}\mathbf{w}$$

   which is necessarily always positive since $\mathbf{C}$ is a covariance matrix and therefore positive semi-definite. In other words, as long as the learning rule is in effect the weights will continue increasing. In a network setting, this will quickly lead to runaway excitation.

2. Since all synaptic weights w_i are allowed to grow unboundedly simultaneously, there is no competition between them. In light of the above the result, in a population of neurons receiving the same feed-forward input, Hebbian learning on the feed-forward weights will lead to a highly redundant representation whereby each neuron is activated in exactly the same way by each input (i.e. proportional to its projection onto the principal eigenvector of $\mathbf{C}$). This greatly limits the power of Hebbian learning in a network.

We now turn to two extensions of the basic Hebb rule that fix these issues.

5.2 BCM rule

The BCM rule (Bienenstock, Cooper & Munro, 1982) addresses the stability and competition issues by introducing a dynamic "sliding threshold":

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = v\mathbf{u}(v - \theta_v)$$
$$\tau_\theta \frac{\mathrm{d}\theta_v}{\mathrm{d}t} = v^2 - \theta_v$$

where $\tau_\theta < \tau_w$. Since an increase in $\|\mathbf{w}\|$ leads to an increase in v, the sliding threshold enforces stability by quickly and strongly increasing the threshold for LTP to prevent the weights from increasing further. Furthermore, a large increase in w_i can make this happen in the absence of growth in the other $w_{j\neq i}$, thus preventing them from growing by pushing up the threshold, effectively implementing competition between weights.

5.3 Synaptic Normalization

The BCM rule effectively implements weight stability and competittion by using the post-synaptic activity v as a proxy for the size of the weights. Alternatively, we could directly address the weight strengths by directly constraining the L_p norm $\|\mathbf{w}\|_p = \sum_i w_i^p$. This is called *synaptic normalization*. Constraining the L_1 norm leads to *subtractive normalization*, whereas constraining the L_2 norm leads to *multiplicative normalization*.

5.3.1 Subtractive Normalization

The learning rule that constrains the L_1 norm of $\mathbf{w}$ is given by

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = v(\mathbf{u} - \bar{u}\mathbf{1})$$

where $\mathbf{1}$ is a vector of ones and

$$\bar{u} = \frac{1}{N_u}\sum_{i=1}^{K} u_k = \frac{\mathbf{1}^\mathrm{T}\mathbf{u}}{N_u} = \frac{\|\mathbf{u}\|_1}{N_u}$$

is a scalar, where N_u is the number of pre-synaptic inputs. It is easy to verify that this rule constrains the total sum of synaptic weights:

$$\frac{\mathrm{d}\|\mathbf{w}\|_1}{\mathrm{d}t} = \sum_i \frac{\mathrm{d}w_i}{\mathrm{d}t} = v\sum_i u_i - vN_u\bar{u} = 0$$

Since the same quantity $v\bar{u}$ is being subtracted from the derivatives of each weight $\frac{\mathrm{d}w_i}{\mathrm{d}t}$, this synaptic normalization rule is termed subtractive.

As above, we consider this rule in the expectation over inputs, plugging in the equilibrium value for v:

$$\tau_w \left\langle \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} \right\rangle = \langle v\mathbf{u}\rangle - \langle \bar{u}v\rangle\mathbf{1} = \langle \mathbf{u}\mathbf{u}^\mathrm{T}\rangle\mathbf{w} - \frac{1}{N_u}\mathbf{1}^\mathrm{T}\langle \mathbf{u}\mathbf{u}^\mathrm{T}\rangle\mathbf{w}\mathbf{1} = \mathbf{Q}\mathbf{w} - \frac{\mathbf{1}^\mathrm{T}\mathbf{Q}\mathbf{w}}{N_u}\mathbf{1}$$

Taking the eigenvalue expansion of $\mathbf{Q}$ and recalling that its eigenvalues are orthogonal (because $\mathbf{Q}$ is symmetric) such that $\mathbf{w}(t) = \sum_i c_i(t)\mathbf{e}_i, c_i(t) = \mathbf{e}_i^\mathrm{T}\mathbf{w}$, we have:

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \sum_{i=1}^{N_u} \lambda_i c_i(t)\mathbf{e}_i - \frac{\lambda_i c_i(t)\mathbf{1}^\mathrm{T}\mathbf{e}_i}{N_u}\mathbf{1}$$

Recalling that the orthogonality of the eigenvectors of $\mathbf{Q}$ implies $\mathbf{e}_i^\mathrm{T}\mathbf{e}_j = \delta_{ij}$, we note that we have a differential equation for each $c_j(t)$:

$$\tau_w \frac{\mathrm{d}c_j}{\mathrm{d}t} = \tau_w \mathbf{e}_j^\mathrm{T}\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \lambda_j c_j(t) - \frac{\sum_{i=1}^{N_u}\lambda_i c_i(t)\mathbf{1}^\mathrm{T}\mathbf{e}_i}{N_u}\mathbf{e}_j^\mathrm{T}\mathbf{1} = \lambda_j c_j(t) - \frac{\sum_{i=1}^{N_u}\lambda_i c_i(t)\mathbf{1}^\mathrm{T}\mathbf{e}_i}{\sqrt{N_u}}\cos\theta_j$$

where θ_j is the angle between the vectors $\mathbf{e}_j$ and $\mathbf{1}$. We thus see that the subtractive normalization only operates on directions of growth of $\mathbf{w}$ close to the identity line $\mathbf{1}$, i.e. directions $\mathbf{e}_j$ in which all the weights grow at about the same rate. Directions of growth $\mathbf{e}_j$ perpendicular to $\mathbf{1}$ are left unaffected, since the normalization term in the derivative of the corresponding coefficient c_j will be 0, leaving only standard Hebbian dynamics $\tau_w \frac{\mathrm{d}c_j}{\mathrm{d}t} = \lambda_j c_j(t)$ (exponential growth with rate equal to the corresponding eigenvalue). Consider the case where $\mathbf{e}_j \propto \mathbf{1}$, such that $\cos\theta_i = \delta_{ij} \Leftrightarrow \mathbf{e}_i^{\mathrm{T}}\mathbf{1} = \delta_{ij}\sqrt{N_u}$. We then have:

$$\begin{aligned}
\tau_w \frac{\mathrm{d}c_i}{\mathrm{d}t} &= \lambda_i c_i(t) - \lambda_j c_j(t)\delta_{ij} = (1-\delta_{ij})\lambda_i c_i(t) \\
&\Rightarrow c_i(t) = c_i(0) e^{\frac{(1-\delta_{ij})\lambda_i t}{\tau_w}} \\
&\Rightarrow \mathbf{w}(t) = c_j(0)\mathbf{e}_j + \sum_{i\neq j} c_i(0) e^{\frac{\lambda_i t}{\tau_w}} \mathbf{e}_i
\end{aligned}$$

If $\mathbf{e}_j \propto \mathbf{1}$ happens to be the principal eigenvector of $\mathbf{Q}$, then in the limit $t \to \infty$ $\mathbf{w}$ will grow in the direction of the eigenvector with the second highest eigenvalue. Otherwise, the long run limit is unaffected by subtractive normalization. The above analysis is easily generalized to the covariance-based Hebb rule by simply incorporating a pre- or post- synaptic activity threshold.

We can use this rule to model ocular dominance in a post-synaptic neuron. Consider two inputs $\mathbf{u} = \begin{bmatrix} u_R & u_L \end{bmatrix}^{\mathrm{T}}$ coming from each eye. Assuming the statistics to each eye are the same, we have:

$$\mathbf{Q} = \begin{bmatrix} \langle u_R u_R \rangle & \langle u_R u_L \rangle \\ \langle u_R u_L \rangle & \langle u_L u_L \rangle \end{bmatrix} = \begin{bmatrix} q_v & q_c \\ q_c & q_v \end{bmatrix}$$

which has two eigenvectors

$$\begin{aligned}
\mathbf{e}_1 &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \lambda_1 = q_v + q_c \\
\mathbf{e}_2 &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \lambda_2 = q_v - q_c
\end{aligned}$$

Since there is likely to be some covariance between the inputs to the two eyes, $q_c > 0 \Rightarrow \lambda_1 > \lambda_2$ and $\mathbf{e}_1$ is the principal eigenvector. Thus, without any normalization, Hebbian learning of the feed-forward input weights will lead to strengthened input connections from both eyes. But since $\mathbf{e}_1 \propto \mathbf{1}$, subtractive normalization can force the weight vector to grow in the direction of $\mathbf{e}_2$ instead. Imposing a lower bound of 0 on the weights, this will lead to a 0 weight for one of the eyes and a large weight for the other (equal to the sum over the initial weights, since the subtractive normalization enforces $\frac{\mathrm{d}\|\mathbf{w}\|_1}{\mathrm{d}t} = 0$), thus producing ocular dominance in the post-synaptic activity.

Now for some caveats. From a biological perspective, the subtractive normalization rule is difficult to reconcile with known synaptic mechanics since it requires normalization by a global signal $\bar{u}$. In other words, the weight dynamics at synapse i requires knowledge of the inputs at all other synapses. It is unclear how this could happen biologically. Another characteristic of subtractive normalization is that the competition between weights is very strong, since the global subtractive term is relatively stronger for weights with smaller derivatives. Indeed, without a lower bound on the weights, this rule will lead to driving weights below 0. With a bound at 0, it will tend to produce a solution with 1 big positive weight and all others set to 0.

5.3.2 Multiplicative Normalization

Also called the Oja Rule (Oja, 1982), the synaptic learning rule that constrains the L_2 norm of $\mathbf{w}$ is given by

$$\tau_w \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = v\mathbf{u} - \alpha v^2 \mathbf{w}$$

where α is a positive constant that bounds the L_2 norm of $\mathbf{w}$:

$$\frac{\mathrm{d}\|\mathbf{w}\|_2}{\mathrm{d}t} = 2\mathbf{w}^{\mathrm{T}}(v\mathbf{u} - \alpha v^2 \mathbf{w}) = 2v^2(1 - \alpha\|\mathbf{w}\|_2)$$

which converges to $\|\mathbf{w}\|_2 = \frac{1}{\alpha}$. Since the normalization term $\alpha v^2 w_i$ is proportional to each weight, we call this type of synaptic normalization multiplicative.

The Oja rule leaves our result for covariance-based synaptic learning rules intact, with the addition of a guarantee of convergence. Specifically, one can easily verify that this rule will lead to $\mathbf{w}(t) \to \mathbf{e}_1/\sqrt{\alpha}$ as $t \to \infty$. The Oja rule is also more biologically feasible than subtractive normalization since it only requires local signals for each synaptic weight change.

5.4 Hebbian Networks

As briefly mentioned above, our analysis of the basic Hebb rule implies that if we impose it on a feed-forward network of neurons with no recurrent connections, they will all learn the same input weights, aligned to the principal eigenvector of $\mathbf{Q}$ (or $\mathbf{C}$). This results in a completely redundant representation of the input where all post-synaptic neurons have exactly the same response to every input. By adding recurrent connections to the network, we can hope to avoid this.

We consider a linear recurrent network, of the form

$$\tau \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = -\mathbf{v} + \mathbf{W}\mathbf{u} + \mathbf{M}\mathbf{v}$$

where $\mathbf{W}$ are the feed-forward connection weights and $\mathbf{M}$ are the recurrent connection weights. Again assuming $\tau << \tau_w$, we will take the network activity at its stable equilibrium [7]:

$$\mathbf{v} = \mathbf{W}\mathbf{u} + \mathbf{M}\mathbf{v} \Leftrightarrow \mathbf{v} = \mathbf{K}\mathbf{W}\mathbf{u}$$

where $\mathbf{K} = (\mathbf{I} - \mathbf{M})^{-1}$. The basic Hebb rule is then:

$$\tau_w \frac{\mathrm{d}\mathbf{W}}{\mathrm{d}t} = \langle \mathbf{v}\mathbf{u}^{\mathrm{T}} \rangle = \mathbf{K}\mathbf{W}\mathbf{Q}$$

Under certain conditions on $\mathbf{K}$, we can use subtractive normalization to generate ocular dominance bands in a population of neurons arranged in 1D space. Consider visual input from each eye $\mathbf{u} = \begin{bmatrix} u_R & u_L \end{bmatrix}^{\mathrm{T}}$ as above, with $\mathbf{Q}$ again as before. In this case, $\mathbf{W} = \begin{bmatrix} \mathbf{w}_R & \mathbf{w}_L \end{bmatrix}$ is an $N \times 2$ matrix, where N is the number of neurons in the population. Using Hebbian learning with subtractive normalization for each set of feed-forward weights $\begin{bmatrix} w_{Ri} & w_{Li} \end{bmatrix}$ to each neuron (i.e. each row of $\mathbf{W}$) ensures that $\mathbf{w}_+ = \mathbf{w}_R + \mathbf{w}_L$ be constant over time. Thus, we can write

$$\tau_w \frac{\mathrm{d}\mathbf{w}_+}{\mathrm{d}t} = \tau_w \frac{\mathrm{d}\mathbf{w}_R}{\mathrm{d}t} + \tau_w \frac{\mathrm{d}\mathbf{w}_L}{\mathrm{d}t} = 0$$
$$\Rightarrow \tau_w \frac{\mathrm{d}\mathbf{w}_-}{\mathrm{d}t} = \tau_w \frac{\mathrm{d}\mathbf{w}_R}{\mathrm{d}t} - \tau_w \frac{\mathrm{d}\mathbf{w}_L}{\mathrm{d}t} = (q_v - q_c)\mathbf{K}\mathbf{w}_- \neq 0$$

as long as the weights are changing. Components of $\mathbf{w}_- = \mathbf{w}_R - \mathbf{w}_L$ that are highly positive indicate post-synaptic activity v_i dominated by right eye input, whereas those that are highly negative reflect dominance by left eye input. If $\mathbf{K}$ is translation invariant - $K_{ij} = f(|i-j|)$ where the indexes i, j designate each post-synaptic neuron's position in 1D space as well as their position in the matrix - then one can show that the principle eigenvectors of $\mathbf{K}$ lead to ocular dominance bands (see D&A pgs. 303-304).

An alternative approach to ensure Hebbian learning doesn't lead to redundant representation is to impose a non-linearity in the network that induces competition in post-synaptic activity. Purely linear recurrent interactions induce very weak competition, limiting the amount of differentiation achievable through Hebbian learning. So we introduce divisive normalization in post-synaptic activity:

$$v_i = \sum_j M_{ij} z_j$$
$$z_i = \frac{(\mathbf{w}_i^{\mathrm{T}}\mathbf{u})^\alpha}{\sum_j (\mathbf{w}_j^{\mathrm{T}}\mathbf{u})^\alpha}$$

where α controls the strength of competition. For large α, for example, only z_i with largest $\mathbf{w}_i^{\mathrm{T}}\mathbf{u}$ survive, with all others reduced to near 0. While the latter equation implements competition for feedforward input between all post-synaptic neurons, the first equation allows for cooperation between nearby neurons when the recurrent connections $\mathbf{M}$ are excitatory and local. Running Hebbian learning on the feedforward weights $\mathbf{w}_i$ in this setting is called *competitive Hebbian learning*. The nonlinear competition allows for strong differentiation of the post-synaptic neurons, which depend on higher order statistics beyond covariances (so in this case we cannot analyze the weight dynamics purely in terms of the eigenvectors of the correlation/covariance matrix). Indeed, the basic Hebb rule in this setting can produce ocular dominance bands in a population without the need for subtractive normalization.

By abstracting away from any physical grounding, competitive Hebbian learning can allow the formation of highly structured cortical maps. In such models, called *competitive feature-based models*, we assume the inputs

[7] Rewriting the dynamics as

$$\tau \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = (\mathbf{M} - \mathbf{I})\mathbf{v} + \mathbf{W}\mathbf{u}$$

we see that the recurrent network will have stable dynamics as long as the largest eigenvalue of $\mathbf{M} - \mathbf{I}$ is less than 0, i.e. the largest eigenvalue of $\mathbf{M}$ is less than 1.

u_i take on the values of different parameters of the stimulus (e.g. ocularity, orientation, location), such that N_u is the number of parameters used to characterize the stimulus and W_{ij} directly represents the selectivity of neuron i for parameter j. We can then accordingly modify our post-synaptic activity variable, e.g. by assuming homogenous Gaussian tuning curves for each parameter:

$$z_i = \frac{\exp\left[-\sum_j \frac{(u_j - W_{ij})^2}{2\sigma_j^2}\right]}{\sum_n \exp\left[-\sum_j \frac{(u_j - W_{nj})^2}{2\sigma_j^2}\right]}$$

The cooperation can then be introduced either via recurrent connections (self-organizing map) or via an extra cooperative term in the learning rule (elastic net). In the *self-organizing map* model, we assume the above equation for v_i, with $\mathbf{M}$ such that all recurrent connections are excitatory and local, to generate similar selectivities in nearby neurons. We then modify the basic Hebb rule to push a neuron's selectivity towards those inputs that excite it most:

$$\tau_w \frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = \langle v_i(u_j - W_{ij})\rangle$$

This is called the *feature-based learning rule*. The alternative is the *elastic net* model, which assumes $v_i = z_i$ and instead introduces an extra term in the learning rule to encourage similar selectivities between nearby neurons:

$$\tau_w \frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = \langle v_i(u_j - W_{ij})\rangle + \beta \sum_{n \in \text{neighborhood of } i} W_{nj} - W_{ij}$$

This is called the *elastic net rule*. Using ocularity, orientation, and location as stimulus parameters, these two models can produce orientation and ocular dominance cortical maps akin to those found in primates (i.e. with ocular dominance bands and iso-orientation countours with pinwheels; D&A fig. 8.10).

Note that the above models of plasticity in a network assume fixed non-plastic recurrent weights $\mathbf{M}$. Instead of making this strong assumption (likely to be false), we could instead apply synaptic learning rules to learn both the feedforward and recurrent weights simultaneously. By using a Hebbian rule for the feedforward weights and an anti-Hebbian rule for the recurrent weights, we can then hope to decorrelate the post-synaptic activity to avoid redundant representation:

$$\tau_w \frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = \langle v_i u_j\rangle - \alpha\langle v_i^2\rangle W_{ij}$$
$$\tau_M \frac{\mathrm{d}M_{ij}}{\mathrm{d}t} = -\langle v_i v_j\rangle + \beta M_{ij}$$

with M_{ii} set constantly to 0. By incorporating multiplicative normalization for the feedforward weights (i.e. Oja rule) and picking a suitable τ_M and β, this rule results in individual feedforward weights $\mathbf{w}_i$ (i.e. rows of $\mathbf{W}$) aligned to different eigenvectors of $\mathbf{Q} = \langle \mathbf{u}\mathbf{u}^{\mathrm{T}}\rangle$, and $M_{ij} = 0$. Indeed, anti-Hebbian plasticity is thought to be the predominant form of plasticity at the synapses of parallel fibres onto Purkinje cells in the cerebellum.

Alternatively, we could directly address the issue of redundant representation in a network by deriving a learning rule for $\mathbf{M}$ that sets correlations in post-synaptic activity to 0, i.e.

$$\langle \mathbf{v}\mathbf{v}^{\mathrm{T}}\rangle = c\mathbf{I}$$

By substituting in the equilibrium value of $\mathbf{v}$, we have

$$\begin{aligned} \langle \mathbf{v}\mathbf{v}^{\mathrm{T}}\rangle &= \mathbf{K}\mathbf{W}\langle \mathbf{u}\mathbf{v}^{\mathrm{T}}\rangle = c\mathbf{I} \\ \Leftrightarrow c\mathbf{K}^{-1} &= \mathbf{W}\langle \mathbf{u}\mathbf{v}^{\mathrm{T}}\rangle \\ \Leftrightarrow \mathbf{M} &= \mathbf{I} - c^{-1}\mathbf{W}\langle \mathbf{u}\mathbf{v}^{\mathrm{T}}\rangle \end{aligned}$$

This gives us the anti-Hebbian *Goodall rule*:

$$\tau_M \frac{\mathrm{d}\mathbf{M}}{\mathrm{d}t} = \mathbf{I} - \mathbf{W}\langle \mathbf{u}\mathbf{v}^{\mathrm{T}}\rangle - \mathbf{M}$$

which, if it converges, will converge to the desired matrix derived above. In addition to decorrelating post-synaptic activity, it also equates the individual variances. The Goodall rule, however, is non-local because of the $\mathbf{Wu}$ term, which implies the dynamics of any *recurrent* synapse weight M_{ij} will depend on the activity at all the *feedforward* synapses on neuron i (although this could be easily addressed by assuming the feedforward mapping is the identity, i.e. $\mathbf{W} = \mathbf{I}$). Our above result also requires the diagonal of $\mathbf{M}$ to be non-zero, so autapses are implied. These two characteristics make this rule somewhat biologically implausible. Computationally, it is also limited by its purely linear dynamics, which limit it to only removing redundancies in second-order statistics.

5.5 Plasticity for Supervised Learning

So far, we have only considered unsupervised learning rules, where the network is asked to learn some useful set of weights given a set of training inputs. Although somewhat less relevant biologically, we might also ask how Hebbian learning performs in a supervised setting, where we want the weights to be modified such that a given target output $\mathbf{v}$ is achieved in response to a corresponding input $\mathbf{u}$. It turns out that an interplay of Hebbian and anti-Hebbian learning is again crucial. We first analyze the case of applying Hebbian learning to see its limitations, which we then address with anti-Hebbian learning.

We first consider the problem of binary classification, using the *perceptron* binary classifier model:

$$v = \begin{cases} +1 & \text{if } \mathbf{w}^{\mathrm{T}}\mathbf{u} - \gamma \geq 0 \\ -1 & \text{else} \end{cases}$$

where the components of $\mathbf{u}$ are also either 1 or -1. We analyze the simplified case of $\gamma = 0$, using the basic Hebb rule with multiplicative normalization:

$$\tau_w \frac{\mathrm{d}w}{\mathrm{d}t} = \langle v\mathbf{u} \rangle - \alpha\mathbf{w} = \frac{1}{D}\sum_{i=1}^{D} v^{(i)}\mathbf{u}^{(i)} - \alpha\mathbf{w}$$

where D is the number of observed data points. Note that the post-synaptic activity is indexed as well, since in the supervised setting we observe input-output pairs $(v^{(i)}, \mathbf{u}^{(i)})$ rather than just inputs. We assume a random set of such pairs and ask how well the perceptron has learned the trained associations on convergence of the Hebbian learning rule to

$$\mathbf{w} = \frac{1}{\alpha D}\sum_{i=1}^{D} v^{(i)}\mathbf{u}^{(i)}$$

Setting $\alpha = \frac{K}{D}$, where K is the number of input components, we have that for an arbitrary input $\mathbf{u}^{(j)}$ from the training data set, the perceptron will output

$$v\left(\mathbf{u}^{(j)}\right) = \mathbf{w}^{\mathrm{T}}\mathbf{u}^{(j)} = \frac{1}{K}\sum_{i=1}^{D} v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} = \frac{v^{(j)}{\mathbf{u}^{(j)}}^{\mathrm{T}}\mathbf{u}^{(j)}}{K} + \frac{1}{K}\sum_{i\neq j}^{D} v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} = v^{(j)} + \eta^{(j)}$$

which looks like a noisy version of the desired output. Indeed, in the limit of large D, the Central Limit Theorem tells us that

$$\lim_{D\to\infty} \frac{K}{D-1}\eta^{(j)} = \lim_{D\to\infty} \frac{1}{D-1}\sum_{i\neq j}^{D} v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{D-1}\right)$$

where μ, σ^2 are the mean and variance of each of the $D-1$ terms in the sum[8]:

$$\mu = \langle v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} \rangle = 0$$
$$\sigma^2 = \mathrm{Var}(v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)}) = K$$

[8] I found this to be a surprisingly non-trivial result, so I derive it here. Let z be the number of positive terms in the sum implied by the dot product ${\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)}$. Since we have assumed each input component and output to be random and independent, $z \sim \mathrm{Binom}(K, 0.5)$. Recalling that $u_k \in \{+1, -1\}$, we can rewrite the dot product as ${\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} = z - (K - z) = 2z - K$. We thus have:

$$\begin{aligned}
\mu &= \langle v^{(i)}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)} \rangle \\
&= 2\langle v^{(i)} z \rangle - \langle (v^{(i)}) \rangle K \\
&= 2\left(\sum_{a=0}^{K} P(v^{(i)} = +1)P(z = a)a + \sum_{a=-K}^{0} P(v^{(i)} = -1)P(z = -a)a\right) \\
&= \sum_{a=0}^{K} P(z = a)a - \sum_{a=0}^{K} P(z = a)a = 0 \\
\sigma^2 &= \langle ({v^{(i)}}{\mathbf{u}^{(i)}}^{\mathrm{T}}\mathbf{u}^{(j)})^2 \rangle - \mu^2 \\
&= \langle {v^{(i)}}^2 (2z - K)^2 \rangle \\
&= 4\langle z^2 \rangle - 4K\langle z \rangle + K^2 \quad \text{since } {v^{(i)}}^2 = 1 \text{ always} \\
&= 4(\mathrm{Var}[z] + (0.5K)^2) - 4K(0.5K) + K^2 \\
&= 4(0.25K + 0.25K^2) - 2K^2 + K^2 \\
&= K
\end{aligned}$$

Thus, for large D,

$$\eta^{(j)} \sim \mathcal{N}\left(0, \frac{D-1}{K}\right)$$

since $\text{Var}[\eta] = \left(\frac{D-1}{K}\right)^2 \text{Var}\left[\frac{K}{D-1}\eta\right] = \left(\frac{D-1}{K}\right)^2 \frac{K}{D-1} = \frac{D-1}{K}$. In this limit, we can get a closed form expression for the probability that the perceptron correctly classifies an arbitrary input from the training set:

$$\begin{aligned} P\left(v(\mathbf{u}^{(j)}) = v^{(j)}\right) &= P(v^{(j)} = 1)P(v(\mathbf{u}^{(j)}) = 1|v^{(j)} = 1) + P(v^{(j)} = -1)P(v(\mathbf{u}^{(j)}) = 0|v^{(j)} = -1) \\ &= 0.5P(\eta^{(j)} > -1) + 0.5P(\eta^{(j)} < 1) \\ &= \text{erf}\left(\sqrt{\frac{K}{D-1}}\right) \end{aligned}$$

where $\text{erf}(\cdot)$ is the error function, or standard Gaussian cumulative distribution. Of course, this gives us a measure only of the *storage* capabilities of the perceptron following Hebbian learning with a large training data set. It is thus intuitive that the perceptron should be able to store trained associations when the dimensionality of the input is greater than the number of inputs ($K > D$), in which case it is easy to find a manifold that can linearly separate the trained inputs[9]. We have no guarantees for *generalization* when the input associations have some structure to be learned - in fact, the setting of $K > D$ is likely to lead to overfitting. Furthermore, this is a highly restrictive setting: we need a large D for the above theoretical results (using the CLT) to hold, and an even larger K to achieve good storage. One of the main reasons for this highly limited performance is that the Hebbian learning rule is sensitive only to correlation structure in the training data, with no regard to the actual responses of the perceptron. We might hope to do better by gauging the weight updates according to the direction of any errors the perceptron is making. This leads to the *perceptron learning rule*:

$$\mathbf{w} \to \mathbf{w} + \epsilon(v^{(i)} - v(\mathbf{u}^{(i)}))\mathbf{u}^{(i)}, \quad \gamma \to \gamma - \epsilon(v^{(i)} - v(\mathbf{u}^{(i)}))$$

which increases the weights whenever the perceptron misclassifies in the negative direction (-1 instead of $+1$, such that $v^{(i)} - v(\mathbf{u}^{(i)}) = 2 > 0$) and decreases them when it makes an error in the opposite direction (such that $v^{(i)} - v(\mathbf{u}^{(i)}) = -2 < 0$). If it correctly classifies the given input, the weights are left unchanged. Thus, this learning rule requires repeated exposure to all inputs, in which case it can be proved that it will converge to perfect classification whenever each class of inputs (i.e. $+1, -1$) are linearly separable. Note that if we expand the weight update rule by multiplying out the terms inside the parenthesis we get a Hebbian and an anti-Hebbian term.

We can get further insight into why we should include such anti-Hebbian learning by considering supervised learning compliment to classification: regression. Here, we want to approximate an arbitrary function $h(u)$ via a linear sum of *basis functions* $f_i(u)$:

$$h(u) \approx v(u) = \sum_i w_i f_i(u)$$

If the set of basis functions $\{f_i(\cdot)\}$ can represent a class of functions via a linear sum, we say that it is *complete* with respect to that class. If the set of corresponding weights $\{w_i\}$ is not unique for the given target function, then we say it is *overcomplete*. From a neural perspective, we might consider a population of N neurons with tuning curves $f_i(u)$ responding to a stimulus u. A downstream post-synaptic neuron that linearly sums its pre-synaptic inputs can then represent any function of the stimulus in a class such that the N tuning curves form a complete set of basis functions for it. More generally, for any arbitrary target function $h(u)$, we can hope that the post-synaptic activity $v(u)$ provides a good approximation of $h(u)$ if the synaptic weights $\mathbf{w}$ minimize the mean squared error over some set of N observed input stimuli, i.e.

$$\begin{aligned} 0 &= \frac{\partial}{\partial \mathbf{w}} \frac{1}{2} \sum_{i=1}^{N} \left(h(u^{(i)}) - v(u^{(i)})\right)^2 \\ &= \sum_{i=1}^{N} \left(h(u^{(i)}) - v(u^{(i)})\right) \mathbf{f}(u^{(i)}) \\ \Leftrightarrow \mathbf{w} &= \left(\sum_{i=1}^{N} \mathbf{f}(u^{(i)})\mathbf{f}(u^{(i)})^{\mathrm{T}}\right)^{-1} \sum_{i=1}^{N} h(u^{(i)})\mathbf{f}(u^{(i)}) \\ &= \left\langle \mathbf{f}(u)\mathbf{f}(u)^{\mathrm{T}} \right\rangle^{-1} \left\langle h(u)\mathbf{f}(u) \right\rangle \end{aligned}$$

[9] Note that the perceptron is only capable of replicating associations that are linearly separable (which implies a maximum of $2K$ different associations).

where $\mathbf{f}(u) = \begin{bmatrix} f_1(u) & f_2(u) & \dots & f_N(u) \end{bmatrix}^{\mathrm{T}}$. Can we learn such a set of weights with a Hebbian learning rule? Consider again the basic Hebb rule with multiplicative normalization, which in this case would coverge to:

$$\mathbf{w} = \frac{1}{\alpha} \langle h(u) \mathbf{f}(u) \rangle$$

In light of our above result, this set of synaptic weights will yield a good approximation of $h(u)$ if $\left\langle \mathbf{f}(u) \mathbf{f}(u)^{\mathrm{T}} \right\rangle = \alpha \mathbf{I}$, i.e. if the post-synaptic activity is decorrelated across the set of observed stimuli. Given a fixed set of stimuli, this is called a *tight frame* condition on the basis functions. Specifically, it implies that for any two observed stimuli $u^{(i)} \neq u^{(j)}$, $\mathbf{f}(u^{(i)})$ is orthogonal to $\mathbf{f}(u^{(j)})$. Thus, Hebbian learning is only effective under highly restrictive conditions. We see now that the issue stems from the fact that the Hebbian learning rule does not account for correlations in the responses to different inputs. Indeed, our analysis shows that it will only perform well if post-synaptic responses are completely decorrelated. In this light, it is easy to see how adding an anti-Hebbian term should alleviate this problem by decorrelating the post-synaptic responses. In fact, an anti-Hebbian term falls out of the derivation the *delta rule*, which is exactly equivalent to stochastic gradient ascent on the mean squared error:

$$\mathbf{w} \to \mathbf{w} + \epsilon \left(h(u^{(i)}) - v(u^{(i)}) \right) \mathbf{f}(u^{(i)})$$

This rule is directly analagous to the perceptron learning rule above, incorporating an anti-Hebbian term that makes it sensitive to the direction of the post-synaptic neuron's errors to enforce the appropriate weight changes.

Turning now to stochastic networks, we can derive an analagous weight update rule for density estimation with a Boltzmann machine. In this case, minimizing the KL divergence between the output distribution of a Boltzmann machine and some target conditional or joint distribution (equivalent to maximizing the likelihood of the observed training data) results in a very similar weight update rule that again consists of the difference between a Hebbian and anti-Hebbian term. It is thus called a *contrastive Hebb rule*. Due to the stochastic nature of the networks output, in this case the two terms are in fact implemented in separate phases, emphasizing the role of the anti-Hebbian learning in decorrelating the network output:

1. *Wake phase*: the Boltzmann machine is fed data, and the feed-forward and/or recurrent weights are updated with a Hebb rule. The network thus learns the correlation structure in the training data.

2. *Sleep phase*: the Boltzmann machine generates random samples in response to data inputs, and the weights are updated with an anti-Hebbian rule. The network thus modifies its weights to decorralte its stochastic output.

See D&A pgs 322-6 for more details.

6 Neural Networks & Bioplausibility

Neural networks are a bit of a big deal right now. They are very successful at learning a particular mapping from inputs to outputs thanks to an algorithm called backpropagation, which is basically an application of the chain rule (but is still a big deal). Since neural networks are inspired by neurons (it's in the name...) we might try and see what the success of deep learning can teach us about the brain.

Unfortunately the vanilla backpropagation algorithm is not biologically plausible, meaning its relevance to biological learning is questionable. To see why this is the case we will consider the simplest setup, a set of inputs and labels $\{\boldsymbol{x}_{0,i}, \boldsymbol{y}(\boldsymbol{x}_{0,i})\}_{i=1}^N$ are provided, and the neural network attempts to predict the label from the input using a layered structure:

$$\boldsymbol{x}_{l+1} = \phi(\boldsymbol{h}_l)$$
$$\boldsymbol{h}_l = \boldsymbol{W}_l \boldsymbol{x}_l$$
$$\hat{\boldsymbol{y}} = \phi(\boldsymbol{h}_L)$$

Where $\boldsymbol{W}_i$ are the weight matrices at each layer i that can be learnt during training, $\phi(.)$ is some elementwise non-linearity, and $\hat{\boldsymbol{y}}$ is the predicted label calculated from the last layer L. We want to make the neural network fit some function, in order to do that we follow a training procedure that minimises some measure of mismatch between the neural network's performance and the real behaviour, for example:

$$\mathcal{L}(\{\boldsymbol{x}_{0,i}, \boldsymbol{y}_i\}_{i=1}^N) = \sum_{i=1}^{N} |\boldsymbol{y}_i - \hat{\boldsymbol{y}}(\boldsymbol{x}_{0,i})|^2$$

In the simplest case we try to minimise this by taking gradient steps on this objective in weight space. These can be simply, but a little tediously calculated, using standard matrix differentiation.

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_l} \frac{\partial \boldsymbol{h}_l}{\partial \boldsymbol{W}_l} = \boldsymbol{\delta}_l \boldsymbol{x}_l^T$$

Where $\boldsymbol{\delta}_l$ is some error vector telling you the gradient of the loss with respect to later parts of the network. This can be iteratively calculated, from final layer backwards (hence the name backpropagation) by noticing:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_{l+1}} \cdot \frac{\partial \boldsymbol{h}_{l+1}}{\partial \boldsymbol{h}_l} = (\boldsymbol{W}_{l+1} \circ \phi'(\boldsymbol{h}_l))^T \cdot \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_{l+1}}$$

Where $\circ$ denotes an elementwise product. Again using our notation: $\boldsymbol{\delta}_l = \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_l}$, this gives a backpropagting equation:

$$\boldsymbol{\delta}_l = (\phi'(\boldsymbol{h}_l) \circ \boldsymbol{W}_l)^T \boldsymbol{\delta}_{l+1}$$

Which can then be used to produce a weight update equation, using some learning rate η:

$$\Delta \boldsymbol{W}_l = -\eta \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_l} = -\eta \boldsymbol{\delta}_l \boldsymbol{x}_l^T$$

And here we see our problem. In order to update the neuron's weights it needs access to the error vector $\boldsymbol{\delta}_l$, the inputs $\boldsymbol{x}_l$ are not a problem, the neuron has access to those. You might think that you could do a backward pass of the error vectors through the same network, however this requires that the neurons know the transpose of the weight matrix, which is a highly non-local thing. The neuron cannot access the requisite values of the weights connecting all neurons to all others, so would be unable to calculate the desired weight update.

As a result, there is now a cottage industry in designing biologically plausible alternative to back propagation that could conceivably work in biology, we'll run through a few of the big ideas here.

6.1 Feedback Alignment

The first one [Lillicrap et al., 2016] we consider gets around the problem of having to know the transpose of the outgoing weight matrix by replacing it with a random matrix:

$$\boldsymbol{\delta}_l = (\phi'(\boldsymbol{h}_l) \circ \boldsymbol{A})^T \boldsymbol{\delta}_{l+1}$$

Where $\boldsymbol{A}$ is some random matrix. This learns really well on simple problems, which is somewhat surprising! It seems that the network has enough expressive power (whatever that is) to both learn to use the random weights to send error signals and to solve the problem. It struggles on harder problems however.

6.2 Learn Feedback Matrices

The next iteration of these provides some mechanism to learn the feedback matrices, for example by simple hebbian learning:

$$\Delta \boldsymbol{A} = \eta \langle \boldsymbol{x}_l \boldsymbol{x}_{l+1}^T \rangle$$

Our goal is to make $\boldsymbol{A}$ look like $\boldsymbol{W}^T$, it turns out that if you add weight decay and some white noise into each layer this does the job:

$$\Delta \boldsymbol{A} = \eta \langle \boldsymbol{x}_l \boldsymbol{x}_{l+1}^T \rangle - \eta \boldsymbol{A} = \eta \langle \boldsymbol{x}_l \boldsymbol{x}_l^T \boldsymbol{W}_{l+1}^T \rangle - \eta \boldsymbol{A} = \eta \langle \boldsymbol{x}_l \boldsymbol{x}_l^T \rangle \boldsymbol{W}_{l+1}^T - \eta \boldsymbol{A} = \eta (\boldsymbol{W}_{l+1}^T - \boldsymbol{A})$$

Which does indeed have the desired weight matrix as the fixed point. This works nearly as well as backprop, but it not motivated by any biology. It's hard to know whether that matters.

6.3 Direct Feedback Alignment

There's another similar approach, where instead of passing the error back iteratively, there is a global error signal $\hat{\boldsymbol{y}} - \boldsymbol{y}$ that is projected to each of the different layers using a different random matrix $\boldsymbol{B}_l$, then:

$$\Delta \boldsymbol{W}_l = \boldsymbol{B}_l (\boldsymbol{y} - \hat{\boldsymbol{y}})$$

It turns out that this too works pretty well! The last layer in these cases is always able to learn in a biologically plausible and exact manner:

$$\Delta \boldsymbol{W}_L = -\eta \boldsymbol{x}_L (\boldsymbol{y} - \hat{\boldsymbol{y}})$$

So the thought is that the last layer does a lot of the hard work, and the previous layers just have to transmit enough information to allow the final layer to do its job.

6.4 Bottleneck Approach

The bottleneck approach is a more principled scheme developed by Tali Tishby. The basic idea is that as the representations pass from one layer to the next only the relevant pieces of information are being preserved, where relevancy is defined by how useful that piece of information is in performing the final classification. This is done by deriving learning rules that maximise the mutual information between each layer's representation and the final label while removing information in the current layer about the input:

$$\max I(\boldsymbol{x}_l, \boldsymbol{y}) - \beta I(\boldsymbol{x}_0, \boldsymbol{x}_l)$$

Calculating the mutual information is very difficult, so you have to come up with alternatives. Roman (at Gatsby) did this using HSIC which turned into a pretty good hebbian learning rule that requires the presence of a third factor, i.e. a third neuron.

6.5 Gated Linear Networks

I did not understand this one so well from lectures, so is even more likely to be wrong than the rest of it!

In this final scheme all the neurons from all the layers are trying to predict the output, so you get some much bigger loss function:

$$\mathcal{L} = \frac{1}{2} \sum_{i,l} (x_i^l - y^*(\boldsymbol{x}_0))^2$$

$$= \frac{1}{2} \sum_{i,l} (\sum_j W_{ij}^{l-1} x_j^{l-1} - y^*(\boldsymbol{x}_0))^2 = \sum_l \mathcal{L}_l$$

Where $y^*(\boldsymbol{x}_0)$ is, inconsistently, the target function. Now we can try and minimise this loss by taking gradient steps on one of the loss terms and we get:

$$\frac{\partial \mathcal{L}^{l-1}}{\partial W_{ij}^{l-1}} = (x_i^l - y^*) x_j^{l-1}$$

If we ignore the other parts of the loss this gives a hebbian learning rule!

Now the extra special addition: there is some additional input, $\boldsymbol{x}_{\text{in}}$, that acts a gating variable. Each neuron has a bank of weights from the set of inputs, and depending on the value of $\boldsymbol{x}_{\text{in}}$ one bank of weights or another is

used to compute the output at each timepoint. This ability to choose which weights you use at different places in the input allows the neuron to split the problem up, and classify differently depending on which region the input belongs to. Learning in this setting is provably convex and converges nicely, further it improves with the depth of the network.

This has been recently mapped into neuroscience, the role of weight banks is played by the different sections of the dendritic tree and the inhibitory neurons provide the gating variable, by turning on or off the activity of each of these branches. Each dendrite classifies the input differently and depending on the third input different sections of the tree are used.

7 Reinforcement Learning

This section assumes familiarity with reinforcement learning. It's mainly meant as reminders/review of biological RL and specific topics which I've seen on past exams. I deliberately don't go into much detail, as it's not something that's emphasized in the course.

7.1 Classical Conditioning

Classical/Pavlovian conditioning encompasses a variety of different training and testing procedures, but its defining feature (in contrast to instrumental conditioning) is that rewards and punishments are delivered to the the animal/agent independent of its actions. In the classic Pavlovian experiment, dogs are repeatedly fed just after a bell is rung. Eventually, the dogs begin to salivate in response to the sound of the bell, anticipating the arrival of food. In this case, the presentation of food is called the *unconditioned stimulus* and the salivation in response to the sight of food is the *unconditioned response*. The sound of the bell is then called the *conditioned stimulus* and the subsequent salivation the *conditioned response*. In the following sections, we construct models of how an animal acquires the expectation of the delivery of reward in response to stimuli.

7.1.1 The Rescorla-Wagner Rule

We describe the presence or absence of stimulus on a particular trial via the binary variable $u \in \{0, 1\}$, and model the value function (the expected reward) as a linear function of the presence of absence of the stimulus via

$$v = wu, \tag{47}$$

, for some weight w. The value of w is learned by minimizing the expected squared error $\left\langle (r - v_w(u))^2 \right\rangle$. Differentiating this expression with respect to w yields the *Rescorla-Wagner rule*:

$$\boxed{w \leftarrow w + \epsilon \underbrace{(r - v)}_{\delta} u}, \tag{48}$$

where ϵ is the learning rate and can be interpreted biologically as the associability of the stimulus with the reward. The prediction error δ can be interpreted as the firing rates of dopaminergic cells in the ventral tegmental area (VTA)—more on this in Section 7.1.3. Under the Rescorla-Wagner rule, it's easy to see that w will converge to the expected value of the reward, $\langle r \rangle$, at which point the average value of δ is zero. Given a stimulus that is always paired with reward, w will converge exponentially quickly to the average reward value. This phase of learning is called *acquisition*. If the reward is then never presented again in relationship to the stimulus, the w will then decay to zero exponentially quickly. This is called *extinction*. This relationship is visualized in Figure 7.1.

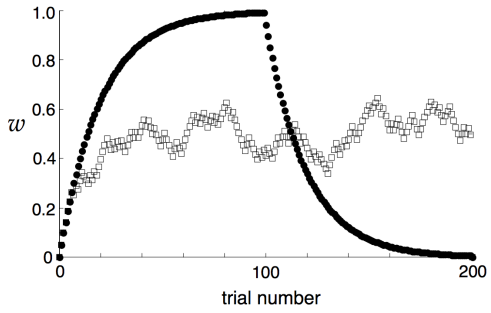


Figure 7.1: Acquisition and extinction in the Rescorla-Wagner rule. The filled circles denote the value of w when the reward is first presented every time the stimulus is, and then when the reward is never given with the stimulus again. The open squares show the path of w when the reward is presented stochastically half the time that the stimulus is presented—here, w fluctuates around $\langle r \rangle = 0.5$.

Blocking *Blocking* is a phenomenon in which two stimuli are presented together before the delivery of reward, but only after the animal has developed an association for one stimulus on its own. Then, when the two stimuli are presented separately, the animal will only display the conditioned response in reaction to the stimulus that was originally associated with reward—it has *blocked* an association from developing with the second stimulus. This effect can be explained with the vector form of the Rescorla-Wagner rule, with $\mathbf{w}, \mathbf{u} \in \mathbb{R}^d$, with $d = 2$ in this case. During the initial training phase, the rule will lead w_1 to converge on $\langle r \rangle$. Then when the second stimulus is presented along with the first, its weight starts at $w_2 = 0$, so the predicted reward is still equal to r: $v = w_1 u_1 + w_2 u_2 = \langle r \rangle$. Then $\delta = 0$, so no further learning takes place.

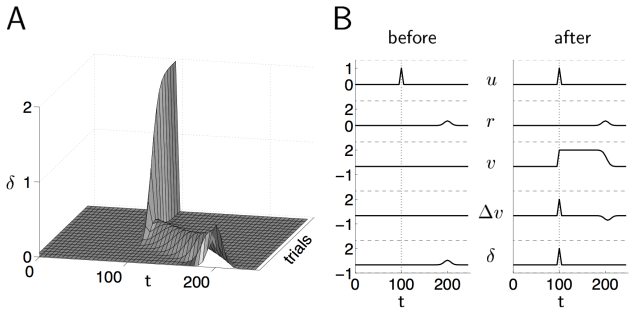


Figure 7.2: (A) The evolution of the TD error across time and trials in an experiment in which the stimulus is presented at time $t = 100$ and a reward is given at time $t = 200$. (B) The relevant values before and after training.

Overshadowing *Overshadowing* occurs when two stimuli are always presented together in training, but the prediction ends up being shared unequally, in that the weight associated with one stimulus is higher than the other. This can be explained by a generalization of the Rescorla-Wagner rule in which each weight receives its own learning rate. The weight with the higher learning rate reach a higher value than the second weight at convergence.

Secondary Conditioning While quite simple, the Rescorla-Wagner rule can explain a variety of phenomena seen in the learning behavior of animals. One pattern that cannot be reproduced by the Rescorla-Wagner rule, however, is that of *secondary conditioning*. In secondary conditioning, one stimulus is associated with reward, and then an association is learned between a second stimulus and the first. The animal then displays a conditioned response to the second stimulus, even though it has never been paired with the reward. This cannot be modeled with the Rescorla-Wagner rule—in fact, because the reward is never presented with the second stimulus, the delta rule would induce w_2 to become negative (inhibitory conditioning). This is related to the problem of delayed reward, and can be explained by *temporal difference* (TD) learning.

7.1.2 TD-Learning

To measure the reward across a trial, its useful to consider the *return* Z, defined as

$$Z := \sum_{t=0}^{T} \gamma^t r_t, \tag{49}$$

where $\gamma \in (0, 1)$ is a discount factor. It's useful, then, to redefine the value function as the expected return across a trial, rather than the expected value of the reward at the next time step:

$$v(u_t) := \langle Z \rangle = \left\langle \sum_{t=0}^{T} \gamma^t r_t \right\rangle. \tag{50}$$

For a linear value function $v(u_t) := wu_t$, the TD rule is given by

$$\boxed{w_t \leftarrow w_t + \epsilon \delta u_t, \qquad \text{with } \delta = r_t + \gamma v(u_{t+1}) - v(u_t)}. \tag{51}$$

We can see that higher values of γ correspond to higher prioritization given to the long-term expectation of future reward, while low values of γ encourage short-sighted behavior. Figure 7.2 shows the evolution of the relevant parameters in an experiment in which the stimulus is presented at time $t = 100$ and a reward is given at time $t = 200$. Over the course of learning, the TD error δ steadily shifts backward in time, ending up at $t = 99$. As it does so, the weight associated with each time step $200, 199, \ldots$ grows, leading to the elevation of the value function from the presentation of the reward on to the delivery of the reward (Figure 7.2B). The value difference $\Delta v_t := v_{t+1} - v_t = \delta_t - r_t$ is negative around $t = 200$ to compensate for the delivery of reward and maintain δ at zero.

Unlike the Rescorla-Wagner rule, TD learning can account for secondary conditioning. In this case, when a second stimulus s_2 is introduced before the first s_1 (which has become associated with the ensuing reward), the positive spike in δ_t at the time that s_1 is presented drives an increase in the value of the weight associated with s_2, thus establishing a positive association between s_2 and the reward through the same process that the association with s_1 was acquired.

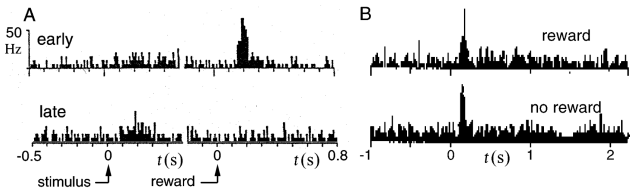


Figure 7.3: Activity of dopaminergic neurons in the VTA obtained from monkeys completing a reaction-time task. (A) The firing rate of a dopaminergic neuron in response to a stimulus followed by a reward early in training (top), and the same neuron late in training (bottom). (B) Dopaminergic responses when a reward is delivered as expected (top) and when it is not (bottom).

7.1.3 Dopamine

The prediction error δ plays a central role in both the Rescorla-Wagner rule and TD learning, and it therefore makes sense to look for a neural representation of this signal. One suggestion for this role are the dopaminergic neurons in the ventral tegmental area (VTA) of the midbrain. There is strong experimental evidence that dopamine is involved in reward learning. Many addictive drugs work by increasing the longevity of the dopamine released onto target structures such as the nucleus accumbens (an area often associated with feelings of pleasure).

In a series of studies performed by Schultz et al. in the 90s, monkeys were trained through instrumental conditioning to respond to stimuli such as lights and sounds in order to obtain food and drink rewards. The activities of cells in the VTA were recorded during learning, and examples are plotted in Figure 7.3. In (A), we can see that the increased firing rate moves from the time of the reward to (top) to the time of the stimulus (bottom), just as the prediction error δ does. Similarly, in the top panel of (B), we can see that when a reward is presented as expected, given the conditioned stimulus, there is no change in firing (indicating a prediction error of zero). On the bottom panel of (B), however, there is a decrease in firing rate below the baseline, indicating a negative prediction error. This is strong evidence that the firing rates of dopaminergic neurons in the VTA encode prediction errors.

7.2 Static Action-Choice

In static action-choice tasks, the reward or punishment immediately follows the action taken. *Indirect actor* methods learn a value function and then choose actions based on the expected rewards (ex. DP, Q-learning). *Direct actor* methods seek to optimize the policy by directly maximizing the expected return, usually via gradient ascent (ex. PPO, TRPO).

These ideas can be well-illustrated with the example of a bee foraging for nectar. If the bee follows an indirect actor paradigm, it can learn the expected volume of nectar at each flower it visits via a delta rule, and then base its actions off these estimates.

If the bee follows a direct actor paradigm, the choice of actions is based directly on maximizing the expected average reward, which can be done via gradient ascent. Compared to the indirect actor method, in this case, the direct actor method learns more slowly and is less adaptable if the expected nectar volumes in the environment changes.

7.3 Sequential Action-Choice

In sequential action-choice tasks, unlike static action-choice tasks, rewards may be delayed for several steps after an action is taken. A maze with rewards at the end is a classic example of such an environment. It's in this setting that the usage of dynamic programming and the Markov decision process frameworks that are hallmarks of RL become useful. In methods like policy iteration, we can subdivide the algorithm into the *critic*, which performs policy evaluation (e.g., with TD learning) and the *actor*, which maintains and improves the policy. Neurally, it has been suggested that the dorsal striatum, a part of the basal ganglia, is involved in the selection and sequencing of actions. Terminals of axons projecting from the substantia nigra pars compacta release dopamine onto synapses within the striatum, suggesting they play a gating role. The activity of these dopamine is similar to that of the VTA neurons discussed earlier.

8 Point Processes

The brain manipulates information through action potentials (aka spikes). It's therefore natural to ask how information is represented in spike trains. While the shapes of action potential time courses may vary slightly, there's no evidence that action potential shape affects vesicle release. Therefore, it seems that the information encoded by a spike is represented entirely by its time of occurrence.

To study the statistics of spike trains, it's necessary to introduce some notation. We formally define a spike train $\mathcal{S}$ as the sequence of times at which a neuron spikes:

$$\mathcal{S} \coloneqq \{t_1, \ldots, t_N\}. \tag{52}$$

This can be written as a function in time using Dirac delta functions:

$$s(t) = \sum_{i=1}^{N} \delta(t - t_i), \tag{53}$$

which visually has the form of a sequence of zero-width spikes. A spike train can also be represented as cumulative counting function of the number of spikes occurring up to a time t:

$$N(t) = \int_0^{\to t} s(\xi)\, d\xi, \tag{54}$$

where the notation $\to t$ indicates that t is not included in the integral. This representation takes the form of a non-decreasing step-function. Finally, we can also represent a spike train as a vector in discrete time with bin width Δt:

$$\mathbf{s} = [s_1, \ldots, s_{T/\Delta t}]^\mathsf{T}; \qquad s_t = \int_{t-\Delta t}^{\to t} s(\xi)\, d\xi. \tag{55}$$

Thus, s_t represents the number of spikes landing in each bin. Note that due to neural refractory periods, $\Delta t \approx 1$ ms results in a binary s_t.

Neural responses to repeated stimuli are highly variable, especially in cortex. This variability likely arises in several ways:

- Noise—this may arise either through vesicle release or thermal noise in conductances.
- Internal processes—ongoing processing within the brain is likely to affect sensory responses. This might result in variability that operates on a slower time-scale than noise.

Denote the spike count on the ith repetition ("trial") by $N_i \coloneqq \int_0^T s_i(\xi)\, d\xi$. Experimental evidence shows that the variability in N_i is on the order of the mean (Figure 8.1A). If we fit a model of the relationship via

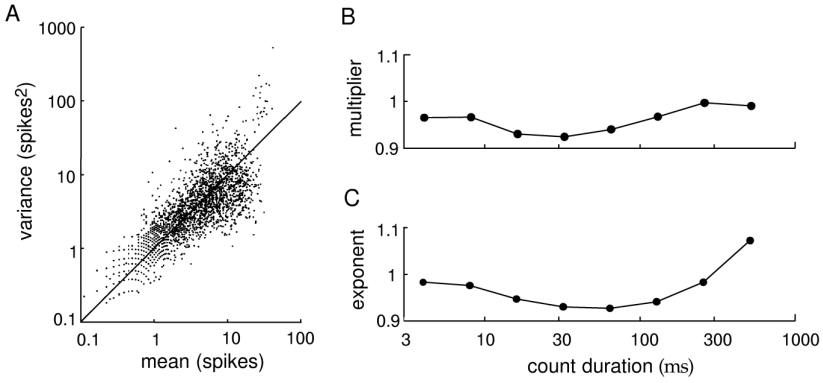


Figure 8.1: Count variability in spike trains.

$$\mathbb{V}[N_i] = A \cdot \mathbb{E}\left[N_i\right]^B, \tag{56}$$

the best-fit values turn out to be $A, B \approx$ 1-1.5 (Figure 8.1B,C). Note that when $B = 1$, A defines the *Fano factor*, the ratio of the variance to the mean. Skipping ahead a bit, a Fano factor of 1 is suggestive of a Poisson process, as the mean of a Poisson distribution equals the variance.

To describe spike trains, we define a *point process* as a probabilistic process that produces events of the type

$$\mathcal{S} = \{t_1, \ldots, t_N\} \subset \mathcal{T}, \tag{57}$$

where we define $\mathcal{T} \coloneqq [0, T]$ to be a time interval. Every point process defined on an ordered set is associated with a dual *counting process* which produces events $N(t)$ of the type

$$\begin{aligned} N(t) &\geq 0 \quad \text{(non-negativity)} \\ N(t') &\geq N(t) \text{ if } t' > t \quad \text{(non-decreasing—"staircase" shape)} \\ N(t) - N(s) &\coloneqq N[s,t) \in \mathbb{Z} \quad \text{(count within an interval)}. \end{aligned} \tag{58}$$

$N(t)$ then gives the number of events occurring with $t_i < t$.

8.1 Homogeneous Point Processes

In the simplest form of point process, events are characterized by two features: *independence* and occurrence at a fixed *rate* λ. We define these terms as follows:

1. *Independence*: For all disjoint intervals $[s, t)$ and $[s', t')$, $N_\lambda[s, t) \perp N_\lambda[s', t')$.
2. *Mean event rate*: $\mathbb{E}\left[N_\lambda[s, t)\right] = (t - s)\lambda$.

Independence implies that knowing the number (or times) of one or more events tells us nothing about the other possible events. Note that condition 2 assumes that

$$\lim_{ds \to 0} N_\lambda[s, s + ds) \in \{0, 1\} \tag{59}$$

(in other words, that as the bin gets infinitely small, there's either a spike or there isn't). This assumption is called *conditional orderliness*—at most event occurs at one time. Without assuming conditional orderliness, we could instead define the process by giving the whole distribution $N_\lambda[s, t)$. Instead, we will use the more restrictive defining assumption to derive the distribution.

We can use the two conditions above to derive the distribution. First, we divide the interval $[s, t)$ into M bins of length Δ (i.e., $M = (t - s)/\Delta$). If $\Delta \ll 1/\lambda$, conditional orderliness implies that the spike count per bin is binary (note also that the inverse rate $1/\lambda$ is *period* of the process). For a binary random variable (aka an indicator variable), the expectation is the same as the probability of the event, so the mean event rate gives

$$P(N[t, t + \Delta) = 1) = \mathbb{E}\left[N_\lambda[t, t + \Delta)\right] = (t + \Delta - t)\lambda = \lambda\Delta. \tag{60}$$

The distribution of $N[s, t)$ is binomial, and the probability of n spikes occurring in the interval is:

$$P(N_\lambda[s, t) = n) = \binom{M}{n} (\lambda\Delta)^n (1 - \lambda\Delta)^{M-n} \tag{61}$$

$$= \frac{M!}{n!(M-n)!} \left(\lambda \underbrace{\frac{t-s}{M}}_{=\Delta} \right)^n \left(1 - \lambda \frac{t-s}{M}\right)^{M-n} \tag{62}$$

and writing $\mu \coloneqq \lambda(t - s)$, we get

$$= \frac{\mu^n}{n!} \frac{\overbrace{M(M-1)\cdots(M-n+1)}^{n \text{ terms}}}{M^n} \left(1 - \frac{\mu}{M}\right)^{-n} \left(1 - \frac{\mu}{M}\right)^{M} \tag{63}$$

taking the limit $\Delta \to 0$ or, equivalently, $M \to \infty$

$$= \frac{\mu^n}{n!} 1^n 1^{-n} e^{-\mu} = e^{-\mu} \frac{\mu^n}{n!}. \tag{64}$$

Therefore, the spike count is Poisson distributed. As mentioned above, however, we could have dispensed with the conditional orderliness assumption and instead made the equivalent defining property

2. *Count distribution*: $N_\lambda[s, t) \sim \text{Poiss}[(t - s)\lambda]$.

We will now derive a number of properties of the homogeneous Poisson process.

Count Variance We first derive the variance of the count distribution (a key characteristic of the Poisson distribution):

$$\begin{aligned}
\mathbb{V}[N_\lambda[s,t)] &= \left\langle (n-\mu)^2 \right\rangle = \left\langle n^2 \right\rangle - \mu^2 \\
&= \left\langle \underbrace{n(n-1)+n}_{\text{good trick to know}} \right\rangle - \mu^2 \\
&= \underbrace{\sum_{n=0}^{\infty} n(n-1)\frac{e^{-\mu}\mu^n}{n!}}_{=\langle n(n-1)\rangle} + \underbrace{\mu}_{=\langle n\rangle} - \mu^2 \\
&= \mu^2 \underbrace{\sum_{n=0}^{\infty} \frac{e^{-\mu}\mu^{n-2}}{(n-2)!}}_{=0 \text{ for } n=0,1} + \mu - \mu^2 \\
&= 0 + 0 + \mu^2 \underbrace{\sum_{(n-2)=0}^{\infty} \frac{e^{-\mu}\mu^{n-2}}{(n-2)!}}_{=1} + \mu - \mu^2 \\
&= \mu^2 + \mu - \mu^2 = \mu.
\end{aligned} \tag{65}$$

Thus, the mean equals the variance and

3. *Fano factor*: $\frac{\mathbb{V}[N_\lambda[s,t)]}{\mathbb{E}[N_\lambda[s,t)]} = 1$.

ISI Distribution We now discuss the statistics governing the *inter-spike interval* (ISI). First, it is fairly straightforward to see that, since the counting processes before and after event t_i are independent, the times to the previous and following spikes are independent as well:

4. *ISI independence*: $\forall i > 1$, $t_i - t_{i-1} \perp t_{i+1} - t_i$.

The full ISI distribution can be derived from the count distribution:

$$\begin{aligned}
P[t_{i+1} - t_i \in [\tau, \tau + d\tau)] &= \overbrace{P[N_\lambda[t_i, t_i+\tau) = 0]}^{=P[\text{no spikes here}]} \times \overbrace{P[N_\lambda[t_i+\tau, t_i+\tau+d\tau) = 1]}^{=P[\text{next spike occurs in infinitesimal interval}]} \\
&= \frac{\mu^0 e^{-\lambda\tau}}{0!} \times \frac{\lambda d\tau e^{-\lambda d\tau}}{1!} \\
&= e^{-\lambda\tau}\lambda d\tau e^{-\lambda d\tau}
\end{aligned}$$

taking $d\tau \to 0$

$$= \lambda e^{-\lambda\tau}\, d\tau, \tag{66}$$

and therefore

5. *ISI distribution*: $\forall i \geq 1$, $t_{i+1} - t_i \sim$ iid Exponential$[\lambda^{-1}]$.

From this it follows that

6. *Mean ISI*: $\mathbb{E}[t_{i+1} - t_i] = \lambda^{-1}$
7. *Variance ISI*: $\mathbb{V}[t_{i+1} - t_i] = \lambda^{-2}$

These two properties imply that the *coefficient of variation* (CV) of the ISIs, defined as the ratio of the standard deviation to the mean, is $CV = \sqrt{\lambda^{-2}}/\lambda^{-1} = 1$.

Joint Density Finally, we consider the joint probability of observing a spike train $\{t_1, \ldots, t_N\}$ in interval $\mathcal{T}$. Spike times are independent and arrive at a uniform rate, giving

$$p(t_1, \ldots, t_N)\, dt_1 \ldots dt_N = P[N \text{ spikes in } \mathcal{T}] \times \prod_i \overbrace{P[i\text{th } \textit{spike} \in [t_i, t_i + dt_i)]}^{\forall i} \times [\#\text{ of equiv. spike orderings}], \tag{67}$$

where the first term is given by the Poisson distribution, the second by the uniform distribution of spike times conditioned on N, and the third is $N!$, giving

$$\begin{aligned} p(t_1, \ldots, t_N)\, dt_1 \ldots dt_N &= \left(\frac{(\lambda T)^N e^{-\lambda T}}{N!} \right) \left(\frac{dt_1}{T} \cdots \frac{dt_N}{T} \right) N! \\ &= \lambda^N e^{-\lambda T}\, dt_1 \ldots dt_N. \end{aligned} \tag{68}$$

We will see another way to write down the same expression while considering the inhomogeneous Poisson process below.

8.2 Inhomogeneous Point Processes

The inhomogeneous Poisson process generalizes the constant event-arrival rate λ to a time-dependent rate $\lambda(t)$, while preserving the assumption of independent spike arrival times. It's possible to quickly summarize the properties of the inhomogeneous process by reference to the homogeneous one.

To begin, the two defining properties are

1. *Independence*: For all disjoint intervals $[s, t)$ and $[s', t')$, $N_{\lambda(t)}[s, t) \perp N_{\lambda(t)}[s', t')$.
2. *Count distribution*: $N_{\lambda(t)}[s, t) \sim \text{Poiss}[\int_s^t \lambda(\xi)\, d\xi]$ (i.e., the rate is the expected number of events in the interval).

The variance in the counts is simply a consequence of the Poisson counting distribution, and so the next property follows directly:

3. *Fano factor*: $\frac{\mathbb{V}[N_{\lambda(t)}[s,t)]}{\mathbb{E}\left[N_{\lambda(t)}[s,t)\right]} = 1$.

ISI Distribution The independence of counting in disjoint intervals means that ISIs remain independent:

4. *ISI independence*: $\forall i > 1$, $t_i - t_{i-1} \perp t_{i+1} - t_i$.

The full distribution of ISIs is found in a similar manner to that of the homogeneous process distribution:

$$\begin{aligned} P[t_{i+1} - t_i \in [\tau, \tau + d\tau)] &= \overbrace{P[N_{\lambda(t)}[t_i, t_i + \tau)]}^{P(\text{no spike in interval})} \times \overbrace{P[N_{\lambda(t)}[t_i + \tau, t_i + \tau + d\tau)]}^{P(\text{spike at } t_i + \tau)} \\ &= e^{-\int_{t_i}^{t_i+\tau} \lambda(\xi)\, d\xi} \times e^{-\int_{t_i+\tau}^{t_i+\tau+d\tau} \lambda(\xi)\, d\xi} \int_{t_i+\tau}^{t_i+\tau+d\tau} \lambda(\xi)\, d\xi \end{aligned}$$

taking $d\tau \to 0$

$$\begin{aligned} &= e^{-\int_{t_i}^{t_i+\tau} \lambda(\xi)\, d\xi} \times \underbrace{e^{-\lambda(t_i+\tau)\, d\tau}}_{d\tau \to 0} \lambda(t_i + \tau)\, d\tau \\ &= e^{-\int_{t_i}^{t_i+\tau} \lambda(\xi)\, d\xi} \lambda(t_i + \tau)\, d\tau, \end{aligned} \tag{69}$$

and thus we have

5. *ISI distribution*: $\forall i \geq 1$, $P(t_{i+1} - t_i) = e^{-\int_{t_i}^{t_i+\tau} \lambda(\xi)\, d\xi} \lambda(t_{i+1})$.

Because the ISI distribution is *not* iid, it is not as useful to consider its mean or variance.

Joint Density The joint probability of the event $\{t_1, \ldots, t_N\}$ can be derived by setting the count in intervals *between* spikes to 0, and the count in an infinitesimal *around* t_i to 1. This gives

$$\begin{aligned} p(t_1, \ldots, t_N)\, dt_1 \ldots dt_N &= \underbrace{P[N[0, t_1) = 0]}_{\text{no spikes before first}} \times P[N[t_1, t_1 + dt_1) = 1] \times \cdots \times \underbrace{P[N(t_N, T) = 0]}_{\text{no spikes after last}} \\ &= e^{\int_0^{t_1} \lambda(\xi)\, d\xi} \times \lambda(t_1)\, dt_1 \times \cdots \times e^{\int_{t_N}^{T} \lambda(\xi)\, d\xi} \\ &= \underbrace{e^{-\int_0^T \lambda(\xi)\, d\xi}}_{\text{all non-spikes}} \underbrace{\prod_{i=1}^{N} \lambda(t_i)\, dt_1 \ldots dt_N}_{\text{all the spikes}}. \end{aligned} \tag{70}$$

This form can be generalized to pretty much any point process. Observe that it takes the typical form of a joint distribution—a product of terms times a normalizer. Note also that if we set $\lambda(t) = \lambda$, we recover the result for the homogeneous process.

Time Rescaling Finally, we derive an additional important property of the inhomogeneous process. Let us rewrite the density above by changing variables from t to u according to

$$u(t) \coloneqq \int_0^t \lambda(\xi)\, d\xi \qquad \text{i.e., } u_i = \int_0^{t_i} \lambda(\xi)\, d\xi \Leftrightarrow \frac{du_i}{dt_i} = \lambda(t_i) \Leftrightarrow du_i = \lambda(t_i)\, dt_i. \tag{71}$$

Then

$$\begin{aligned} p(u_1, \ldots, u_n) &= \frac{p(t_1, \ldots, t_n)}{\prod_i \frac{du_i}{dt_i}} \\ &= e^{-u(T)} \frac{\prod_{i=1}^N \lambda(t_i)}{\prod_{i=1}^N \lambda(t_i)} \\ &= e^{-u(T)}. \end{aligned} \tag{72}$$

Comparing this to the density for a homogeneous Poisson process shows that the variables u_i are distributed according to a homogeneous Poisson process with mean rate $\lambda = 1$. Thus, $u(t)$ effectively *rescales time* to transform the inhomogeneous process into a homogeneous process. This process is called time rescaling, and is central to the study of point processes in time.

8.3 Self-Exciting and Renewal Processes

A *self-exciting process* has an intensity function that is conditioned on past events:

$$\lambda(t) \to \lambda(t|N(t), t_1, \ldots, t_{N(t)}). \tag{73}$$

We can then define the notation $H(t)$ to represent the event *history* at time t—representing both $N(t)$ and the times of the corresponding events. Then the self-exciting intensity function can be written $\lambda(t|H(t))$. This is actually the most general form of a point process—we can re-express any (conditionally orderly) point process in this form. To see this, consider the point process to be the limit as $\Delta \to 0$ of a binary time series $\{b_1, b_2, \ldots, b_{T/\Delta}\}$ and note that

$$P(b_1, b_2, \ldots, b_{T/\Delta}) = \prod_i P(b_i|b_{i-1}, \ldots, b_1) \propto \prod_i \lambda(b_i|b_1, \ldots, b_{i-1})\Delta, \tag{74}$$

and taking the limit $\Delta \to 0$ gives

$$= f(\lambda(t|H(t))), \tag{75}$$

for some function $f(\cdot)$.

Renewal Processes If the intensity of a self-exciting process depends only on the time since the last spike, i.e.,

$$\lambda(t|H(t)) = \lambda(t - t_{N(t)}), \tag{76}$$

then the process is called a *renewal process*. ISIs from a renewal process are iid and so we can could equivalently have defined the process by its ISI density. This gives an (almost) easy way to write the probability of having observed $\{t_1, \ldots, t_N\}$ in T. Suppose, for simplicity, that there was an event at $t_0 = 0$. Then if the ISIs are distributed according to a probability density $p(\tau)$:

$$p(t_1, \ldots, t_N)\, dt_1 \ldots dt_N = \prod_{i=1}^N \underbrace{p(t_i - t_{i-1})}_{\text{prob. of interval}} \underbrace{\left(1 - \int_0^{T - t_N} p(\tau)\, d\tau\right)}_{\text{prob. of no events from } t_N \text{ to } T}, \tag{77}$$

where the last term gives the probability that no more spikes are observed after t_N. Note that if we had not assumed that there was a spike at time $t = 0$, we would have needed a similar term at the front. The conditional intensity—sometimes called the *hazard function*—for the renewal process defined by ISI density $p(\tau)$ is

$$\lambda(t|t_{N(t)})\, dt = \frac{\overbrace{p(t - t_{N(t)})}^{\text{prob. of event at } t \text{ given prev. event at } t_{N(t)}}}{\underbrace{1 - \int_0^{t - t_{N(t)}} p(\tau)\, d\tau}_{\text{prob. of no events in interval of len. } t - t_{N(t)}}} dt, \tag{78}$$

which is indeed a function only of $t - t_{N(t)}$. Note that the general form for the ISI distribution for a renewal process is given by

$$p(\tau) = \tau e^{-\int_0^\tau \tau' \, d\tau'}, \tag{79}$$

an exponential distribution—similar to that of the inhomogeneous Poisson process.

Gamma-Interval Process A renewal process with ISI density given by

$$t_{i+1} - t_i \sim^{\text{iid}} \text{Gamma}[\alpha, \beta], \tag{80}$$

where

$$\tau \sim \text{Gamma}[\alpha, \beta] \Rightarrow p(\tau) = \frac{\beta^\alpha}{\Gamma(\alpha)} \tau^{\alpha-1} e^{-\beta\tau} \tag{81}$$

is a *gamma-interval process*. This is an important renewal process in theoretical neuroscience, because the ISI distribution has a refractory-like component (see Figure 8.2). A homogeneous Poisson process is a gamma-

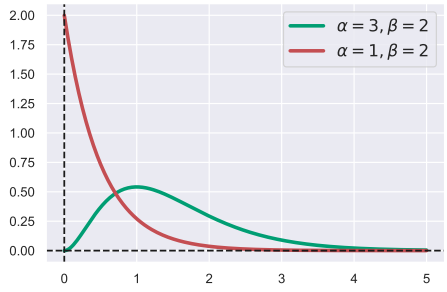


Figure 8.2: Example ISI distributions for gamma-interval processes. The polynomial build-up to the peak for $\alpha > 1$ (given by $\tau^{\alpha-1}$) can be interpreted as a refractory period. Setting $\alpha = 1$ recovers an exponential distribution, the ISI distribution for a homogeneous process.

interval process (and therefore a renewal process) with $\alpha = 1$ (as setting $\alpha = 1$ converts a gamma distribution to an exponential distribution). The parameter α is sometimes called the "order" or the "shape" parameter of the gamma-interval process. Larger values of α shape the polynomial rising part of the gamma density, thus implementing a relative refractory period. The long-time behavior is dominated by the exponential decay with coefficient β.

Interestingly, it's possible to construct a gamma-interval process of integral order α by drawing every αth event from a homogeneous Poisson process (this is trivially true for $\alpha = 1$). Consider the case that $\alpha = 2$ (depicted in Figure 8.3). The ISI probabilities for a process that accepts times t_1 and t_2 and rejects a spike at time t is just the product of the ISI probabilities of the homogeneous process from t_1 to t and from t to t_2, as disjoint intervals are independent. However, we must also integrate (average) over all possible settings of t. Letting $\tau \coloneqq t_2 - t_1$, we have

$$P(\tau) = \int_{t_1}^{t_2} \overbrace{P(t - t_1)P(t_2 - t)}^{\text{homog. process ISI probs.}} \, dt \tag{82}$$

$$= \int_{t_1}^{t_2} \lambda e^{-\lambda(t-t_1)} \lambda e^{-\lambda(t_2-t)} \, dt \tag{83}$$

$$= \lambda^2 \int_{t_1}^{t_2} e^{-\lambda t} e^{\lambda t_1} e^{-\lambda t_2} e^{\lambda t} \, dt \tag{84}$$

$$= \lambda^2 [t_2 - t_1] e^{-\lambda(t_2-t_1)} \tag{85}$$

$$= \lambda^2 \tau e^{-\lambda\tau} = \text{Gamma}[2, \lambda]. \tag{86}$$

This can be easily generalized to higher values of α. Thus, taking every αth spike results in a gamma process of order α and rate parameter $\beta = \lambda$.

Inhomogeneous Renewal Processes In an *inhomogeneous* renewal process, the rate depends both on the time since the last spike and on the current time:

$$\lambda(t) \to \lambda(t, t - t_{N(t)}). \tag{87}$$

This is also called an "inhomogeneous Markov interval" process. There are two popular ways to construct an inhomogeneous renewal process:

1. *Time Rescaling*: Given unit-mean ISI density $p(\tau)$ (*since you're rescaling time, so might as well choose to have unit rate*) and time-varying intensity $\rho(t)$, define

$$p(t_1, \ldots, t_N)\, dt_1 \ldots dt_N \coloneqq \prod_{i=1}^{N} p\left(\int_{t_{i-1}}^{t_i} \rho(\xi)\, d\xi\right)\left(1 - \int_0^{\int_{t_N}^{T} \rho(\xi)\, d\xi} p(\tau)\, d\tau\right) \tag{88}$$

2. *Spike-Response*:

$$\lambda(t, t - t_{N(t)}) \coloneqq f(\rho(t), h(t - t_{N(t)})) \tag{89}$$

for a simple f. Often, f just multiplies the two functions (or, equivalently, adds log-intensities). The term "spike-response" comes from the use of such spike-triggered currents to create a potentially more tractable approximation to an integrate-and-fire neuron.

These definitions differ in how ISI density depends on ρ. In *rescaling*, higher rates make time pass faster, so ISI interactions are rescaled. *Spike-response*: a refractory h may not suppress spikes as well at higher rates, but the duration of influence does not change.

8.4 General Spike-Response Processes

This category of processes has come to be used with increasing frequency recently, particularly in a generalized linear form. The product form of spike-response renewal processes can be written as

$$\lambda(t, t - t_{N(t)}) = \exp(\rho(t) + h(t - t_{N(t)})) \tag{90}$$

and then generalized to include influence from all (or > 1) past spikes:

$$\lambda(t|H(t)) = \exp\left(\rho(t) + \sum_j h(t - t_{N(t)-j})\right). \tag{91}$$

Often, we want to estimate the parameters of a point-process model from spike data. Assuming a generalized linear form makes this easier. To do this, we can write the history influence h in terms of a linear combination of basis functions $h_i(\tau)$:

$$\lambda(t|H(t)) = \exp\left(\rho(t) + \sum_{ij} \alpha_i h_i(t - t_{N(t)-j})\right). \tag{92}$$

Note that this essentially defines an exponential family form for the intensity function—learning the weights α_i corresponds to learning the parameters of the sufficient statistics with encoding functions h_i, which define the biophysical properties of the neuron and shouldn't change. If $\rho(t)$ is also written as a linear function of external covariates, then the complete model can be fit by the standard methods used for generalized linear models (GLMs).

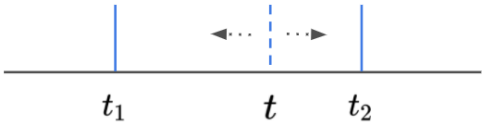


Figure 8.3: Constructing a gamma process by drawing every other spike in a homogeneous Poisson process. The times t_1 and t_2 are spike times that are accepted by the constructed process. The "skipped" spike time time is t.

The Doubly-Stochastic Poisson (or Cox) Process In the *doubly-stochastic* or *Cox* process, $\lambda(t)$ itself is either a random variable or depends on another random process. One example is the randomly scaled IHPP:

$$\lambda(t) \coloneqq s \cdot \rho(t), \tag{93}$$

with $\rho(t)$ fixed and $s \sim \text{Gamma}(\alpha, \beta)$. These models are useful for modeling a stimulus-dependent response $\rho(t)$ which is modulated by cortical excitability. The counting process for such a DSPP has a relatively high variance, with Fano factor greater than 1. DSPP models also provide a useful way to introduce dependencies between two or more point processes, through correlations in their intensity functions, and are common inputs

to log-linear spike response models. One important example of a DSPP is *Gaussian process factor analysis* (GPFA). The intensity functions for GPFA are written as

$$\lambda^i(t) = \exp\left(\sum_k C_{ik}\rho^k(t)\right), \qquad \text{with } C_{ik} \sim \mathcal{GP}(0, K), \tag{94}$$

where K is some covariance function.

Joint Models Some examples of joint models are 2D point processes (not considered useful, as they don't really match the modelling intuition. We really want to think about factors correlated in time - though not impossible to use), superimposed processes, and infinitely divisible Poisson processes. It's also useful to consider *multivariate processes*, in which the intensity function for the ith neuron can be written in the general form

$$\lambda^i(t) = f(t, H^i(t), H^j(t), \dots). \tag{95}$$

If $f(\cdot,\cdot)$ is linear, the process is called a *Hawkes process*, and can be written as

$$\lambda^i_H(t) = g^i(t) + \sum_{j,n_j} \underbrace{h_{ij}(t - t^j_{n_j})}_{\text{basis fns}}, \tag{96}$$

where j indexes neighboring neurons and n_j indexes the spike times of the neighboring neurons. When the dependence is log-linear, it's called a generalized Hawkes process—or a GLM, i.e.,

$$\lambda^i_{\text{GLM}}(t) \coloneqq \exp(\lambda^i_H(t)). \tag{97}$$

8.5 Measuring Point Processes

Given data, we can construct generative models for point processes, but it's also important to consider the techniques used to measure and analyze them. Consider a data set in which spike trains from a single neuron are obtained from repeated experiments under constant experimental conditions:

$$s^{(k)}(t) = \sum_{i=1}^{N^{(k)}} \delta(t - t_i^{(k)}) \qquad \text{for trials } k = 1, \dots, K. \tag{98}$$

Visually, we can think of this as multiple sequences of delta-spikes occurring at different times, indexed by k. How can we characterize $s^{(k)}(t)$ and its relationship to the stimulus (or task)?

One family of options are parametric point-process models, possibly dependent on a stimulus $a(t)$:

$$s^{(k)}(t) \sim \lambda\left(t, a[0,t), N^{(k)}(t), t_1^{(k)}, \dots, t^{(k)}_{N^{(k)}(t)}, \theta\right). \tag{99}$$

In other words, this model attempts to predict the activity from the input stimulus. Such a framework constitutes an *encoding* model. Encoding models are discussed in depth in Section 10. The inverse approach—termed *decoding*—is to construct an algorithm that estimates $a(t)$ from $s^{(k)}(t)$:

$$\hat{a}(t) = F_\theta[s^{(k)}[0,t), \theta]. \tag{100}$$

One non-parametric option is to estimate statistics (usually *moments*) of the distribution of $s^{(k)}(t)$.

Another problem is simultaneously modelling responses from multiple cells. If no two processes can generate events at precisely the same time (a form of conditional orderliness), or if simultaneous spiking events are independent, then dependencies between the processes arise through dependence on all previous events in all cells:

$$\lambda^{(c)}(t) \coloneqq \lambda^{(c)}\left(t | N^{(c)}(t), t_1^{(c)}, \dots, t_1^{(c)}, \{N^{(c')}(t), t_1^{(c')}, \dots, t_1^{(c')}\}, \theta\right) \tag{101}$$

where c' indexes all other cells. This is analogous to the self-exciting point process intensity function. Dependencies can also be expressed in other forms, for example by DSPPs with the latent random process shared (or correlated) between cells. Such representations may often be more natural or causally accurate.

8.5.1 Mean Intensity and the PSTH

We now return to the case of multiple trials from a single neuron. The simplest non-parametric characterization of a spike process is with the *mean intensity*:

$$\bar{\lambda}(t) \coloneqq \langle s(t) \rangle = \lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} s^{(k)}(t). \tag{102}$$

Note that this is *not* the same as the intensity function for the point process marginalized over history (unless it's Poisson, in which case, the intensity is already history-independent):

$$\bar{\lambda}(t, a(\cdot)) \coloneqq \int \int p(t_1, \ldots, t_{N(t)}) \lambda(t, a(\cdot), N(t), t_1, \ldots, t_{N(t)}) \, dt_1 \ldots dt_{N(t)} \, dN(t). \tag{103}$$

For finite K, estimating $\bar{\lambda}$ by summing δ-functions yields spiky results. Instead, we can bin using a histogram:

$$\widehat{\bar{N}[t, t+\Delta t)} = \frac{1}{K} \underbrace{\sum_{k=1}^{K} N^{(k)}[t, t+\Delta t)}_{\text{sum spikes w/in bin across trials}}. \tag{104}$$

This is called the *peri-(post-)stimulus time histogram* (PSTH). If we'd like $\bar{\lambda}(t)$ to be smooth, we can use a kernel $\phi(\tau)$:

$$\widehat{\bar{\lambda}(t)} = \frac{1}{K} \sum_{k=1}^{K} \int \phi(\tau) s^{(k)}(t-\tau) \, d\tau. \tag{105}$$

Note the similarity to kernel density estimation (without normalization). The width of ϕ can be chosen adaptively, depending on the local density of spikes. In general, sampling from a smoothed function makes more sense than smoothing a binned histogram. Alternatively, one can also impose a smooth prior prior (e.g., a GP) on a time-varying aspect of the intensity: $\lambda(t)$ for an inhomogeneous Poisson process, or, e.g., $\rho(t)$ for an inhomogeneous gamma-interval of order γ:

$$\begin{aligned} \boldsymbol{\rho} &\sim \mathcal{N}(\mu \mathbf{1}, K_\theta) \\ p(t_1, \ldots, t_N | \boldsymbol{\rho}) &= \prod_{i=1}^{N} \left[\underbrace{\frac{\gamma^\gamma}{\Gamma(\gamma)} \left(\sum_{j=t_{i-1}}^{t_i - 1} \rho_j \Delta \right)^{\gamma - 1}}_{\text{rescaled time under intensity fn}} \exp\left(-\gamma \sum_{j=t_{i-1}}^{t_i - 1} \rho_j \Delta \right) \right] \end{aligned} \tag{106}$$

The posterior on $\rho(t)$ can then be found via approximate inference (e.g., Laplace, EP, etc.).

8.5.2 Autocorrelation and Auotocovariance

The *autocorrelation function* for a process that generates spike trains $s(t)$ is

$$R_{ss}(\tau) = \left\langle \frac{1}{T} \int s(t) s(t-\tau) \, dt \right\rangle, \tag{107}$$

where $\langle \cdot \rangle$ denotes an expectation with respect to random draws of $s(t)$ from the process. This is the *time-averaged* local second moment of the joint on $s(t)$ (note that $\bar{\lambda}(t)$ is the *non*-time-averaged first moment). Also note that, since $s(t)$ is a sum of δ functions, $R_{ss}(0) = \int \delta^2 \, dt = \infty$ under this definition.

Alternatively, we could define R_{ss} as the time-averaged conditional first moment. In other words, the mean intensity at $t + \tau$, conditioned on an event at time t, averaged over t:

$$R_{ss}^{alt}(\tau) = \frac{1}{T} \int \langle \lambda(t+\tau | \exists i : t_i = t) \rangle \, dt, \tag{108}$$

where $\langle \cdot \rangle$ here denotes an expectation with respect to $N(T)$ and $t_{j \neq i}$. In this case, $R_{ss}^{alt}(0) = 0$. For the rest of the discussion, we'll stick to the first (second-moment) definition.

Using the identity $\left\langle x^2 \right\rangle = \mathbb{V}[x] + \mu^2 = \left\langle (x-\mu)^2 \right\rangle + \mu^2$, we can decompose the autocorrelation function as follows:

$$R_{ss}(\tau) = \bar{\Lambda}^2 + \frac{1}{T} \int (\bar{\lambda}(t) - \bar{\Lambda})(\bar{\lambda}(t-\tau) - \bar{\Lambda}) \, dt + \underbrace{\left\langle \frac{1}{T} \int (s(t) - \bar{\lambda}(t))(s(t-\tau) - \bar{\lambda}(t-\tau)) \, dt \right\rangle}_{\coloneqq Q_{ss}(\tau)}, \tag{109}$$

where $\bar{\Lambda}$ is the time-averaged mean rate. $Q_{ss}(\tau)$ is called the *autocovariance* function. This has several notable properties.

- For an (inhomogeneous) Poisson process $Q_{ss}(\tau) = \delta(\tau)$ by independence.
- For a general self-exciting process, $Q_{ss}(\tau)$ gives (to second order) dependence on nearby spike times.

- The autocovariance function is often used to look for oscillatory structure in spike trains (where spikes tend to repeat around fixed intervals, but at random phase with respect to the stimulus) or similar spike-timing relationships.
- But, as any point process is self-exciting, *any* non-Poisson process will have non-δ autocovariance, even if nearby spike-timing relationships are not the most natural (or causal) way to describe the generative process. For example, consider the effects of random—but slow— variations in a non-constant $\lambda(t)$, as in a DSPP.

Estimating Correlation Functions Correlation functions are typically estimated by constructing *correlograms*: histories of time *differences* between (not necessarily adjacent) spikes. The covariance function is estimated by subtracting an estimate of the correlation of the mean intensity:

$$\begin{aligned}\frac{1}{T}\int \hat{\bar{\lambda}}(t)\hat{\bar{\lambda}}(t-\tau)\,dt &= \frac{1}{TK^2}\int \sum_k s^{(k)}(t)\sum_{k'} s^{(k')}(t-\tau) \\ &= \frac{1}{TK^2}\sum_{kk'}\int s^{(k)}(t)s^{(k')}(t-\tau)\,dt.\end{aligned} \tag{110}$$

This is called the *shift* or *shuttle* correction. An estimate may also be constructed in the frequency domain, e.g., through a power spectrum, spectrogram, or coherence (for multiple processes). This is usually based on a Fourier transform of binary-binned spike trains.

Cross-Correlations In the case that we are also measuring the relationships among multiple cells, the techniques are analogous to those for single processes. The *cross-correlogram* estimate of the *cross-correlation* function is

$$R_{s^{(c)}s^{(c')}}(\tau) = \left\langle \frac{1}{T}\int s^{(c)}(t)s^{(c')}(t-\tau)\,dt \right\rangle, \tag{111}$$

where c indexes the cells. Accordingly, the shift- or shuttle-corrected correlogram estimate of the cross-variance function is then

$$Q_{s^{(c)}s^{(c')}}(\tau) = \left\langle \frac{1}{T}\int\int (s^{(c)}(t)-\bar{\lambda}^{(c)}(t))(s^{(c')}(t-\tau)-\bar{\lambda}^{(c')}(t-\tau))\,dt \right\rangle. \tag{112}$$

As for autocovariograms, structure in a cross-covariogram need not imply that dependencies between individual spike times are the most natural way to think about the interaction between the processes—DSPPs with shared latents may also give significant cross-covariance structure.

8.6 Point Process Tips

- Always keep in mind that, in general, the mean rate $\bar{\lambda}$ corresponds to the inverse of the mean inter-spike interval. Therefore, to find a mean firing rate, derive $p(\tau)$, find $\bar{\tau} = \mathbb{E}_{p(\tau)}[\tau]$, and then $\bar{\lambda} = 1/\bar{\tau}$.

9 Information Theory

9.1 Quantifying Uncertainty

We'd like to quantify the amount of information carried by neural responses about the stimuli that induce them. More formally, we define *information* as the removal of uncertainty about a variable quantity. Consider the sequence of events defined by

$$S \to R \to P(S|R). \tag{113}$$

We'd like to know how informative R is about S. For example, S could be identifiable with complete certainty from the value of R, or R could carry no information about S, i.e.,

$$P(S|R) = [0,\, 0,\, 1,\, 0, \ldots,\, 0]^\mathsf{T} \qquad \text{(complete information)} \tag{114}$$

$$P(S|R) = \left[\frac{1}{M}, \frac{1}{M}, \ldots, \frac{1}{M}\right]^\mathsf{T} \qquad \text{(no information)}. \tag{115}$$

These probabilities also depend on $P(S)$, however. To start our analysis, we'll consider the uncertainty inherent in a probability distribution, termed the *entropy*. Let $S \sim P(S)$. The entropy is the minimum number of bits needed, on average, to specify the value that S takes, assuming $P(S)$ is known. Equivalently, this is the minimum average number of yes/no questions needed to guess S. Note that for the most part we'll be working with discrete distributions here, as formally extending most information theoretic principles to continuous distributions is non-trivial.

9.1.1 Entropy and Conditional Entropy

Suppose there are M equiprobable stimuli: $P(s_m) = p = 1/M$. To specify which stimulus appears on a given trial, we would need to assign each a binary number. The number of bits B_s required is

$$\begin{aligned} 2^{B_s} \geq M \Rightarrow B_s &\leq \log_2 M \\ &= -\log_2 \frac{1}{M}. \end{aligned} \tag{116}$$

Now suppose we code N such stimuli, drawn iid, at once. We get

$$\begin{aligned} B_N &\leq \log_2 M^N \\ &\to -N \log_2 \underbrace{\frac{1}{M}}_{=p} = -\sum_s \log_2 p \quad \text{as } N \to \infty \\ \Rightarrow B_s &\to -\log_2 p \text{ bits}. \end{aligned} \tag{117}$$

This is called block coding. It is useful for extracting theoretical limits. The nervous system is unlikely to use block codes, though Maneesh says it may in space.

Now suppose the stimuli are not equiprobable. Write $P(s_m) = p_m$. Then

$$P(S_1, S_2, \ldots, S_N) = \prod_m p_m^{n_m}, \tag{118}$$

where n_m is the number of $S_i = s_m$. As $N \to \infty$ only "typical" sequences, with $n_m = p_m N$, have non-zero probability of occurring, and they are all equally likely. This is called the Asymptotic Equipartition Property (AEP). Thus, eq. 117 gives

$$\begin{aligned} B_N &\to -\log_2 \prod_m p_m^{n_m} = -\sum_m n_m \log_2 p_m \\ &= -\sum_m p_m N \log_2 p_m = -N \underbrace{\sum_m p_m \log_2 p_m}_{-\mathbf{H}[s]}. \end{aligned} \tag{119}$$

Then $\mathbf{H}[S] = \mathbb{E}\left[-\log_2 P(S)\right]$, also written $\mathbf{H}[P(S)]$ is the *entropy* of the stimulus distribution. Note that we could also derive the expression for entropy via the law of large numbers. From eq. 117, we have

$$-\frac{1}{N} \log P(S_1, S_2, \ldots, S_N) = -\frac{1}{N} \log_2 \prod_i P(S_i) = -\frac{1}{N} \sum_i \log_2 P(S_i) \to^{N\to\infty} \mathbb{E}\left[-\log_2 P(S_i)\right]. \tag{120}$$

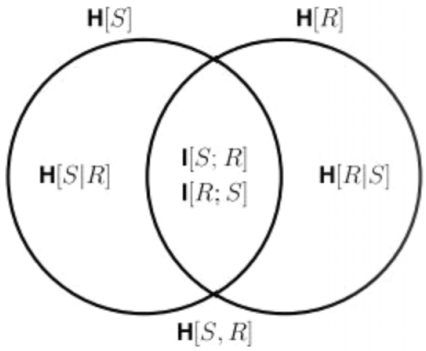


Figure 9.1: Mutual information diagram.

Entropy is a measure of the "available information" in the stimulus ensemble. Now suppose we measure a particular response r which depends on the stimulus according to $P(R|S)$. How uncertain is the stimulus once we know r? Bayes rule gives us

$$P(S|r) = \frac{P(r|S)P(S)}{\sum_s P(r|s)P(s)}, \tag{121}$$

so we can write

$$\mathbf{H}[S|r] = -\sum_s P(s|r) \log_2 P(s|r). \tag{122}$$

And so the *average* uncertainty in S for $r \sim P(R) = \sum_s P(R|s)P(s)$ is then

$$\begin{aligned} \mathbf{H}[S|R] &= \sum_r P(r)\mathbf{H}[S|r] = \sum_r P(r) \left[-\sum_s P(s|r) \log_2 P(s|r) \right] \\ &= -\sum_{s,r} P(s,r) \log_2 P(s|r). \end{aligned} \tag{123}$$

This is called *conditional entropy* of S on R. It is easy to see that:

1. $\mathbf{H}[S|R] \leq \mathbf{H}[S]$ (conditioning cannot increase uncertainty)
2. $\mathbf{H}[S|R] = \mathbf{H}[S, R] - \mathbf{H}[R]$
3. $\mathbf{H}[S|R] = \mathbf{H}[S] \Leftrightarrow S \perp R$.

9.1.2 Mutual Information

A natural definition of the average information gained about S from R is

$$\mathbf{I}[S; R] \coloneqq \mathbf{H}[S] - \mathbf{H}[S|R]. \tag{124}$$

This is the average *mutual information* between S and R. It measures the reduction in uncertainty about S given R. It follows from the definition that

$$\begin{aligned} \mathbf{I}[S; R] &= \sum_s P(s) \log \frac{1}{P(s)} - \sum_{s,r} P(s,r) \log \frac{1}{P(s|r)} \\ &= \sum_{s,r} P(s,r) \log \frac{1}{P(s)} + \sum_{s,r} P(s,r) \log P(s|r) \\ &= \sum_{s,r} P(s,r) \log \frac{P(s|r)}{P(s)} \\ &= \boxed{\sum_{s,r} P(s,r) \log \frac{P(s,r)}{P(s)P(r)}} \\ &= \mathbf{I}[R; S]. \end{aligned} \tag{125}$$

This symmetry suggests a Venn-like diagram (Figure 9.1). All of the implied additive and equality relationships hold for two variables, but this representation doesn't hold for more than two. (If you add a third variable, the region of overlap could have negative area—more later).

Kullback-Leibler (KL) Divergence The *KL divergence* is a useful information theoretic quantity that measures the difference between two distributions:

$$\mathsf{KL}[P(S)\|Q(S)] = \sum_s P(s)\log\frac{P(s)}{Q(s)} \tag{126}$$

$$= \underbrace{\sum_s P(s)\log\frac{1}{Q(s)}}_{\text{cross entropy}} - \mathbf{H}[P] \tag{127}$$

The cross entropy can be thought of as being the code length drawn from the wrong distribution. The KL divergence can be thought of as the excess cost (in bits or nats) of building the code according to Q when the true distribution is P. To see that the KL is always non-negative, we can write

$$\begin{aligned} -\mathsf{KL}[P\|Q] &= \sum_s P(s)\log\frac{Q(s)}{P(s)} \\ &\leq \log\sum_s P(s)\frac{Q(s)}{P(s)} \qquad \text{(by Jensen)} \\ &= \log\sum_s Q(s) = \log 1 = 0. \end{aligned} \tag{128}$$

Therefore $\mathsf{KL}[P\|Q] \geq 0$, with equality if and only if $P = Q$. [10]

Importantly, the mutual information can be expressed as a KL divergence as follows:

$$\mathbf{I}[S;R] = \sum_{s,r} P(s,r)\log\frac{P(s,r)}{P(s)P(r)} = \mathsf{KL}[P(S,R)\|P(S)P(R)]. \tag{129}$$

Therefore, we can say that (1) the mutual information is always non-negative, $\mathbf{I}[S;R] \geq 0$, and (2) conditioning never increases entropy:

$$\mathbf{I}[S;R] \coloneqq \mathbf{H}[S] - \mathbf{H}[S|R] \geq 0 \Rightarrow \mathbf{H}[S|R] \leq \mathbf{H}[S]. \tag{130}$$

9.2 Properties of Mutual Information and Entropy

9.2.1 Multiple Responses

Two responses to the same stimulus, R_1 and R_2, may provide either more or less information jointly than independently:

$$I_{12} \coloneqq \mathbf{I}[S;R_1;R_2] = \mathbf{H}[R_1,R_2] - \mathbf{H}[R_1,R_2|S]. \tag{131}$$

We then have

$$R_1 \perp R_2 \Rightarrow \mathbf{H}[R_1,R_2] = \mathbf{H}[R_1] + \mathbf{H}[R_2] \tag{132}$$

$$R_1 \perp R_2|S \Rightarrow \mathbf{H}[R_1,R_2|S] = \mathbf{H}[R_1|S] + \mathbf{H}[R_2|S], \tag{133}$$

from which the following relationships can be deduced: Redundancy occurs when the information provided

$R_1 \perp R_2$	$R_1 \perp R_2\|S$		
no	yes	$I_{12} < I_1 + I_2$	redundant
yes	yes	$I_{12} = I_1 + I_2$	independent
yes	no	$I_{12} > I_1 + I_2$	synergistic
no	no	?	any of the above

by R_1 and R_2 about S has an overlap, and so the combined mutual information is less than the sum of the individual mutual informations. Synergy is a case of explaining away in the corresponding graphical model. We also note that $I_{12} > \max\{I_1, I_2\}$, which implies that the second response cannot destroy information.

[10] The later statement is true because Jensen's inequality for concave f:

$$\sum_s P(s) f(\frac{Q(s)}{P(s)}) \leq f(\sum_s P(s)\frac{Q(s)}{P(s)})$$

is only an equality if the term inside the function is constant; i.e. $Q(s) = \alpha P(s)$. Since the case $\mathsf{KL}[P||Q] = 0$ is a case of Jensen equality, and since both distributions have to be normalised, this is only true if $Q(s) = P(s)$.

9.2.2 The Data Processing Inequality

Suppose $S \to R_1 \to R_2$ form a Markov chain; that is, $R_2 \perp S|R_1$. Then

$$\begin{aligned} P(R_2, S|R_1) &= P(R_2|R_1)P(S|R_1) \\ \Rightarrow \frac{P(R_2, S|R_1)}{P(R_2|R_1)} &= P(S|R_1) \\ \Rightarrow P(S|R_1, R_2) &= P(S|R_1). \end{aligned} \tag{134}$$

Thus,

$$\begin{aligned} \mathbf{H}[S|R_2] &\geq \mathbf{H}[S|R_1, R_2] = \mathbf{H}[S|R_1] \\ \Rightarrow \mathbf{I}[S; R_2] &\leq \mathbf{I}[S; R_1] \qquad (\mathbf{I}[X; Y] \coloneqq \mathbf{H}[X] - \mathbf{H}[X|Y]) \end{aligned} \tag{135}$$

This is the *data processing inequality*. It implies that any computation based on R_1 that does not have separate access to S cannot add information (in the Shannon sense) about the world. Equality holds if and only if $S \to R_2 \to R_1$ as well. In this case, R_2 is called a *sufficient statistic* for S (if that is the causal relationship; otherwise R_1 is the sufficient statistic). This is related to D-separation in graphical modeling.

9.2.3 Entropy Rate

So far we have discussed S and R as single (or iid) random variables. But real stimuli and responses form a time series. Let $\mathcal{S} = \{S_1, S_2, \dots\}$ form a stochastic process. We have

$$\begin{aligned} \mathbf{H}[S_1, S_2, \dots, S_n] &= \mathbf{H}[S_n|S_1, S_2, \dots, S_{n-1}] + \mathbf{H}[S_1, S_2, \dots, S_{n-1}] \\ &= \underbrace{\mathbf{H}[S_n|S_1, S_2, \dots, S_{n-1}] + \mathbf{H}[S_{n-1}|S_1, S_2, \dots, S_{n-2}] + \cdots + \mathbf{H}[S_1]}_{n \text{ terms}}. \end{aligned} \tag{136}$$

(Note that products of probabilities translate to sums of entropies, as the entropies exist in log space.) The *entropy rate* of $\mathcal{S}$ is defined as

$$\mathbf{H}[\mathcal{S}] \coloneqq \lim_{n\to\infty} \frac{1}{n}\mathbf{H}[S_1, S_2, \dots, S_n], \tag{137}$$

or alternatively as

$$\mathbf{H}[\mathcal{S}] \coloneqq \lim_{n\to\infty} \mathbf{H}[S_n|S_1, S_2, \dots, S_{n-1}]. \tag{138}$$

If $S_i \sim^{iid} P(s)$, then $\mathbf{H}[\mathcal{S}] = \mathbf{H}[S]$ (i.e., $\mathbf{H}[S_i|S_j] = \mathbf{H}[S_i]$). If $\mathcal{S}$ is Markov (and stationary) then $\mathbf{H}[\mathcal{S}] = \mathbf{H}[S_n|S_{n-1}]$.

9.2.4 Continuous Random Variables

The discussion so far has involved discrete S and R. Now let $S \in \mathbb{R}$ with density $p(s)$, and suppose we discretize with length Δs. Then the entropy is

$$\begin{aligned} \mathsf{H}_\Delta[S] &= -\sum_i p(s_i)\Delta s \log[p(s_i)\Delta s] \\ &= -\sum_i p(s_i)\Delta s(\log p(s_i) + \log \Delta s) \\ &= -\sum_i p(s_i)\Delta s \log p(s_i) - \log[\Delta s] \underbrace{\sum_i p(s_i)\Delta s}_{\to 1 \text{ as } \Delta s \to 0} \\ &= -\sum_i p(s_i)\Delta s \log p(s_i) - \log \Delta s \\ &\to -\int p(s) \log p(s)\, ds + \infty \qquad \text{as } \Delta s \to 0. \end{aligned} \tag{139}$$

The issue, fundamentally, is that with a finite number of bits, you can't exactly encode any $S \in \mathbb{R}$. We define the *differential entropy* by ignoring the lingering infinity:

$$h(S) \coloneqq -\int p(s) \log p(s)\, ds. \tag{140}$$

Note that unlike the discrete entropy, $h(S)$ can be negative, as well as $\pm\infty$.

Other information theoretic quantities can be defined similarly in the continuous case. The *conditional* differential entropy is

$$h(S|R) \coloneqq -\int p(s,r)\log p(s|r)\,ds\,dr, \tag{141}$$

and, like the differential entropy itself, may be poorly behaved. Interestingly, however, the mutual information is well-defined, as

$$\begin{aligned}\mathsf{I}_\Delta[S;R] &= \mathsf{H}_\Delta[S] - \mathsf{H}_\Delta[S|R] \\ &= -\sum_i \Delta s p(s_i)\log p(s_i) - \log\Delta s - \int p(r)\left(-\sum_i \Delta s p(s_i|r)\log p(s_i|r) - \log\Delta s\right)dr \\ &\to h(S) - h(S|R) \qquad \text{as } \Delta s \to 0,\end{aligned} \tag{142}$$

where the good behavior is due to the cancellation of the $\log\Delta s$s in the second line. To be even more well-behaved in a formal sense, we could instead define the differential mutual information as a KL divergence between the distribution in question and a uniform distribution—all KL divergences are well-behaved in the continuous case.

9.2.5 Maximum Entropy Distributions

We have that $\mathbf{H}[R_1,R_2] \le \mathbf{H}[R_1]+\mathbf{H}[R_2]$, with equality if and only if $R_1 \perp R_2$. If we let $\mathbb{E}_p[f(s)] = \int p(s)f(s)\,ds$ for some function f, we'd like to know which distribution $p(s)$ has the maximum entropy possible given the constraint on the expectation. To find this out, we can use Lagrange multipliers and variational derivatives:

$$\mathcal{L} = \int p(s)\log p(s)\,ds - \lambda_0\left[\int p(s)\,ds - 1\right] - \lambda_1\left[\int p(s)f(s)\,ds - a\right] \tag{143}$$

$$\frac{\delta\mathcal{L}}{\delta p(s)} = 1 + \log p(s) - \lambda_0 - \lambda_1 f(s) = 0 \tag{144}$$

$$\Rightarrow \log p(s) = \lambda_0 + \lambda_1 f(s) - 1 \tag{145}$$

$$\Rightarrow p(s) = \frac{1}{Z}e^{\lambda_1 f(s)}, \tag{146}$$

where $Z = \exp(1-\lambda_0)$[11]. The constants λ_0 and λ_1 can be found by solving the constraint equations. Then,

$$f(s) = s \Rightarrow p(s) = \frac{1}{Z}e^{\lambda_1 s} \qquad \text{Exponential (need } p(s)=0 \text{ for } s<T) \tag{147}$$

$$f(s) = s^2 \Rightarrow p(s) = \frac{1}{Z}e^{\lambda_1 s^2} \qquad \text{Gaussian.} \tag{148}$$

In other words, when we're looking for the first moment, the maximum entropy distribution is an exponential distribution, and when we're looking for the second moment (without constraint on the first moment), the maximum entropy distribution is Gaussian. Both results together imply that the maximum entropy point process (for a fixed mean arrival rate) is a homogeneous Poisson—independent, exponentially distributed ISIs.

9.3 Channel Coding

We now consider the conditional $P(R|S)$ which defines the *channel* linking S to R:

$$S \xrightarrow{P(R|S)} R. \tag{149}$$

The mutual information

$$\mathsf{I}[S;R] = \mathsf{KL}[P(S,R)\|P(S)P(R)] = \sum_{s,r} \overbrace{P(s,r)}^{P(s)P(r|s)} \log \frac{\overbrace{P(s,r)}^{P(s)P(r|s)}}{P(s)P(r)} = \sum_{s,r} P(s)P(r|s)\log\frac{P(r|s)}{P(r)} \tag{150}$$

depends on marginals $P(s)$ and $P(r) = \sum_s P(r|s)P(s)$ as well and thus is unsuitable to characterize the conditional alone. Instead, we characterize the channel by its *capacity* C, defined as

$$\mathsf{C}_{R|S} \coloneqq \sup_{P(s)} \mathsf{I}[S;R]. \tag{151}$$

[11] We should also have enforced the positivity of probability distributions, but it turns out not to matter in this case.

In other words, the capacity gives the theoretical maximum information that can be transmitted through the channel given all possible source distributions. In other words, the capacity is given by choice for $P(S)$, given the conditional distribution $P(R|S)$, that maximizes the mutual information between R and S. Clearly, this is limited by the properties of the noise.

As an aside, we define spike count or *noise correlation* to be the correlation between fluctuations in responses to a repeated stimulus, and *signal correlation* to be the correlation between two cell's mean responses to different stimuli.

9.3.1 The Joint Source-Channel Coding Theorem (JSCT)

This is a remarkable result of central importance to information theory. Consider the following framework

$$S \xrightarrow{\text{encoder}} \tilde{S} \xrightarrow[\mathsf{C}_{R|\tilde{S}}]{\text{channel}} R \xrightarrow{\text{decoder}} \hat{T}. \tag{152}$$

Any source ensemble S with entropy $\boldsymbol{H}[S] < \mathsf{C}_{R|\tilde{S}}$ can be transmitted perfectly (in sufficiently long blocks) with $P_{error} \to 0$. The proof is beyond our scope, but some of the key ideas involved are block coding, error correction, joint typicality, and random codes.

This leads to the *channel coding problem*, which seeks to find the *encoding* $P(\tilde{S}|S)$ (this may be deterministic) that maximizes $\mathbf{I}[S;R]$, given channel $P(R|\tilde{S})$ and source $P(S)$. By the data processing inequality (eq. 135) and the definition of capacity (eq. 151), we have

$$\mathbf{I}[S;R] \leq \mathbf{I}[\tilde{S};R] \leq \mathsf{C}_{R|\tilde{S}}. \tag{153}$$

By the JSCT, then, equality can be achieved (in the limit of increasing block size). Thus, $\mathbf{I}[\tilde{S};R]$ should saturate $\mathsf{C}_{R|\tilde{S}}$. The *Blahut Arimoto algorithm* is a method to find the $P(\tilde{S})$ that saturates $\mathsf{C}_{R|\tilde{S}}$ for a general discrete channel.

9.3.2 The Blahut-Arimoto Algorithm

Given a channel characterized by the conditional distribution $P(R|S)$, we wish to find a source distribution $P(S)$ that maximizes the mutual information $I(R;S)$. First, we show that

$$I(R;S) \geq \sum_{s,r} P(s)P(r|s) \log \frac{Q(s|r)}{P(s)} \tag{154}$$

for any conditional distribution $Q(S|R)$.

Proof. Call the expression on the right hand side of the inequality in eq. 154 $\tilde{I}(R;S)$. We can subtract $\tilde{I}(R;S)$ from $I(R;S)$:

$$\begin{aligned}
I(R;S) - \tilde{I}(R;S) &= \sum_{s,r} P(s)P(r|s) \log \frac{P(r|s)}{P(r)} - \sum_{s,r} P(s)P(r|s) \log \frac{Q(s|r)}{P(s)} && (155)\\
&= \sum_{r,s} P(r) \frac{P(s)P(r|s)}{P(r)} \log \frac{P(s|r)/P(r)}{Q(s|r)/P(s)} && (156)\\
&= \sum_{r} P(r) \sum_{s} \frac{P(s)P(r|s)}{P(r)} \log \frac{P(s)P(r|s)/P(r)}{Q(s|r)} && (157)\\
&= \sum_{r} P(r) \sum_{s} P(s|r) \log \frac{P(s|r)}{Q(s|r)} && (158)\\
&= \sum_{r} P(r) D_{KL}[P(S|r)||Q(S|r)] && (159)\\
&\geq 0, && (160)
\end{aligned}$$

where the inequality is due to the non-negativity of the KL divergence. Note that equality is achieved when $Q(S|R) = P(S|R)$. □

We can use this result to derive an iterative algorithm to find the optimal $P(S)$. We'll do so using a method reminiscent of EM: alternating a saturation of the bound on the right hand side of eq. 154 with respect to Q and to $P(S)$.

Algorithm 1: Blahut-Arimoto

Initialize iteration counter $k = 0$, $P^0(s)$ (e.g., uniformly),
while *not converged* **do**
  update $Q(S|R)$**:**
    $Q^{k+1}(s|r) \leftarrow \frac{P(r|s)P^k(s)}{\sum_{s'} P(r|s')P^k(s')}$
  update $P(S)$**:**
    $P^{k+1}(s) \leftarrow \frac{\phi(s)}{\sum_{s'} \phi(s')}$, where $\phi(s) = \exp\left(\sum_r P(r|s) \log Q^{k+1}(s|r)\right)$
  check $\tilde{I}(Q^{k+1}, P^{k+1})$ for convergence
  $k \leftarrow k + 1$
end

we can see that $\tilde{I}$ is maximized with respect to Q when

$$Q(s|r) = P(s|r) = \frac{P(r|s)P(s)}{\sum_{s'} P(r|s')P(s')}. \tag{161}$$

To maximize $\tilde{I}$ with respect to $P(s)$, we can set up a Lagrangian:

$$\mathcal{L} = \tilde{I} + \lambda \left(\sum_{s'} P(s') - 1 \right), \tag{162}$$

where λ is a Lagrange multiplier preserving the measure of $P(s)$. We can then take the variational derivative to find the optimal $P(s)$:

$$\frac{\delta \mathcal{L}}{\delta P(s)} = \frac{\delta}{\delta P(s)} \left[\sum_{s',r} P(s')P(r|s') \log \frac{Q(s'|r)}{P(s')} + \lambda \left(\sum_{s'} P(s') - 1 \right) \right] \tag{163}$$

$$= \sum_r P(r|s) \log \frac{Q(s|r)}{P(s)} + \lambda \tag{164}$$

$$= \sum_r P(r|s) \log Q(s|r) - \sum_r P(r|s) \log P(s) + \lambda \tag{165}$$

$$= \sum_r P(r|s) \log Q(s|r) - \log P(s) \underbrace{\sum_r P(r|s)}_{=1} + \lambda \tag{166}$$

$$= \sum_r P(r|s) \log Q(s|r) - \log P(s) + \lambda = 0 \tag{167}$$

$$\Rightarrow \log P(s) = \sum_r P(r|s) \log Q(s|r) + \lambda \tag{168}$$

$$\Rightarrow P(s) = e^{\lambda} e^{\sum_r P(r|s) \log Q(s|r)} \tag{169}$$

$$\Rightarrow P(s) = \frac{\exp\left(\sum_r P(r|s) \log Q(s|r)\right)}{\sum_{s'} \exp\left(\sum_r P(r|s') \log Q(s'|r)\right)}. \tag{170}$$

This suggests the iterative process summarized in Algorithm 1. From the analysis above, we have that

$$Q^{k+1} = \operatorname*{argmax}_Q \tilde{I}(Q, P^k) \quad \text{and} \tag{171}$$

$$P^{k+1} = \operatorname*{argmax}_P \tilde{I}(Q^{k+1}, P), \tag{172}$$

so we know that each iteration will perform coordinate ascent on $\tilde{I}$, analogously to EM:

$$I(Q^k, P^{k-1}) = \tilde{I}(Q^k, P^{k-1}) \leq \tilde{I}(Q^k, P^k) \leq \tilde{I}(Q^{k+1}, P^k) = I(Q^{k+1}, P^k). \tag{173}$$

Finally, to show that this process will converge to a *unique* maximum, we need to show that $\tilde{I}$ is concave. Specifically, we need to show that for a fixed $P(S)$, $\tilde{I}$ is concave in $Q(S|R)$, and for a fixed $Q(S|R)$, $\tilde{I}$ is concave with respect to $P(s)$. For the former, we can simply observe from the form of $\tilde{I}$ (eq. 154) that $\log Q(S|R)$ is concave, so for fixed P, $\tilde{I}$ is concave with respect to Q. Similarly, as $P \log \frac{1}{P}$ is concave, $\tilde{I}$ is concave with respect to P when Q is fixed.

9.3.3 Entropy Maximisation & Histrogram Equalisation

A neuron transmitting a response, R, to some encoded stimulus, $\tilde{S}$, might plausibly be understood by trying to maximise the mutual information between signal and response:

$$I[\tilde{S};R] = \underbrace{H[R]}_{\text{marginal entropy}} - \underbrace{H[R|\tilde{S}]}_{\text{noise entropy}} \tag{174}$$

If the noise entropy is small and constant perhaps we can instead just consider maximising the marginal entropy.

$$\frac{\delta H[R]}{\delta p(r)} = \frac{\delta}{\delta p(r)}[-\int_0^{r_{\max}} drp(r)\log p(r) - \mu\int_0^{r_{\max}}(p(r)-1)dr] = \begin{cases} -\log p(r) - 1 - \mu & 0 < r < r_{\max} \\ 0 & \text{else} \end{cases} \tag{175}$$

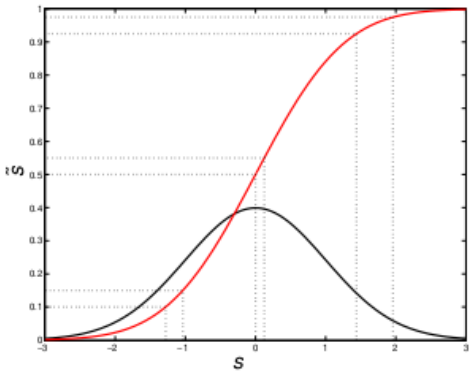
$$p(r) = \begin{cases} \frac{1}{r_{\max}} & 0 < r < r_{\max} \\ 0 & \text{else} \end{cases} \tag{176}$$

So, if we assume there's a relatively small noise source and the signal is encoded through some function, $\tilde{s} = f(s)$:

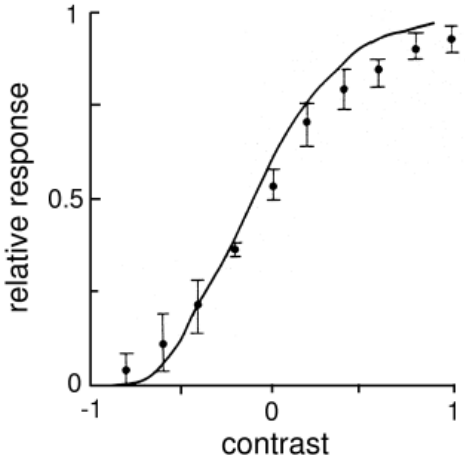
$$\frac{1}{r_{\max}} = p(r) \approx p(\tilde{s}) = p(s)\frac{ds}{d\tilde{s}} = \frac{p(s)}{f'(s)}$$

$$f(s) = r_{\max}\int_{\infty}^{s} p(s')ds'$$

i.e. the best choice for encoding function is the cumulative distribution, which sets equally probable bins in s to equal lengths of $\tilde{s}$, hence the term histogram equalisation, fig 9.2a. This principle has been used to explain the tuning response of fly retinal cells, fig 9.2b, [Laughlin et al., 1981].



(a) Example of histogram equalisation. Stimulus distributed according to the black curve are encoded by the red curve such that $\tilde{s}$ is uniform.



(b) Retinal cells appear to have such a response curve, [Laughlin et al., 1981].

9.3.4 Gaussian Channels & Water-Filling Algorithm

Gaussian channels are an oft studied simple model of a noisy channel. We're going to take a similar route to histogram equalisation, but this time we will explicitly model the effect of noise as Gaussian, and then seek to maximise the mutual information.

We begin with a preliminary, finding the differential entropy of a multivariate Gaussian, which we'll use later:

$$p(\boldsymbol{z}) = |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}}e^{-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})}$$

$$h(\boldsymbol{z}) = -\int d\boldsymbol{z} p(\boldsymbol{z})[-\frac{1}{2}\log|2\pi\boldsymbol{\Sigma}| - \frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})] \tag{177}$$

$$= \frac{1}{2}\log|2\pi\boldsymbol{\Sigma}| - \frac{1}{2}\int d\boldsymbol{z} p(\boldsymbol{z})\,\mathrm{Tr}[\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})(\boldsymbol{z}-\boldsymbol{\mu})^T] \tag{178}$$

$$= \frac{1}{2}\log|2\pi\boldsymbol{\Sigma}| - \frac{1}{2}\,\mathrm{Tr}[\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}] \tag{179}$$

$$= \frac{1}{2}\log|2\pi e\boldsymbol{\Sigma}| = \frac{1}{2}\log(2\pi e)^d|\boldsymbol{\Sigma}| \tag{180}$$

We then assume that the encoded signal $\tilde{\boldsymbol{S}}$ is sent through a channel when it is corrupted with mean-zero Gaussian noise $\boldsymbol{Z}$ with covariance $\boldsymbol{K_z}$ to create the response $\boldsymbol{R} = \tilde{\boldsymbol{S}} + \boldsymbol{Z}$.

$$I[\tilde{\boldsymbol{S}}, \boldsymbol{R}] = h(\boldsymbol{R}) - h(\boldsymbol{R}|\tilde{\boldsymbol{S}}) \tag{181}$$

$$= h(\boldsymbol{R}) - h(\boldsymbol{Z}|\tilde{\boldsymbol{S}}) \tag{182}$$

$$= h(\boldsymbol{R}) - \frac{1}{2}\log(2\pi e)^d|\boldsymbol{K_z}| \tag{183}$$

If there are no further constraints then maximising the mutual information, $I[\tilde{\boldsymbol{S}}, \boldsymbol{R}]$, just means maximising the differential entropy, $h(\boldsymbol{R})$, and both tend to infinity. We therefore constraint the power in the signal: $\frac{1}{n}\sum_i \tilde{s}_i = \frac{1}{n}\,\mathrm{Tr}[\tilde{\boldsymbol{S}}\tilde{\boldsymbol{S}}^T] \leq P$. This constrains the variance of the response:

$$\langle R^2\rangle = \langle(\tilde{S}+Z)^2\rangle \leq P + K_z$$

The distribution with largest entropy given constrained variance is the normal, therefore we know that the differential entropy of R will be smaller or equal to the differential entropy of a normal variable with covariance $P + K_z$:

$$h(R) \leq h(\mathcal{N}(0, P+K_z)) = \frac{1}{2}\log 2\pi e(P+K_z)$$

And hence:

$$I[\tilde{S}, R] \leq \frac{1}{2}\log 2\pi e(P+K_z) - \frac{1}{2}\log 2\pi e K_z = \frac{1}{2}\log 2\pi e(1+\frac{P}{K_z})$$

Therefore, the channel capacity, which is defined as the maximal mutual information, is:

$$C = \frac{1}{2}\log 2\pi e(1+\frac{P}{K_z})$$

And this capacity is achieved if $\tilde{S} \sim \mathcal{N}(0, P)$, such that $R \sim \mathcal{N}(0, P+K_z)$.

10 Neural Encoding

Basically the whole business of systems neuroscience is built around the following process: come up with something you think should be represented in the brain and show it has some neural correlate.

That this has worked at all is perhaps surprising; it certainly doesn't sound foolproof. Part of the difficulty is that we're simultaneously trying to answer two questions:

1. What quantities are represented/encoded/present in the brain?
2. How is are those quantities being encoded?

Either question would be (relatively) easy if we knew the answer to the other. Unfortunately we do not, so we either guess an encoding scheme and try to extract the quantity or we guess a quantity that is being encoded and try to figure out the encoding scheme. It is also unclear to me whether searching for neural correlates of things will eventually lead to us understanding the brain, but I can't come with a better approach so it will have to do.

Part of the reason we're able to do this is the modularity of the brain. We think there are visual regions, that are largely invariant to sound, but respond to specific aspects of the visual scene. This means people like Hubel and Wiesel can win Nobel prizes for finding the encoding of orientation.

There is a further conceptual difficulty with this approach: just finding a correlation between some phenomenon in the brain and an external variable doesn't mean it is something the brain uses. Experiments in which we are able to perturb neuronal activity and observe whether they match predictions go some way to answering this, though it feels like these techniques are still very much in their infancy. Yet it remains a key technique. We will run through a series of different encoding models that people use to find the external correlate of an observed neural response.

If the model concerns spike times they generally try to work out something like, $P(\text{spike}|\text{stim}, \text{history})$. This is intractable to measure directly for all stimulus settings, therefore you need some way to estimate the probability, either by fitting a lower dimensional parametric model, or by selecting the stimuli efficiently. This usually proceeds in one of three ways:

1. Forget the history dependence and find $P(\text{spike}|\text{stim})$
2. Average over trials, compute the PSTH, then try to fit it.
3. Attempt to capture history effects in some simple model.

Another classic approach to the encoding problem is tuning curves, which plot the firing rate as a function of stimulus input. This is of course a crude summary (brains are dynamic), but has given insight in the past.

Finally, more exploratory approaches assume a particular form of encoding, then look for the corresponding stimulus. We begin by looking at one of these models.

10.1 Linear Models

10.1.1 The Spike-Triggered Average

We begin with the simplest type of model—a linear one. Given a movie composed of a series of stimulus frames presented to an animal and a time-aligned series of spikes in response, we can average the stimulus frames in a certain window of time preceding each spike as a simple estimate of the average stimulus that induces a particular neuron to fire. This is called the *spike-triggered average* (STA)—the process used to calculate it can be visualized in Figure 10.2A. The STA can be interpreted from both a decoding and encoding perspective. From a decoding perspective, it is a way of constructing the mean stimulus given a spike; in other words, the mean of $P(s|r = 1)$. From an encoding perspective, the STA can be used as a *predictive filter* to predict the neural response. In other words, we can interpret the firing rate of the neuron at time t as being generated via the convolution

$$r(t) = \int_0^T s(t-\tau)w(\tau)d\tau, \tag{184}$$

where w is the STA filter and T is the window size. We can also rewrite this convolution as a matrix multiplication with a *design matrix* S composed of shifted windows of the stimulus (see Figure 10.1):

$$S\mathbf{w} = \mathbf{r}, \tag{185}$$

where $\mathbf{r}$ is a vector containing $r_T, r_{T+1}, \ldots$. Also note that $\mathbf{w}$ is flipped (for the convolution–see Figure 10.1). We can solve this via linear regression, yielding a closed-form expression for the STA:

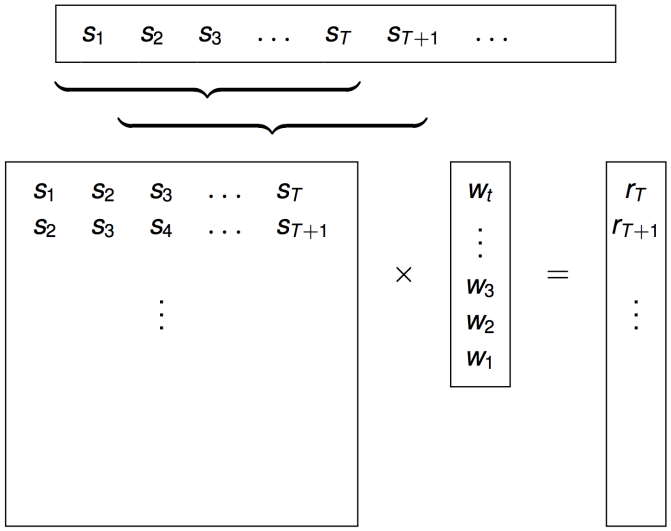


Figure 10.1: A visualization of the linear regression used to compute the STA.

$$\mathbf{w} = (\underbrace{S^\mathsf{T} S}_{\Sigma_{SS}})^{-1} \quad \underbrace{S^\mathsf{T}\mathbf{r}}_{\text{"unwhitened" STA}}, \tag{186}$$

where Σ_{SS} is the stimulus covariance matrix—pre-multiplying $S^\mathsf{T}\mathbf{r}$ with its inverse "whitens"" the STA, accounting for a non-normalized stimulus distribution.

As $r(t)$ is modelled as the result of a convolution, we can also use the Fourier transform and the convolution theorem to construct $\mathbf{w}$:

$$W(\omega) = \frac{S(\omega)R(\omega)}{|S(\omega)|^2}, \tag{187}$$

where the Fourier transform of the stimulus $s(\omega)$ usually becomes diagonal in the Fourier basis. This form is usually referred to as a *Wiener filter*.

An important property of the STA is that is an unbiased estimator of the true filter for a spherical (Gaussian) stimulus distribution. This is known as *Bussgang's Theorem*. Stated more formally, Bussgang's Theorem is as follows:

Theorem 1. *If we have samples $\{\mathbf{x}_i, y_i\}$, where y_i is a random variable whose expectation is given by $\mathbb{E}[y_i|\mathbf{x}_i] = f(\mathbf{w}\cdot\mathbf{x}_i)$, then the cross-correlation $\sum_i y_i\mathbf{x}_i$ (i.e., the "spike-triggered average" if y_i is binary) provides an unbiased estimate of $\alpha\mathbf{w}$ (i.e., $\mathbf{w}$ times an unknown constant α if:*

i. *$P(\mathbf{x})$ is spherically symmetric, where we define spherical symmetry to mean that*

$$\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n,\ ||\mathbf{x}_1|| = ||\mathbf{x}_2|| \Rightarrow P(\mathbf{x}_1) = P(\mathbf{x}_2). \tag{188}$$

ii. *$\mathbb{E}[y\mathbf{x}] \neq \mathbf{0}$ (i.e., the expected STA is not zero).*

Proof. We want $\mathbb{E}[\sum_i y_i\mathbf{x}_i] = \alpha\mathbf{w}$. We can write

$$\mathbb{E}\left[\sum_i y_i\mathbf{x}_i\right] = \sum_i \mathbb{E}[y_i\mathbf{x}_i] \tag{189}$$

$$= \sum_i \left[\sum_{j,k} y_j\mathbf{x}_k P(y_j, \mathbf{x}_k)\right] \tag{190}$$

$$= \sum_i \left[\sum_{j,k} y_j\mathbf{x}_k P(y_j|\mathbf{x}_k)P(\mathbf{x}_k)\right] \tag{191}$$

$$= \sum_i \left[\sum_k \mathbf{x}_k P(\mathbf{x}_k) \underbrace{\sum_j y_j P(y_j|\mathbf{x}_k)}_{\mathbb{E}[y_j|\mathbf{x}_k]}\right] \tag{192}$$

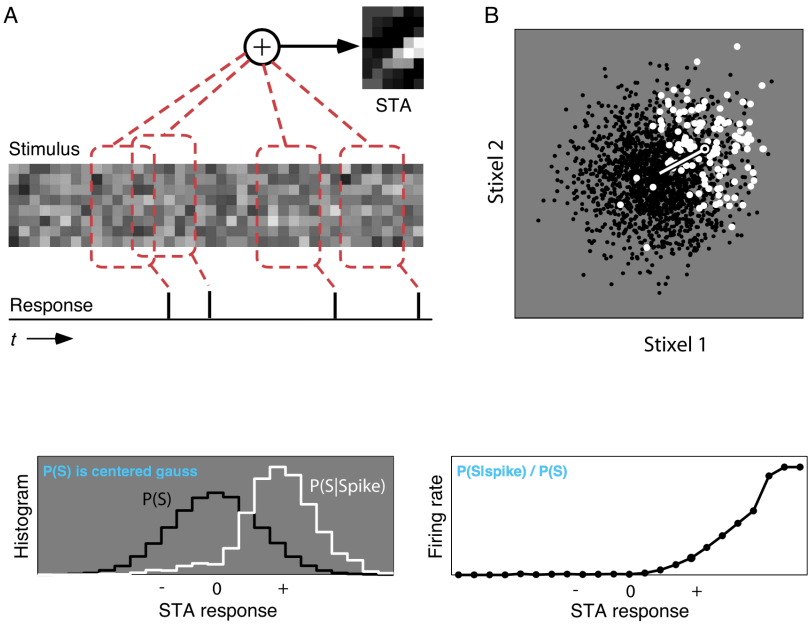


Figure 10.2: Visualization of the STA for a white noise stimulus. (A) The STA is the average of the aggregated stimulus a set window-length of time before each spike. (B) The dots are stimulus values in 2D space—the black dots are those that did not trigger a spike, and the white dots are those that did. The white vector from the center is the STA (note that it maps to the center of the white cloud of points). The probability of spiking is proportional to the projection of a stimulus point onto this vector. By rotational symmetry, the expected value of the spike-inducing (white dot) stimuli lies along this vector, thus making it an unbiased estimate of the maximum likelihood value. (Bottom Left) A histogram of all the stimulus values filtered by the STA $P(S)$ (a normal distribution centered at zero) and the spike-triggered stimuli $P(S|\text{spike})$—there is a noticeably higher response for stimuli that induced a spike. (Bottom Right) The resulting firing rate for an LN neuron (more on this below). If $P(\text{spike}) = 1$, then $P(\text{spike}|S) = P(S|\text{spike})/P(S)$. The firing rate curve is thus obtained by dividing the two histograms on the bottom left.

$$= \sum_i \left[\sum_k \mathbf{x}_k P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) \right]. \tag{193}$$

Then by spherical symmetry, we observe that every $\mathbf{x}_k$ in the sum must have a corresponding $\mathbf{x}_{k'}$ equidistant from the origin (and thus with equal probability) that is symmetric about the STA $\mathbf{w}$. Their sum is then a scaled version of $\mathbf{w}$. That is, $\mathbf{x}_k + \mathbf{x}_{k'} = \beta \mathbf{w}$ for some $\beta > 0$. We have

$$\mathbb{E}\left[\sum_i y_i \mathbf{x}_i\right] = \sum_i \left[\sum_k \mathbf{x}_k P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) \right] \tag{194}$$

$$= \frac{1}{2} \sum_i \left[\sum_k \mathbf{x}_k P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) + \sum_{k'} \mathbf{x}_{k'} P(\mathbf{x}_{k'}) f(\mathbf{w} \cdot \mathbf{x}_{k'}) \right] \tag{195}$$

$$= \frac{1}{2} \sum_i \left[\sum_{k,k'} (\mathbf{x}_k + \mathbf{x}_{k'}) P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) \right] \tag{196}$$

$$= \frac{1}{2} \sum_i \left[\sum_k P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) \beta \mathbf{w} \right] \tag{197}$$

$$= \frac{\beta}{2} \left[\sum_{i,k} P(\mathbf{x}_k) f(\mathbf{w} \cdot \mathbf{x}_k) \right] \mathbf{w} \tag{198}$$

$$= \alpha \mathbf{w}, \tag{199}$$

as desired. □

If data is elliptically distributed and not spherical, it can be whitened—the resulting linear regression weights (eq. 186) are unbiased. However, linear weights are *not* necessarily maximum likelihood (or otherwise optimal),

even for spherical/elliptical stimulus distributions. They may also be biased for general stimuli (binary/uniform or natural).

10.1.2 Limitations

The (whitened) STA gives the minimum squared-error linear model. However, as expected, there are issues. First, as is frequently true in the case of linear regression, there is the potential for overfitting and the need for regularization. The standard methods typically used in regression can be applied here. Second, this framework does nothing to prevent the model from predicting negative firing rates, though this can be ameliorated by re-framing the predictions as relative to some background or baseline firing rate. Third, and perhaps most obvious, real neurons are simply not linear. This doesn't mean that the STA isn't useful, however. It is highly interpretable and can provide an unbiased estimate of cascade filters used in nonlinear models (see later).

In general, though, we'd like to be able to make an absolute measure of model performance. Two things make this difficult. First, measured responses can never be predicted perfectly, even in theory, as the measurements themselves contain inherent noise. Second, even if we discount this, a model may predict poorly because either (i) it is the wrong model, or (ii) the parameters are mis-estimated due to noise. However, there are several possible approaches at our disposal for model evaluation.

First, we can compare the mutual information between the true response and the model prediction $\mathsf{I}[\mathbf{r};\hat{\mathbf{r}}]$ to the mutual information between the true response and the stimulus $\mathsf{I}[\mathbf{r};\mathbf{s}]$, although mutual information estimators are biased. Second, we can compared $\mathbb{E}\left[(\mathbf{r}-\hat{\mathbf{r}})^2\right]$ to $\mathbb{E}\left[(\mathbf{r}-\bar{\boldsymbol{\lambda}})^2\right]$, where $\bar{\boldsymbol{\lambda}}$ is the PSTH gathered over a very large number of trials. However, it may require an impractical amount of data to accurately estimate the PSTH. Third, we can compare the *predictive power* to the *predictable power* (similar to ANOVA methods), where power is meant to mean the percent of variance that could be predicted given the perfect model.

10.2 Nonlinear Models

Despite these corrections and analyses, linear models often simply fail to predict neural responses accurately. Here, we'll discuss several nonlinear approaches to neural encoding: Volterra/Wiener expansions, linear-nonlinear (Wiener) cascades, and nonlinear (Hammerstein) cascades.

10.2.1 Volterra/Wiener Expansions

The Volterra Expansion The *Volterra* expansion is a polynomial-like expansion for functionals (or operators). Let $y = F[x(t)]$, where y is the response and x is the stimulus. Then we approximate the response as

$$\begin{aligned} y(t) \approx k^{(0)} &+ \int k^{(1)}(\tau)x(t-\tau)\,d\tau + \int\int k^{(2)}(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)\,d\tau_1\,d\tau_2 \\ &+ \int\int\int k^{(3)}(\tau_1,\tau_2,\tau_3)x(t-\tau_1)x(t-\tau_2)x(t-\tau_3)\,d\tau_1\,d\tau_2\,d\tau_3 + \cdots, \end{aligned} \tag{200}$$

where $k^{(n)}$ are a series of kernels such that the approximation to the functional becomes exact as the number of terms goes to infinity. For finite expansions, however, the relationship of the kernels to the functional is not straightforward—the values of lower-order kernels change as the order of the expansion is increased. A polynomial kernel is a typical choice. For estimation, it's easy to see that the model is linear in the kernels themselves, so it can be estimated just like a linear (first-order) model with an expanded "input."

The Wiener Expansion The *Wiener expansion* gives functionals of different orders that are *orthogonal* for white noise input $x(t)$. We write this as a series of functionals $G_0, G_1, \ldots,$

$$\begin{aligned} G_0[x(t);h_0] &= h_0 \\ G_1[x(t);h_1] &= \int h_1(\tau)x(t-\tau)\,dx\,d\tau \\ G_2[x(t);h_2] &= \int\int h_2(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)\,d\tau_1\,d\tau_2 - P\int h_2(\tau_1,\tau_1)\,dx\,d\tau_1 \\ &\cdots \end{aligned} \tag{201}$$

such that it's easy to verify that $\langle G_i[x(t)]G_j[x(t)]\rangle = 0$ for $i \neq j$. This orthogonality allows the kernels to be estimated independently. However, they depend on the stimulus.

10.2.2 Linear-Nonlinear Cascades: STC and MID

A linear-nonlinear (LN) cascade model is a hierarchical model of encoding in which a linear weighting is applied to an input stimulus, followed by a nonlinear function (usually a non-negative function to prevent negative firing rates). The output of the nonlinearity is then interpreted as the mean rate in a Poisson process. In the simplest form of LN model, there is only a single linear filter—usually the STA (depicted schematically in Figure 10.3, left panel). However, the firing distribution changes along relevant directions in stimulus space (and, usually, along all linear combinations of relevant directions), and therefore more filters are typically needed (Figure 10.3, right panel). The mean of the distribution is captured by the STA, but this only provides one filter. Additional filters can be found via the spike-triggered covariance or the binned (or kernelized) KL divergence, as we'll see below.

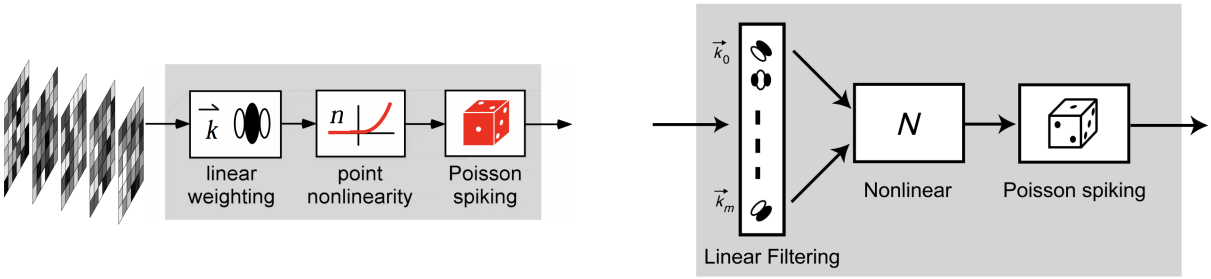


Figure 10.3: Schematics for a single-filter LNP model (left) and a multi-filter LNP model (right).

Spike-Triggered Covariance The *spike-triggered covariance* (STC) is exactly what it sounds like—the covariance of the stimuli occurring in a fixed time window before each spike in the response. By finding its eigenvectors, we can identify the directions of maximum variance in stimulus space corresponding to a spike in the response. This can be done as follows. First, given the stimulus design matrix X (each row is a shifted window of the stimulus, as defined above), project out the STA:

$$\tilde{X} := X - (X\mathbf{k}_{sta})\mathbf{k}_{sta}^{\mathsf{T}}. \tag{202}$$

The prior and spike covariances are then

$$C_{prior} = \frac{1}{N}\tilde{X}^{\mathsf{T}}\tilde{X}; \qquad C_{spike} = \frac{1}{N_{spike}}\tilde{X}^{\mathsf{T}}\mathrm{diag}(Y)\tilde{X}. \tag{203}$$

The STC Λ is then simply

$$\Lambda = C_{prior} - C_{spike}. \tag{204}$$

Then to acquire m filters, we simply find the directions (eigenvectors) with the greatest change in variance:

$$\mathbf{k}_1, \ldots, \mathbf{k}_m = m\text{-}\operatorname*{argmax}_{\|\mathbf{v}\|=1} \mathbf{v}^{\mathsf{T}}\Lambda\mathbf{v}. \tag{205}$$

In other words, we want the eigenvectors of Λ with the largest (absolute) eigenvalues. This process is visualized in Figure 10.4. To derive firing rates and reconstruct the nonlinearity, we can then repeat the same histogram-based process as depicted in Figure 10.2, except in this case, the histograms (and firing rates) are computed in the direction of each filter (eigenvector). This can be visualized in Figure 10.5.

It's important to note that that STC requires that the nonlinearity alter the variance, as otherwise $C_{prior} = C_{spike} \Rightarrow \Lambda = 0$. If this is the case, the derived subspace is unbiased provided that the stimulus distribution is (i) radially (elliptically) symmetric and (ii) independent. In other words, the STC prediction is unbiased if the stimulus distribution is Gaussian and the stimulus dimensions are independent—otherwise, the STC filters will likely fail to identify relevant dimensions. It may be possible to correct for non-Gaussian stimulus by transformation, subsampling, or weighting (with the latter two options coming at the cost of some variance).

Maximally Informative Dimensions It's also possible to consider non-parametric nonlinearities. The method of *maximally informative dimensions* (MID) extends the variance difference idea used to compute the STC to arbitrary differences between marginal and spike-conditioned stimulus distributions:

$$\mathbf{k}_{MID} = \operatorname*{argmax}_{\mathbf{k}} \mathsf{KL}[P(\mathbf{k}^{\mathsf{T}}\mathbf{x}) \| P(\mathbf{k}^{\mathsf{T}}\mathbf{x}|spike)]. \tag{206}$$

In other words, MID finds the dimensions of the stimulus that are most predictive (informative about) whether or not the cell will spike. In practice, measure the KL divergence requires binning or smoothing—this turns out to be equivalent to fitting a non-parametric nonlinearity (via binning or smoothing). This is difficult to use for high-dimensional LNP models, but the maximum-likelihood viewpoint suggests separable or "cylindrical" basis functions. We can also consider parametric nonlinearities, which leads us to the generalized linear model.

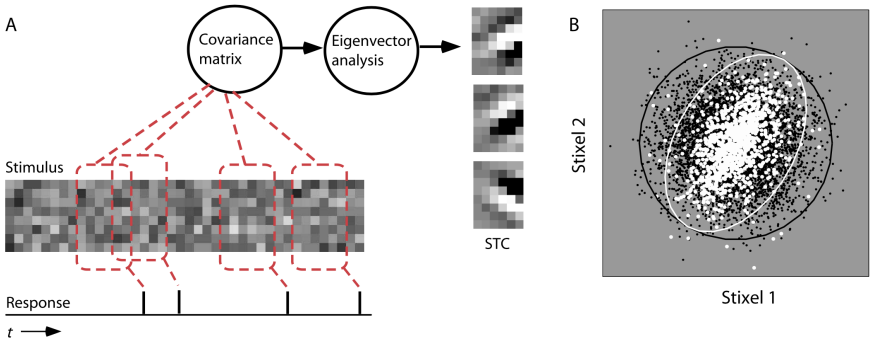


Figure 10.4: The spike-triggered covariance (STC). (A) A visualization of the flow of computation for the STC—its eigenvectors form effective filters for LN models. (B) A visualization of the principle 2D subspace in stimulus space obtained via the eigenvectors of the STC.

10.2.3 Generalized Linear Models

Generalized linear models (GLMs) are LN models with specified (parametric) nonlinearities and exponential family noise. In general, for a monotonic nonlinear function $f(\cdot)$, we visualize the spiking process as

$$y \sim P[f(\beta \mathbf{x})], \tag{207}$$

where β is a weight and $P(\cdot)$ is an exponential family mass function. It turns out that the continuous time point process likelihood with GLM-like dependence of λ on covariates is approximated in the limit of the bin width $\to 0$ by either a Poisson or Bernoulli GLM.

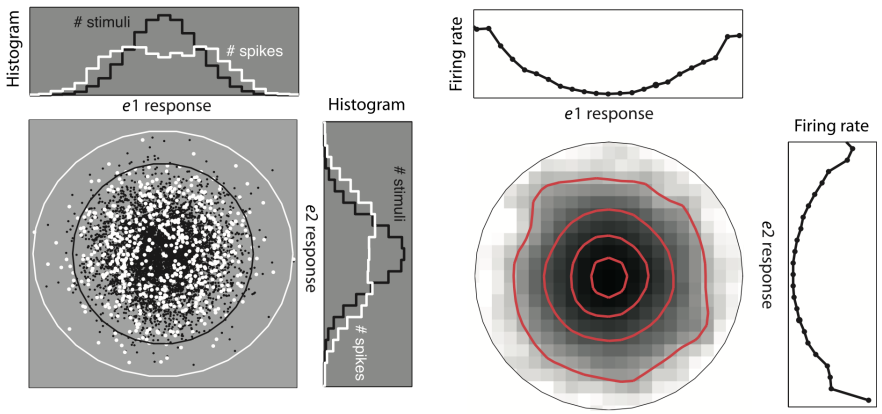


Figure 10.5: Reconstructing the nonlinearity along the directions of the eigenvectors of the STC.

If we choose P to be a Poisson distribution, setting $f(\cdot) = \exp(\cdot)$ is called *canonical* for the distribution—in other words, the natural parameters are $\beta \mathbf{x}$. In general, choosing the canonical link functions (nonlninearities) for a given distribution gives a concave likelihood—in other words, there's a unique maximum. For the Poisson distribution, this property generalized to any f which is convex and log-concave:

$$\log P(y|x) = \log \frac{f(\beta \mathbf{x})^y}{y!} e^{-f(\beta \mathbf{x})} = y \log f(\beta \mathbf{x}) - f(\beta \mathbf{x}) - \log y!. \tag{208}$$

This family of link functions includes, for example: thresholded linear functions, threshold-polynomial, and "soft-threshold": $f(z) = \alpha^{-1} \log(1 + e^{\alpha z})$ (e.g., the softplus). To find the maximum likelihood parameters (i.e., β and any parameters in the nonlinearity—which could be a neural network, for example), we can use gradient ascent on the log-likelihood or iteratively reweighted least-squared (IRLS). Regularization by L_2 or L_1 (sparse) penalties—which corresponds to MAP estimation with Gaussian/Laplacian priors—preserves concavity.

The GLM with History-Dependence We can also add recurrent computation to a GLM by including a history-dependent feedback term in the computation, such that conditional intensity/spike rate λ for an exponential link function is given by

$$\lambda(t) = f(\mathbf{k}^\mathsf{T} \mathbf{x}(t) + \mathbf{h}^\mathsf{T} \mathbf{y}(t)) = e^{\mathbf{k}^\mathsf{T} \mathbf{x}(t)} \cdot e^{\mathbf{h}^\mathsf{T} \mathbf{y}(t)}. \tag{209}$$

This model is visualized in the top panel of Figure 10.6. In other words, the rate is a product of stimulus- and spike-history-dependent terms, and the output is longer a Poisson process. This is also known as a soft-threshold integrate-and-fire (IF), because unlike traditional IF models in which the firing jumps from zero once the voltage (or filter output) reaches a threshold value, here the spike rate increases smoothly with the filter output (Figure 10.6, bottom). The shape of the post-spike filter (or "waveform") has a strong effect on the spiking behavior of the neuron—note that when it is zero everywhere, the model reduces to the standard GLM, and the spiking pattern is irregular, following a Poisson process. Some example waveforms and the resulting dynamic behaviors for a constant stimulus are plotted in Figure 10.7.

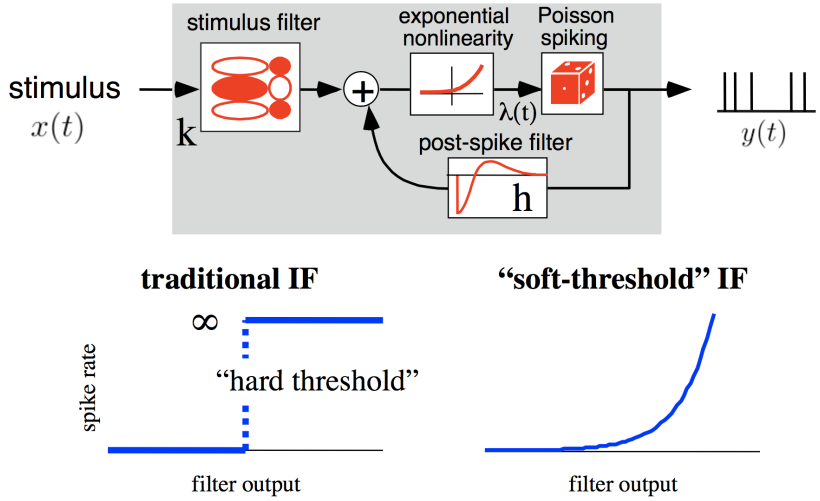


Figure 10.6: The GLM with history dependence. (Top) A model schematic. (Bottom) Spike rate as a function of filter output for a traditional integrate-and-fire (IF) model and the soft-threshold GLM.

This framework also allows for the coupling of multiple neurons, wherein the activity of one neuron is fed as additional input to the nonlinearity of another (Figure 10.8).

Looking beyond LN models, the idea of responses depending on one or a few linear stimulus projections has been dominant, but cannot capture all nonlinearities. Some experimentally-observed phenomena hint at shortcomings of the LN framework:

1. Contrast sensitivity might require normalization by $\|\mathbf{s}\|$.
2. Linear weightings may depend on *units* of stimulus measurement, such as amplitude, energy, and thresholds. This perspective is used in *Hammerstein cascade* models.
3. Neurons, particularly in the auditory system, are known to be sensitive to combinations of inputs, evident in phenomena such as forward suppression, spectral patterns, and time-frequency interactions.
4. Experiments with realistic stimuli have revealed nonlinear sensitivity to only parts of a stimulus.

Many of these questions can be addressed via multilinear (Cartesian tensor) approaches, which we do not cover here.

10.3 Encoding Tips

- In general, compute the STA via $\mathbf{k} = \langle X^\top r \rangle$, where $\langle \cdot \rangle$ is expectation.
- Don't forget that in computing the STC, projecting out the STA in the first step reduces the number of dimensions by one, and that the resulting vector(s) are orthogonal to the STA.

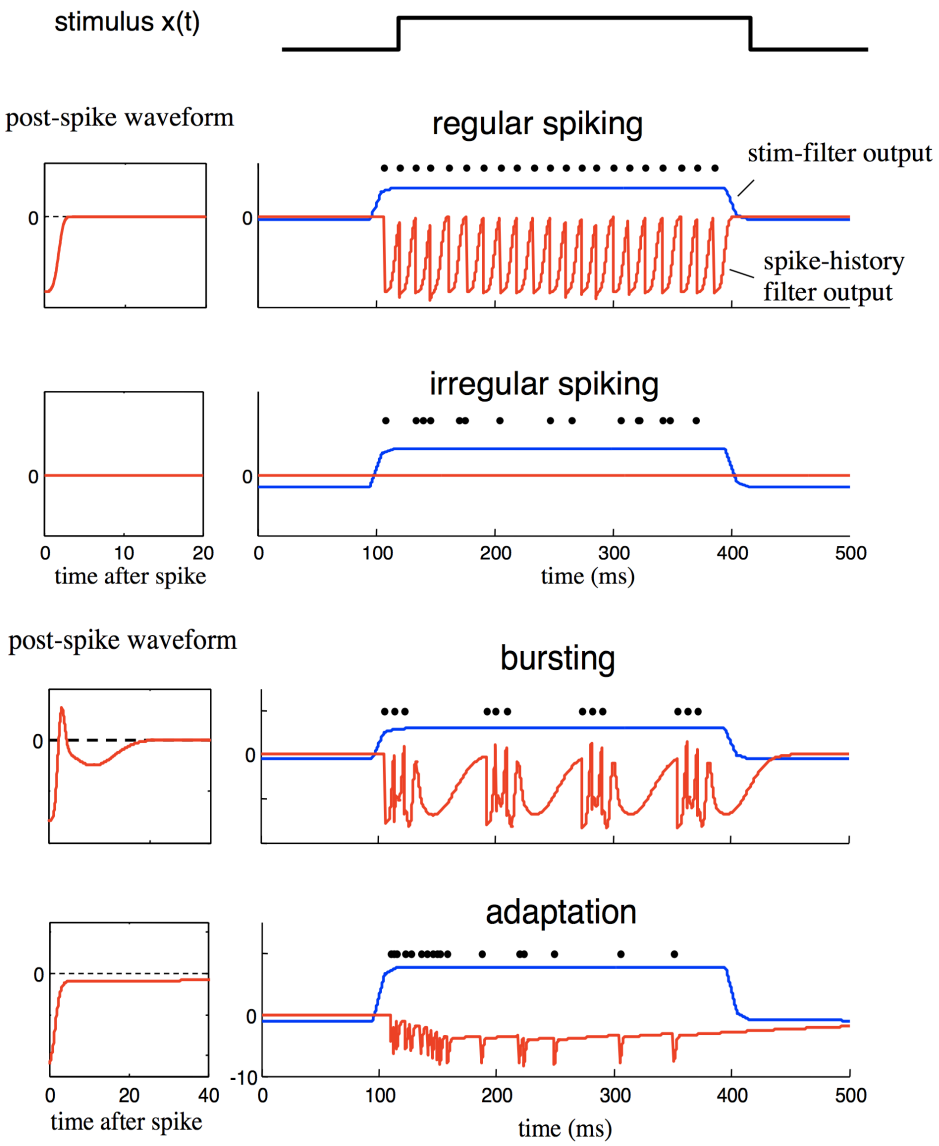


Figure 10.7: The dynamics of a GLM with history dependence in response to constant stimulus depend on the shape of the post-spike filter. Negative values in the post-spike filter inhibit firing, and positive values promote it—for example, in the "bursting" model, there is a brief period of inhibition following by positive feedback (causing the burst) and then negative feedback to stop the burst. In the regular spiking example, the inhibitory period after a spike sets the interval length by inhibiting spiking for a short time.

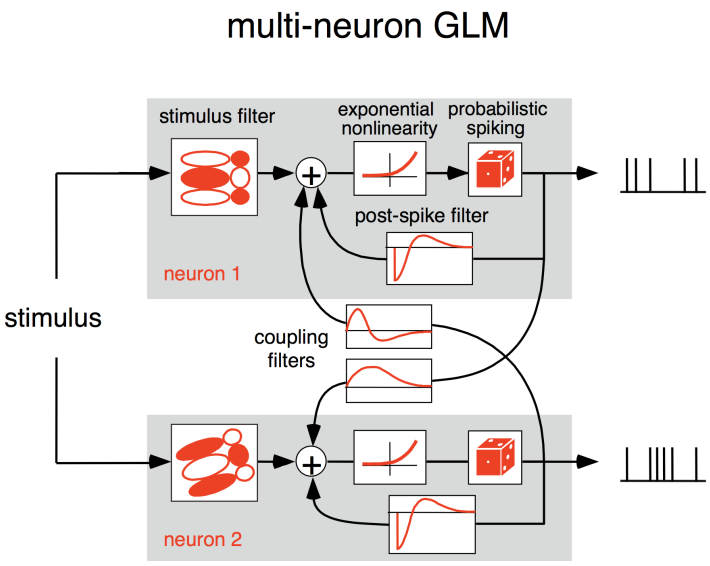


Figure 10.8: A multi-neuron GLM with history dependence.

11 Population Coding

Here, we focus on methods for understanding the population-level behavior of neurons, rather than seeking to understand the encoding properties of one or a few neurons at a time. Most neural codes are *distributed*, in that each neuron fires for a range of stimulus values and computations, and often, population activity must be considered altogether to identify (decode) the input stimulus.

There are number of factors that make this complicated, not the least of which is the simple fact that neurons are noisy: there are synaptic release failures, branch-point spike propagation failures, channel noise, and network chaos with which to contend. Neurons often display *heterogeneous dynamics*—there is no simple rule that governs a neuron's response, and the time course of its activity can vary significantly over time. They also display *mixed selectivity*, firing in response to multiple features of a given task. Overall network computation is carried in the coordinated activity of many neurons. It is usually assumed that relevant information is encoded in lower-dimensional latent variables.

Distributed population coding has several advantages. The code can build in redundancy (robustness) to account for sources of noise (including events like cell death), and so maintain high precision in identifying stimuli. Using many neurons with distributed codes might both encourage and oppose efficient representations. Consider just the difference between some Cartesian code compared to a multi-dimensional distributed code. The former could be considered efficient but not able to handle problems with multiple values, while the latter can also represent multiple values, but may require more neurons to do so (and hence be less efficient, see Fig. 11.3). In terms of other utilities, there is the capacity for processing of signals and memory, and even for local computation (scalar coding of a variable). Finally, the code might also represent concepts related to a stimulus or stimulus distribution, like confidence, uncertainty and multipotent plans.

11.1 Optimal Encoding and Decoding

There are two common varieties of question that can be asked about population codes. First, given assumed encoding functions, how well can we (or downstream areas) decode the encoded stimulus value? Second, what encoding schemes would be optimal, in the sense of allowing decoders to estimate stimulus values as well as possible? Before considering these questions, its useful to first formulate some ideas about rate coding in the context of single cells.

11.1.1 Rate Coding and Tuning Curves

In rate coding (where we capture instantaneous intensities in response to a time-varying stimulus), we imagine that the firing rate r of a cell represents a single (possibly multidimensional) stimulus value s at any one time:

$$r = f(s). \tag{210}$$

Even if s and r are embedded in time-series, we assume (i) that coding is instantaneous (with a fixed lag) and (ii) that r (and therefore s) is constant over a short time Δ. The actual number of spikes n produced in Δ is then taken to be distributed around $r\Delta$, often according to a Poisson distribution.

The function $f(s)$ is known as a *tuning curve*. Some commonly assumed forms are

- Gaussian (pops up surprisingly often): $f(\theta) = r_0 + r_{max} \exp\left[\frac{1}{2\sigma^2}(\theta - \theta_{pref})^2\right]$
- Cosine (requires Cartesian/tight directions – see comment on cricket cercal system; also used in motor cortex): $f(\theta) = r_0 + r_{max} \cos(\theta - \theta_{pref})$
- Wrapped Gaussian: $f(\theta) = r_0 + r_{max} \sum_n \exp\left[-\frac{1}{2\sigma^2}(\theta - \theta_{pref} - 2\pi n)^2\right]$
- von Mises ("Circular Gaussian"): $f(\theta) = r_0 + r_{max} \exp[\kappa \cos(\theta - \theta_{pref})]$
- Thresholded ramp: $f(\theta) = r_0 + \Theta(\theta - \theta_{thr}) r_{max} \rho \cdot (\theta - \theta_{thr})$
- Periodic (grid): $f(\theta) = f_1(\sin(2\pi\theta/\lambda))$

The thresholded cosine form can be used in a neat example explaining the cricket cercal system, which senses air currents via hairs which converge on a ganglion in the abdomen. The tuning curves occur in sets of 4 all $\frac{\pi}{2}$ apart and are akin to Cartesian coding. The $\frac{\pi}{2}$ relationship means that the dot product of adjacent vectors is 0, and the dot product of vectors π apart are negative reflections. This produces a rapid, potentially optimal scheme of decoding: the cricket can simple hop directly opposite and away from the wind!

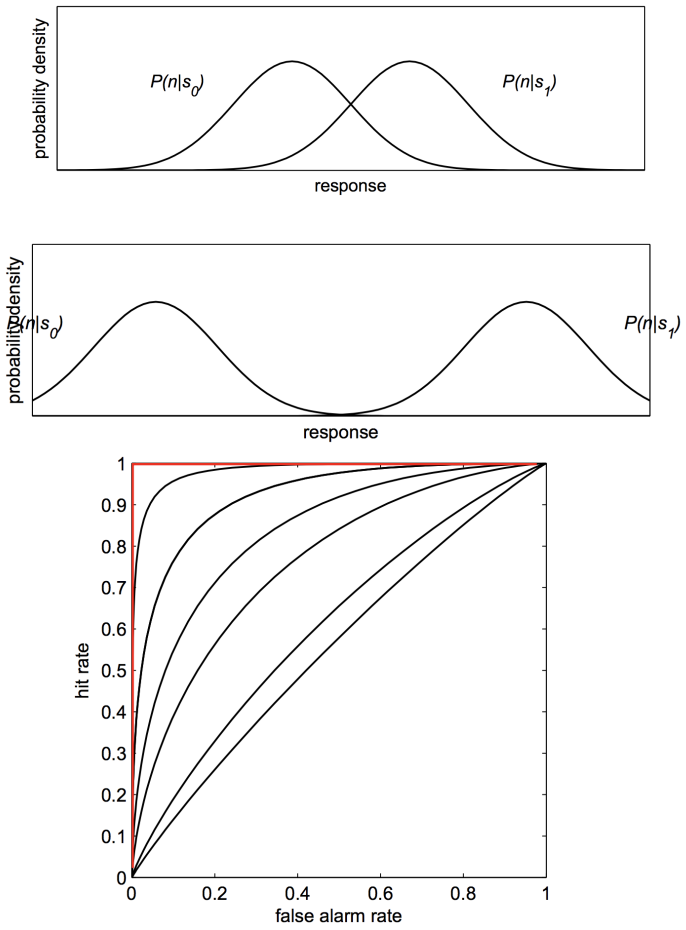


Figure 11.1: Discriminability of stimuli in a binary choice rask. As the overlap between the response distributions decreases (the distributions move further apart), the false positive rate at which the distributions can be distinguished decreases. In other words, when the distributions are completely apart, perfect classification can be achieved with zero false positives (i.e., the top left of the ROC plot), and when they complete overlap, classification happens at chance (the diagonal line).

11.1.2 Discrete Choices

Suppose we'd like to make a binary choice based on firing rate, e.g., the presence/absence of a signal, up/down, horizontal/vertical, etc. If we call a stimulus corresponding to one option s_0 and the other s_1, we can expect a bimodal distribution over the number of spikes $P(n|s)$: The overlap between these distributions then corresponds to how distinguishable two stimuli are—at high overlaps, decoding the stimulus is essentially at chance, but classification is much easier for lower overlaps. This relationship can be visualized via an ROC curve (Figure 11.1). Given $n_0 \sim P(n|s_0)$ and $n_1 \sim P(n|s_1)$, the area under the ROC curve (AUC) equals $P(n_1 > n_0)$. We can more formally define the *discriminability* d' for two equal variance Gaussians as

$$d' := \frac{\mu_1 - \mu_0}{\sigma}. \tag{211}$$

Then for any threshold

$$d' = \Phi^{-1}(1 - FA) - \Phi^{-1}(1 - HR), \tag{212}$$

where Φ is the standard normal CDF, FA is the false alarm (false positive) rate, and HR is the hit (true positive) rate. In other words, the discriminability is the difference in quantile functions. However, this definition is unclear for non-Gaussian distributions.

11.1.3 Continuous Estimation and the Fisher Information

The problem of decoding such a real-valued stimulus from firing rates is called *estimation*. Consider a one-dimensional stimulus that takes on continuous values (e.g., angle, contrast, direction, speed). Suppose a neuron fires n spikes in response to stimulus s according to some distribution (tuning curve)

$$P(n|f(s)\Delta). \tag{213}$$

Given an observation of n, we'd like to estimate s. In order to do so, it's useful to consider the limit given $N \to \infty$ measurements n_i all generated by the same true stimulus s^*. In other words, each time s^* is presented, we record the number n_i of resulting spikes. The log-posterior over the stimulus s is then given by

$$\log P(s|\{n_i\}) = \sum_i \log P(n_i|s) + \log P(s) - \log Z(\{n_i\}), \tag{214}$$

where Z is a normalizer. Taking $N \to \infty$, we have

$$\frac{1}{N} \log P(s|\{n_i\}) \to \langle \log P(n|s) \rangle_{n|s^*} + 0 - \log Z(s^*), \tag{215}$$

and so

$$\begin{aligned} P(s|\{n_i\}) &\to \frac{1}{Z} e^{N \langle \log P(n|s) \rangle_{n|s^*}} \\ &= \frac{1}{Z'} e^{-N[\langle \log P(n|s^*) \rangle_{n|s^*} - \langle \log P(n|s) \rangle_{n|s^*}]} \\ &= \frac{1}{Z'} e^{-N \mathsf{KL}[P(n|s^*) \| P(n|s)]}. \end{aligned} \tag{216}$$

Note that the normalizer Z is changing from line to line, but never depends on s. We can now do a Taylor expansion around the KL divergence in s around s^*:

$$\begin{aligned} \mathsf{KL}[P(n|s^*) \| P(n|s)] &= -\langle \log P(n|s) \rangle_{n|s^*} + \langle \log P(n|s^*) \rangle_{n|s^*} \\ &= -\underbrace{\langle \log P(n|s^*) \rangle_{n|s^*}}_{=0 \text{ (entropy)}} - (s - s^*) \Big\langle \underbrace{\frac{d}{ds} \log P(n|s)\Big|_{s^*}}_{=0 \text{ (see } \dagger, eq.219)} \Big\rangle_{n|s^*} \\ &\quad - \frac{1}{2}(s - s^*)^2 \left\langle \frac{d^2}{ds^2} \log P(n|s)\Big|_{s^*} \right\rangle_{n|s^*} + \cdots + \langle \log P(n|s^*) \rangle_{n|s^*} \\ &= -\frac{1}{2}(s - s^*)^2 \left\langle \frac{d^2}{ds^2} \log P(n|s)\Big|_{s^*} \right\rangle_{n|s^*} + \cdots \\ &= \frac{1}{2}(s - s^*)^2 J(s^*) + \cdots, \end{aligned} \tag{217}$$

where

$$J(s^*) = -\left\langle \frac{d^2}{ds^2} \log P(n|s)\Big|_{s^*} \right\rangle_{n|s^*} \tag{218}$$

is the *Fisher information* (matrix). To explain †, take the explicit expectation:

$$\begin{aligned} \left\langle \nabla_s \log P(n|s)\big|_{s^*} \right\rangle_{n|s^*} &= \int \log \nabla_s P(n|s)\big|_{s^*} P(n|s^*) dn \\ &= \int \frac{\nabla_s P(n|s)}{P(n|s)}\Big|_{s^*} p(n|s^*) dn \\ &= \int \nabla_s P(n|s)\big|_{s^*} = \nabla_s \int P(n|s)\big|_{s^*} dn = \nabla_s 1\big|_{s^*} = 0 \end{aligned} \tag{219}$$

Therefore, in asymptotia, the posterior is

$$P(s|\{n_i\}) \to \frac{1}{Z} e^{-N \mathsf{KL}[P(n|s^*) \| P(n|s)]} \tag{220}$$

$$= \frac{1}{Z} e^{-N \frac{1}{2}(s-s^*)^2 J(s^*)} = \frac{1}{Z'} e^{-\frac{1}{2}(s-s^*)^2 J(s^*)} \tag{221}$$

$$= \mathcal{N}(s^*, J(s^*)^{-1}). \tag{222}$$

Thus, the Fisher information can be seen as measuring the sensitivity of the neural response to changes in the stimulus. Note that this is only a *local* measure of information, as it's contingent on the Taylor expansion around the true stimulus value s^*. Further intuition can be gained by deriving an alternate form of the Fisher

information (here extended to a multidimensional stimulus) using the score trick ($\nabla_s \log P = \frac{1}{P}\nabla_s P$):

$$\begin{aligned}
J(s^*) &= -\left\langle \nabla_s^2 \log P(n|s)\big|_{s^*} \right\rangle_{n|s^*} \\
&= -\left\langle \nabla_s \Big(\frac{1}{P(n|s)} \nabla_s P(n|s)^\top \big|_{s^*} \Big) \big|_{s^*} \right\rangle_{n|s^*} \\
&= \left\langle \frac{1}{P(n|s)^2} \nabla_s P(n|s)\big|_{s^*} \nabla_s P(n|s)^\top \big|_{s^*} \right\rangle_{n|s^*} - \underbrace{\left\langle \frac{1}{P(n|s^*)} \nabla_s^2 P(n|s)^\top \big|_{s^*} \right\rangle_{n|s^*}}_{= \int P(n|s^*) \frac{1}{P(n|s^*)} \nabla_s^2 P(n|s)^\top \big|_{s^*} dn = \nabla_s P(n|s)^\top \big|_{s^*} = 0 \text{ (see †,eq.219)}} \\
&= \left\langle \left(\frac{1}{P(n|s)} \nabla_s P(n|s)\big|_{s^*} \right) \left(\frac{1}{P(n|s)} \nabla_s P(n|s)\big|_{s^*} \right)^\top \right\rangle_{n|s^*} \\
&= \left\langle \left(\nabla_s \log P(n|s)\big|_{s^*} \right) \left(\nabla_s \log P(n|s)\big|_{s^*} \right)^\top \right\rangle_{n|s^*}
\end{aligned} \tag{223}$$

Thus, the Fisher information is the variance of the score function.

The Fisher information is important even outside the large data limit due to the deeper result that is due to Cramér and Rao, which states that for any N, any *unbiased* estimator $\hat{s}(\{n_i\})$ of s will have the property that

$$\mathbb{V}[\hat{s}] = \left\langle (\hat{s}(\{n_i\}) - s^*)^2 \right\rangle_{n_i|s^*} \geq \frac{1}{J(s^*)}. \tag{224}$$

Thus, the Fisher information gives a lower bound on the variance of any unbiased estimator. This is called the *Cramér-Rao bound*. For estimators with bias $b(s^*) = \langle \hat{s}(\{n_i\}) - s^* \rangle$ the bound is

$$\left\langle (\hat{s}(\{n_i\}) - s^*)^2 \right\rangle_{n_i|s^*} \geq \frac{(1 + b'(s^*))^2}{J(s^*)} + b^2(s^*). \tag{225}$$

The Fisher information is a key tool in the analysis of neural population codes.

The Fisher Information and Tuning Curves We model each observed spike count n as $n = r\Delta +$ noise, where $r = f(s)$ (in other words, r is the firing rate given by the tuning curve) and Δ is a count. The Fisher information is then:

$$\begin{aligned}
J(s^*) &= \left\langle \left(\frac{d}{ds} \log P(n|s)\Big|_{s^*} \right)^2 \right\rangle_{n|s^*} \\
&= \left\langle \left(\frac{d}{dr\Delta} \log P(n|r\Delta)\Delta f'(s^*)\Big|_{f(s^*)} \right)^2 \right\rangle_{n|s^*} \\
&= J_{\text{noise}}(r\Delta)\Delta^2 f'(s^*)^2.
\end{aligned} \tag{226}$$

Note that we use the chain rule, so $\frac{d}{ds} r\big|_{s^*} = \frac{d}{ds}\Delta f(s)\big|_{s^*}$. In this case, we can see that the Fisher information will be non-negative, its value changing according to the squared derivative of the tuning curve $f(s)$ (Fig. 11.2). The Fisher information goes to zero whenever the firing rate hits a local maximum or minimum.

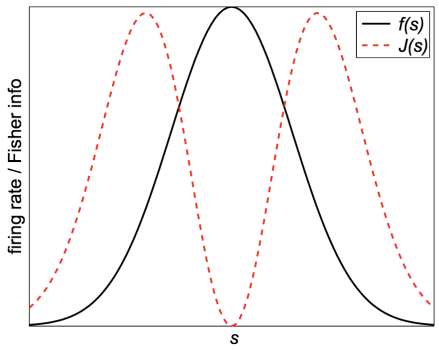


Figure 11.2: Relationship between the Fisher information of a stimulus $J(s)$ and the associated tuning curve $f(s)$. Maximal information is gained on the slopes.

Poisson Neurons For Poisson neurons, the conditional spike distribution is given by

$$P(n|r\Delta) = \frac{e^{-r\Delta}}{n!}(r\Delta)^n, \tag{227}$$

so

$$\begin{aligned}
J_{\text{noise}}(r\Delta) &= \left\langle \left(\frac{d}{dr\Delta} \log P(n|r\Delta)\Big|_{r^*\Delta} \right)^2 \right\rangle_{n|s^*} \\
&= \left\langle \left(\frac{d}{dr\Delta} - r\Delta + n\log(r\Delta) - \log n!\Big|_{r^*\Delta} \right)^2 \right\rangle_{n|s^*} \\
&= \left\langle \left(-1 + \frac{n}{r^*\Delta} \right)^2 \right\rangle_{n|s^*} \\
&= \left\langle \frac{\overbrace{(n-r^*\Delta)^2}^{\langle (n-r^*\Delta)^2\rangle := \mathbb{V}[n|r\Delta]=r^*\Delta}}{(r^*\Delta)^2} \right\rangle_{n|s^*} \\
&= \frac{r^*\Delta}{(r^*\Delta)^2} \\
&= \frac{1}{r^*\Delta}.
\end{aligned} \tag{228}$$

Therefore a simple estimator can achieve the Cramér-Rao lower bound! Note that this isn't too surprising, as the optimal estimator for the mean $r^*\Delta$ is $\widehat{r^*\Delta} = n$ and $\mathbb{V}[n] = r^*\Delta$. The complete stimulus Fisher information, using eq. 226 and $r^* := f(s^*)$, is then

$$\begin{aligned}
J(s^*) = J_{\text{noise}}(r\Delta)\Delta^2 f'(s^*)^2 &= \frac{1}{r^*\Delta}\Delta^2 f'(s^*)^2 \\
&= \frac{f'(s^*)^2}{f(s^*)}\Delta.
\end{aligned} \tag{229}$$

Population Fisher Information and independent noise The FI for independent random variables is additive:

$$\begin{aligned}
J_n(s) &= \left\langle -\frac{\partial^2}{\partial s^2} \log P(\mathbf{n}|s) \right\rangle \\
&= \left\langle -\frac{\partial^2}{\partial s^2} \sum_a \log P(n_a|s) \right\rangle \\
&= \sum_a \left\langle -\frac{\partial^2}{\partial s^2} \log P(n_a|s) \right\rangle \\
&= \sum_a J_{n_a}(s).
\end{aligned} \tag{230}$$

For cells obeying Poisson statistics, $\sum_a J_{n_a}(s) = \Delta \sum_a \frac{f_a'(s)^2}{f_a(s)}$.

Population Fisher Information and correlated (Gaussian) noise The FI when $\mathbf{r} \sim \mathcal{N}(\mu(s), \Sigma(s))$ can be derived:

$$
\begin{aligned}
J(\theta) &= -\left\langle \nabla_s^2 \log p(\mathbf{r}|s)\big|_{s^*} \right\rangle_{n|s^*} \\
&= -\frac{1}{2}\Big\langle \nabla_s^2 \log|\Sigma^{-1}|\big|_{s^*} - \nabla_s^2 \operatorname{Tr}\left[\Sigma^{-1}(\mathbf{r}-\mu)(\mathbf{r}-\mu)^\top\right]\big|_{s^*} \Big\rangle_{n|s^*} \\
&= -\frac{1}{2}\Big\langle \nabla_s \operatorname{Tr}\left[\Sigma(\Sigma^{-1})'\right]\big|_{s^*} - \nabla_s \operatorname{Tr}\left[(\Sigma^{-1})'(\mathbf{r}-\mu)(\mathbf{r}-\mu)^\top\right]\big|_{s^*} + 2\nabla_s \operatorname{Tr}\left[(\Sigma^{-1}\mu'(\mathbf{r}-\mu)\right]\big|_{s^*} \Big\rangle_{n|s^*} \\
&= -\frac{1}{2}\Big\langle \operatorname{Tr}\left[\Sigma(\Sigma^{-1})''\right] + \operatorname{Tr}\left[\Sigma'(\Sigma^{-1})'\right] - \operatorname{Tr}\left[(\Sigma^{-1})''\underbrace{(\mathbf{r}-\mu)(\mathbf{r}-\mu)^\top}_{\Sigma \text{ in expectation}}\right] + 4\operatorname{Tr}\left[(\Sigma^{-1})'\mu'\underbrace{(\mathbf{r}-\mu)}_{0 \text{ in expectation}}\right] \\
&\quad + 2\operatorname{Tr}\left[(\Sigma^{-1})'\mu''\underbrace{(\mathbf{r}-\mu)}_{0 \text{ in expectation}}\right] + 2\operatorname{Tr}\left[\Sigma^{-1}\mu'\mu'^\top\right] \Big\rangle_{n|s^*} \\
&= -\frac{1}{2}\Big(\operatorname{Tr}\left[\Sigma' \underbrace{(\Sigma^{-1})'}_{=-\Sigma^{-1}\Sigma'\Sigma^{-1}}\right] + \underbrace{\operatorname{Tr}\left[\Sigma(\Sigma^{-1})''\right] - \operatorname{Tr}\left[(\Sigma^{-1})''\Sigma\right]}_{=0 \text{ (cyclic trace property)}} + 2\operatorname{Tr}\left[\Sigma^{-1}\mu'\mu'^\top\right] \Big) \\
&= \underbrace{\frac{1}{2}\operatorname{Tr}\left[\Sigma'\Sigma^{-1}\Sigma'\Sigma^{-1}\right]}_{\text{“the trace term”}} - \underbrace{\mu'^\top\Sigma^{-1}\mu'}_{\text{gets most focus}} .
\end{aligned}
\tag{231}
$$

This is sometimes called “linear” Fisher Information based on the performance of a locally optimal linear decoder. The second term gets more focus in the literature, probably because it implies that high FI is achieved when the mean is arranged orthogonally. In contrast, consider the eigendecomposition $\Sigma = \sum_i \lambda_i \mathbf{v}_i \mathbf{v}_i^\top$. Then we can compute $J_1(s) = \sum_i \lambda_i^{-1}(\mathbf{v}_i^\top \mu')^2$, i.e. a large eigenvalue along μ' will decrease FI. Therefore, this latter scenario is sometimes called a differential or information-limiting correlation.

11.1.4 Optimal Tuning Curve Widths

Consider a population of neurons $a = 1, \ldots, N$ coding for a multidimensional stimulus $s \in \mathbb{R}^D$, with homogeneous tuning curves given by

$$
f_a(s) = r_{max}\phi\left(\frac{\sum_{d=1}^D (s_d - c_d^a)^2}{\sigma^2}\right) = r_{max}\phi\left(\frac{\xi^a}{\sigma^2}\right), \tag{232}
$$

where c_d^a are the uniformly distributed tuning curve centers and $\phi(\cdot)$ is a monotonically decreasing function (e.g., $\phi(x) = e^{-x}$ for a Gaussian tuning curve). Note that because the tuning curve width is constant in each dimension, the tuning curves are circularly symmetric. We also assume that the tuning curves are independent, such that $P(r|s) = \prod_a P(r_a|f_a(s))$. Then we can see quickly that the Fisher information for the population is given by the sum of the Fisher information for each neuron as per eq. 230: $\sum_a J_a(s)$. The question we'd like to ask, then, is what tuning curve width σ maximizes the population Fisher information?

To figure this out, we first find the Fisher information for a single neuron a, this time using the covariance form (eq. 223):

$$
J_{ij}^a(s) = \left\langle \left(\frac{\partial}{\partial s_i}\log P(r_a|f_a(s))\right)\left(\frac{\partial}{\partial s_i}\log P(r_a|f_a(s))\right)^\intercal \right\rangle. \tag{233}
$$

The derivative is

$$
\begin{aligned}
\frac{\partial}{\partial s_i}\log P(r_a|f_a(s)) &= \frac{1}{P(r_a|s)}\frac{\partial P(r_a|f_a(s))}{\partial s_i} && (234)\\
&= \frac{1}{P(r_a|s)}\frac{\partial P(r_a|f_a(s))}{\partial f_a(s)}\frac{\partial f_a(s)}{\partial \phi(\xi^a/\sigma^2)}\frac{\partial \phi(\xi^a/\sigma^2)}{\partial \xi^a}\frac{\partial \xi^a}{\partial s_i} && (235)\\
&= \frac{1}{P(r_a|s)}\frac{\partial P(r_a|f_a(s))}{\partial f_a(s)} r_{max}\frac{1}{\sigma^2}\phi'(\xi^a/\sigma^2)2(s_i - c_i^a), && (236)
\end{aligned}
$$

and so we can write

$$
J_{ij}^a(s) = K_a(\xi_a)\frac{(s_i - c_i^a)(s_j - c_j^a)}{\sigma^4}, \tag{237}
$$

where

$$
K_a(\xi_a) = \left\langle 4\left(\frac{1}{P(r_a|s)}\frac{\partial P(r_a|f_a(s))}{\partial f_a(s)} r_{max}\phi'(\xi^a/\sigma^2)\right)^2 \right\rangle \tag{238}
$$

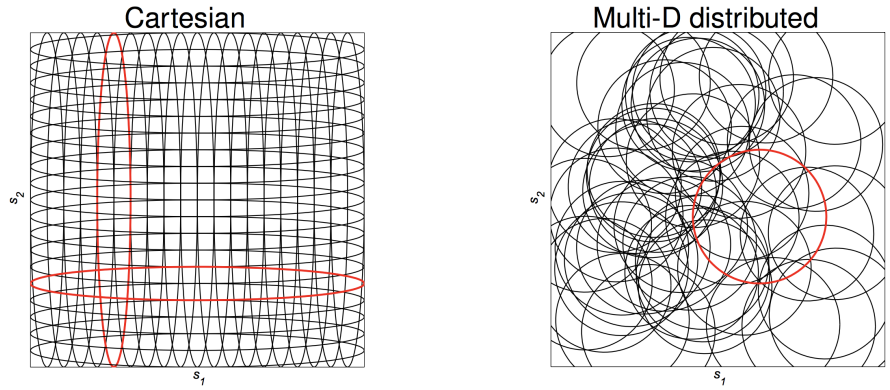


Figure 11.3: Examples of coding frameworks. (Left) A Cartesian code. This is efficient in the number of neurons required, but has difficulty encoding multiple values. (Left) A distributed code. This style requires a high number of neurons, but is more effective at representing multiple stimulus values.

is the only term where the expectation appears since it is the only term dependent on the activity r_a over which the expectation is defined. If we define

$$\xi_i^a := \frac{s_i - c_i}{\sigma}, \tag{239}$$

such that $\xi^a = \sigma^2 \sum_i (\xi_i^a)^2$, we can see that since $\phi(\xi^a/\sigma^2)$ is a monotonically decreasing function of $(\xi_i^a)^2$, it is symmetric around $\xi_i^a = 0$. Thus, $\phi'(\xi^a/\sigma^2)^2$ and $f_a(s)$ are as well, so $K_a(\xi_a)$ is also symmetric around $\xi_i^a = 0$.

Because we assumed that the tuning curve centers are uniformly distributed about the stimulus space, we can say $P(c_i^a) = p_c$. Taking the limit $N \to \infty$ to approximate sums with integrals, we can then write

$$J_{ij}(s) = \sum_a J_{ij}^a(s) \tag{240}$$

$$= \int \cdots \int p_c J_{ij}^a \, dc_1^a \, dc_2^a \ldots dc_D^a \tag{241}$$

$$= \int \cdots \int p_c K_a(\xi_a) \frac{(s_i - c_i^a)(s_j - c_j^a)}{\sigma^4} \, dc_1^a \, dc_2^a \ldots dc_D^a. \tag{242}$$

To evaluate this integral, we can exploit the fact that $K_a(\xi_a)$ is symmetric around $\xi_i^a = 0$ by performing a change of variables $c_i^a \to \xi_i^a$, such that $dc_i^a = -\sigma \, d\xi_i^a$:

$$J_{ij}(s) \approx \frac{1}{\sigma^2} \int \cdots \int p_c K_a(\xi_a) \xi_i^a \xi_j^a \, \sigma d\xi_1^a \, \sigma d\xi_2^a \ldots \sigma d\xi_D^a \tag{243}$$

$$= \frac{\sigma^D}{\sigma^2} \int \cdots \int p_c K_a(\xi_a) \xi_i^a \xi_j^a \, d\xi_1^a \, d\xi_2^a \ldots d\xi_D^a. \tag{244}$$

Because $K_a(\xi^a)$ is symmetric around $\xi_i^a = 0$, we have $\int_{-\infty}^{\infty} p_c K_a(\xi^a) \xi_i^a = 0$, so when $i \neq j$,

$$J_{ij}(s) \approx \frac{\sigma^D}{\sigma^2} \int \cdots \xi_j^a \int p_c K_a(\xi_a) \xi_i^a \, d\xi_i^a \, d\xi_{i+1}^a \ldots d\xi_D^a = 0. \tag{245}$$

However, when $i = j$, we get

$$J_{ii}(s) \approx \frac{\sigma^D}{\sigma^2} \int \cdots \int p_c K_a(\xi_a) (\xi_i^a)^2 \, d\xi_1^a \, d\xi_2^a \ldots d\xi_D^a = \sigma^{D-2} \eta A, \tag{246}$$

where A is independent of σ.

We can therefore see that the total Fisher information in the population is proportional to σ^{D-2}, where D is the dimensionality of the stimulus. This yields some interesting insights about the optimal tuning curve width σ:

- If $D = 1$, the tuning curves should want to be as narrow as possible, down to the smallest resolution between neighboring c_i^a ($\sigma \to 0$ to maximise FI).
- If $D = 2$, the Fisher information is independent of tuning curve widths.
- If $D > 2$, the wider the tuning curve the better—optimality is achieved when the tuning curve spans the full stimulus space, so the actual limit is set by the range of stimuli.

This analysis was derived by [Zhang and Sejnowski, 1999], but the write-up is pretty much verbatim from [Menendez, 2018]. It also turns out that if the tuning curve widths are allowed to vary between stimulus dimensions (i.e., they are no longer circularly symmetric), then maximizing the Fisher information gives a Cartesian code in which the optimal tuning curve width is narrow in some dimensions and wide in others (Figure 11.3, left). There is an inherent trade off in population codes, since $f'(s) \propto \sigma^{-1}$ (slope) while $N(s) \propto \sigma^{D}$: we cannot account for coding multiple values.

11.2 Overview of Latent Variable Approaches

To make sense of population dynamics, we can also turn to latent variable methods, assuming that while the activity space is high-dimensional, only a handful of underlying states are truly meaningful. In this framework, coordinates on the latent manifold of possible activity combinations effectively become latent variables—if the meaningful content is truly low-dimensional, so are the dynamics (e.g., they exist as a path on the manifold; Figure 11.4).

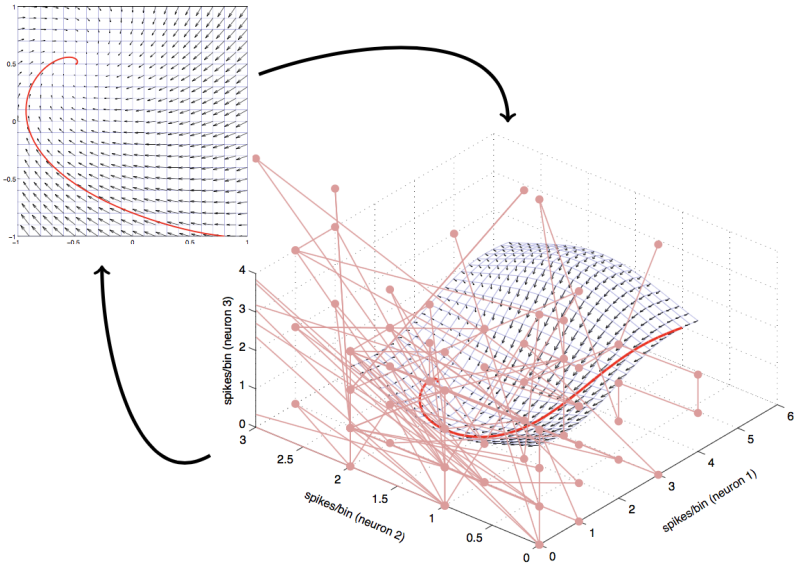


Figure 11.4: Example of a 3D (aka, three neuron) neural activity space with low(er) dynamics. The top matrix is the stimulus, which is encoded onto the neural manifold through the top arrow with noisy observations; the bottom arrow is therefore decoding.

Under this assumption, there are three families of approaches we can consider: (i) static dimensionality reduction, which requires that the dominant force behind neural variability is not noise but rather computational variability within the manifold, (ii) low-dimensional latent dynamics, which assumes that noise may lift data off the manifold, but only its projection onto the manifold can influence the future evolution of the system, and (iii) supervised modeling.

11.2.1 Static Dimensionality Reduction

Under this framework, there are a number of methods we can use. These include linear Gaussian methods, such as (P)PCA and factor analysis (FA). It's important to remember that PPCA and PCA are invariant to rotations of the input, while FA is not, and FA is invariant to measurement scale, while PCA and PPCA are not. Therefore, (P)PCA are generally preferred when we expected rotational symmetry, i.e., when there's nothing special about one axis of the input space compared to another. However, this doesn't turn out to be the case in neural population analysis—rather, invariance to scale results in a more accurate model, and FA is generally preferred (Figure 11.5).

The assumptions of Gaussian noise and mean-independent, stationary variance are unrealistic for real spike counts, particularly in small bins. Square-rooting improves things, but is inaccurate for small counts and transforms the shape of the manifold. Instead, one can use a conditionally Poisson count distribution via (i) Poisson factor analysis (PFA), (ii) exponential family PCA, or (iii) a covariance transformation. Options (i) and (ii) follow the frameworks of FA and PCA, respectively, but with non-Gaussian noise assumptions—typically Poisson. Exponential family PCA also typically includes a regularization factor to limit the rank of solutions. We can also use methods like canonical correlations analysis (CCA).

Dynamics There are a number of dynamics-based approaches to modeling population coding. Examples include slow feature analysis (SFA), Gaussian processes (GPs) and Gaussian process factor analysis (GPFA),

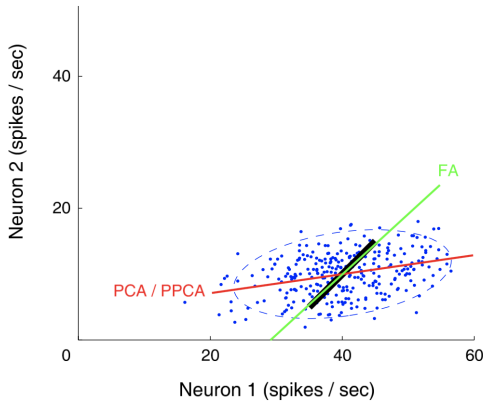


Figure 11.5: The scale-invariance of FA typically better captures the latent variables in neural data compared to PCA. Here, the black line represents the underlying mean. Note that the PCA estimate is pulled along the axis of greatest variation—it is not scale-invariant.

linear-Gaussian state-space models (LGSSMs, such as Kalman filters), linear dynamical systems (LDS; related to the Kalman filter), Poisson noise LDS (here, EM is intractable—requires approximation), and recurrent linear models.

Supervised Approaches In this family of approaches, models try to predict known experimental factors or covariates (e.g., movement speed or stimulus identity) from multivariate data. The methods considered are also used to study structure in in the condition averages—this is equivalent to having one trial per condition. Averaging might make the noise more Gaussian, but still without equal variance.

To predict categorical factors, such as stimulus identities or behavioral instructions, methods like multifactor decomposition of variance are essential. More generally, we can also use, for instance, linear discriminant analysis (LDA), demixed PCA (DPCA), or Wiener filtering.

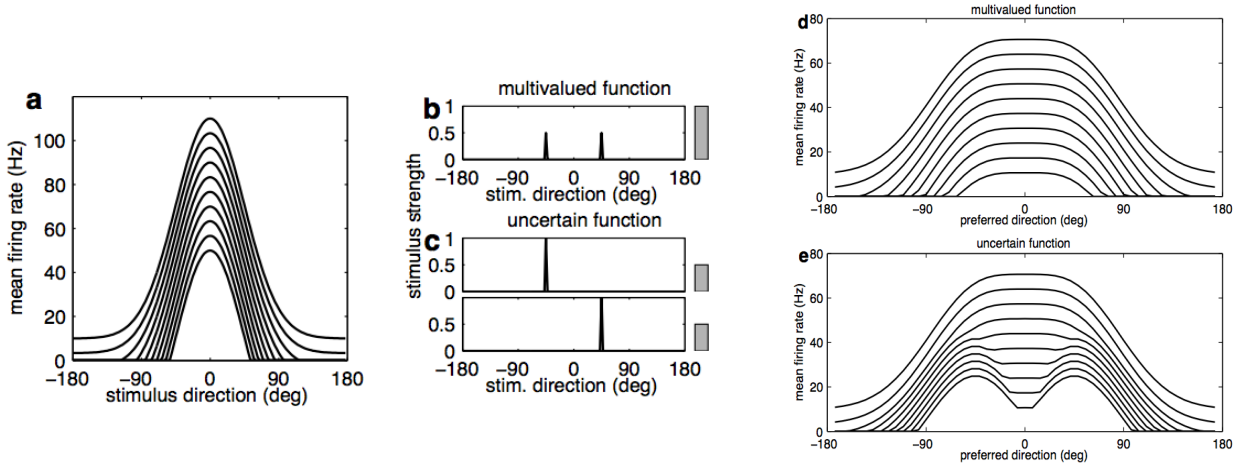


Figure 12.1: Encoding multiplicity and uncertainty with a DDPC. Panel (a) shows the tuning curves for 10 neurons with a preferred stimulus orientation of 0°. Panels (b) and (c) show two types of input stimuli: (b) is a multi-valued function, and (c) contains a function whose value is uncertain. However, it is important to note that both functions have an expected value of 0°. Panels (d) and (e) show the resulting DDPC representations of the multi-valued and uncertain functions, respectively. We can see that despite having identical expectations, the DDPC is able to differentiate between the two. Figure from [Sahani and Dayan, 2003].

12 Doubly Distributional Population Codes (Dayan & Sahani, 2003)

The authors present a new model for population coding that, unlike previous approaches, is able to account for the presence of multiple stimuli and representational uncertainty [Sahani and Dayan, 2003]. The authors define the standard model of population coding as follows: the firing rate of the ith neuron r_i is distributed around the mean rate $f_i(s)$ for a given value of a stimulus s, where the plot of f_i for different values of s defines the neuron's tuning curve. This is written as

$$r_i \sim f_i(s). \tag{247}$$

The population firing rate $\mathbf{r} = \{r_i\}$ defines the representation of the stimulus, and the populations that respond to different types of stimuli can be broadly overlapping, thus adding robustness to noise and cell death. They show that standard population coding models of neural activity fail to account for two common scenarios that occur in stimulus presentation: *multiplicity* and *uncertainty*. Multiplicity is when multiple values of s need to be represented simultaneously (e.g., different sounds are produced by spatially separate sources). Uncertainty arises through two possible sources: noise and ill-posedness in perceptual inference, an issue arising from the fact that cortical representations must be themselves be computed from the uncertain outputs of sensory neurons, not directly from the value of s (which is essentially a latent variable). This results in the possibility that two different values of s could produce the same intermediate representation via sensory neurons–in other words, two different stimuli could result in the same input to cortical neurons.

A generalization of the standard model is the *distributional population code* (DPC), which views the population activity as being a *function* $m(s)$ over the stimulus, rather than encoding only a single value. This is an exact generalization because if $m(s)$ is a delta function on s, then it recovers the old model. Using the same notation as above, the ith firing rate is given by

$$r_i \sim \sigma_i \left(\int f_i(s) m(s)\, ds \right), \tag{248}$$

where $\sigma_i(\cdot)$ is a fixed nonlinearity. When $m(s)$ is a delta function, $r_i \sim \sigma_i(f_i(s))$, equivalent to the original model. When that's not the case, the population of firing rates $\mathbf{r}$ is sufficient to decode an approximation to $m(\cdot)$. However, a crucial shortcoming is that DPCs can encode either multiplicity *or* uncertainty, but it cannot distinguish between the two, as they are both represented the same way, nor can it represent both at the same time.

To meet this need, the authors introduce the *doubly distributional population code* (DDPC), which encodes uncertainty about the functions $m(s)$. In a DDPC, the population activity encodes a probability distribution $p[m]$ over functions $m(s)$, with $m(s)$ capturing potential multiplicity and $p[m]$ capturing uncertainty. As an example, if multiple moving stimuli are presented, $m(s)$ would represent the activation strengths of different sensory receptors. Then a perfectly certain $m(\cdot)$ would be encoded by neuron i with linear filter function $f_i(s)$

and nonlinearity $\sigma_i(\cdot)$, producing mean

$$\phi_i[m] = \sigma_i\left(\int f_i(s)m(s)\,ds\right). \tag{249}$$

However, issues such as ill-posedness also require that uncertainty about $m(s)$ be represented via a distribution $p[m]$, which can be learned through experience. The mean activity of the ith neuron is then the average over $p[m]$ of its DPC activity:

$$r_i(p[m]) = \langle \phi_i[m] \rangle_{p[m]} = \left\langle \sigma_i\left(\int f_i(s)m(s)\,ds\right)\right\rangle_{p[m]}. \tag{250}$$

This representation of $p[m]$ in firing rates is the *doubly distributional population code* (DDPC). The authors then demonstrate empirically that the DDPC is indeed able to differentiate between cases of multiplicity and uncertainty (Figure 12.1), and then through a MAP framework, that this difference is sufficient to *decode* the activity to an accurate representation of the original stimulus.

13 Deep (and Other) Learning

Probably, different forms of learning rules are used in different parts of the brain: consider examples of tonotopic tuning in auditory cortex, perceptual learning (orientation discrimination), distributed plasticity in V1, human semantic development and illusory correlations, and category coherence. At the synapse level, Hebbian learning lasts only a short time (about 4 hours) and for various reasons is unlikely to really be what (all of) the brain does; one should also consider the role of third factors, like dopamine in STDP. Deep learning has become a popular method of supervised learning. Hopefully, it may be applicable to understanding information processing in neuroscience, too, especially with the popular ideas of using the whole brain and "everything being everywhere." An obvious example is the hierarchy and serial propagation of information that occurs in the visual system. This section aims to cover, for the most part, this otherwise neglected topic.

13.1 Classical Learning

Classical learning is still relevant, and employed in ocular dominance, LTP/LTD, and the predicted statistics of receptive fields. However, this is largely covered in other sections of these notes, see .

13.1.1 Hebb

Assume post synaptic neurons v are governed by presynaptic firing rates u with weights w:

$$\tau \frac{dV}{dt} = -v + w^\top u \tag{251}$$

$$\tau_w \frac{dw}{dt} = vu \tag{252}$$

On average,

$$\tau_w \left\langle \frac{dw}{dt} \right\rangle = \langle uv \rangle = \langle uu^\top \rangle w = Qw,$$

where Q is the correlation matrix. For this reason, Hebbian learning is also known as correlational learning (with the caveat that this occurs when changes in weight are driven be *steady state values*, a key requirement for the derivation above).

Hebbian learning has two main problems:

- w is unbounded.
- There is no competition – the correlations become redundant.

To overcome these, one might add a threshold to equation 252:

$$\tau_w \frac{dw}{dt} = (v - \theta_v)u \tag{253}$$

Letting $\theta_v = \langle v \rangle$,

$$\tau_w \left\langle \frac{dw}{dt} \right\rangle = \langle u(v - \langle v \rangle v) \rangle = \langle u(u^\top w - \langle u^\top \rangle w \rangle = (\langle uu^\top \rangle - \langle u \rangle \langle u^\top \rangle) w = Cw,$$

where C is the covariance matrix. When this algorithm is run over time, the dynamics should converge and can be solved as an eigenvector problem, $\tau_w \dot{w} = \mathbf{C}\mathbf{w}$. Let $w(t) = \sum_i c_i(t) e_i$, i.e. $c_i(t) = w(t)^\top e_i$. That is, rewrite the problem as a rotation in eigenspace. With some work and solving a linear DE, this should get you to the solution:

$$w(t) = \sum_i c_i(0) e^{\lambda_i t / \tau_w}. \tag{254}$$

Think of the exponent in terms of the covariance matrix, and you should notice that the component with largest eigenvalue grows exponentially larger than the others in the time limit. Therefore, $\lim_{t \to \infty} w \propto e_1$ and $v \propto e_1^\top u$.

13.1.2 BCM

The BCM rule is an improvement on Hebb in that it fixes the boundedness problem by adding a sliding threshold to equation 253.

Here,

$$\tau_w \frac{dw}{dt} = vu(1-\theta_v) = v^2u(1-v) \tag{255}$$

$$\tau_\theta \frac{d\theta_v}{dt} = v^2 - \theta_v, \tag{256}$$

so that when $v = 1$, weight change stops, and when $v = 0$, the fixed point is unstable. From eq. 253, $\theta_v = v^2$ on average at steady state. Therefore, the threshold increases supralinearly, and it is more difficult to get LTP as it activates. To fix this, one might use the ouput neuron to scale this quantity.

13.1.3 Oja

Oja's rule removes the sliding threshold:

$$\tau_w \frac{dw}{dt} = vu - \underbrace{\alpha}_{\alpha>0} v^2 w. \tag{257}$$

This can be interpreted in terms of the change in the norm of the weights,

$$\frac{d\left\|w\right\|^2}{dt} = 2w^\top(vu - \alpha v^2 w) = 2v^2(1 - \alpha\left\|w\right\|^2),$$

so that the fixed point is given by $\alpha = \frac{1}{\|w\|^2}$.

13.2 Supervised Learning – Perceptrons

Andrew argues that supervised learning is relevant for the brain, as opposed to only relying on unsupervised learning. Correlations are not everything (for example, learning environmental features through instruction seems useful), and supervised and unsupervised learning can always (and probably are) combined.

Perceptrons are a common simple example of supervised learning that you will no doubt have encountered by now (although apparently there may be some applications in the cerebellum?). Imagine some nodes x_i converging with weights w on the output $y = \text{sign}(w^\top x)$ – that is, a simple classifier. We want to learn a model that does this, or a decision boundary that allows separation of classes. Note that $w^\top x = \|w\|\,\|x\|\cos(\theta)$, so we can think of the inputs x as being separated orthogonally via the decision boundary.

It is well-known that perceptrons cannot solve the XOR problem. But what can they do? That is, given $\{x^1, x^2, ..., x^P\} \in \mathbb{R}^N$, how many dichotomies, or choice of positive and negative points, can it achieve? For this we consider Cover's Theorem. Later, we will look at how to learn and find these dichotomies, employing the Perceptron Convergence Theorem, which really just says when these dichotomies can exist.

13.2.1 Cover's Theorem

13.2.2 Perceptron Convergence Theorem

13.3 Deep Linear Networks

Deep linear networks are layers of neurons, connected between layers. Think "divide and conquer!" They comprise:

- Architecture
- Initiation
- Loss or objective function
- Learning rules, for example gradient descent
- Dataset or environment (distribution of inputs).

More specifically, we have:

1. Inputs: $x \in \mathbb{R}^{N_L}$
2. Weights: $w_i \in \mathbb{R}^{N_L \text{x} N_L}$
3. Hidden layers: $h_L \in \mathbb{R}^{N_L}$ such that $h_i = f(w_i h_i) \forall i, h_0 = x$.

13.3.1 Backpropagation

This is basically just chain rule.

13.3.2 Deep Linear Networks

Consider at first a two-layer network with $\hat{y} = w_2 w_1 x$ and let $L(\{w\}) = \left\langle \frac{1}{2} \left\| y - \hat{y} \right\|^2 \right\rangle$, the standard L_2 loss.

Then:

$$\Delta w_1 = \langle \delta_1 x^\top \rangle = \langle w_2^\top (y - \hat{y}) x^\top \rangle = w_2^\top (\langle y x^\top \rangle - w_2 w_1 \langle x x^\top \rangle) \tag{258}$$

$$\Delta w_2 = \langle \delta_2 h^\top \rangle = \langle (y - \hat{y}) x' w_1^\top \rangle = (\underbrace{\langle y x^\top \rangle}_{\Sigma^{yx}} - w_2 w_1 \underbrace{\langle x x^\top \rangle}_{\Sigma^x}) w_1^\top \tag{259}$$

This is a pleasingly symmetric answer, as the equations are neatly coupled. Now let's consider these weight trajectories over time. To do so, do singular value decomposition to write $\Sigma^{yx} = USV^\top$, and let $\Sigma^X = I$. Then, $R^\top R = I$, so

$$w_1 = R w_1 V^\top$$
$$w_2 = U w_2 R^\top.$$

Further,

$$\tau \frac{d}{dt}(R w_1 V^\top) = R w_2^\top U^\top (\Sigma^{yx} - U w_2 R^\top R w_1 V^\top \Sigma^x) \tag{260}$$

$$\iff \tau \frac{d}{dt} w_1 = w_2^\top (U^\top \Sigma^{xy} - U^\top U w_2 w_1 V^\top V) = w_2^\top (S - w_2 w_1), \tag{261}$$

$$\text{and } \tau \frac{d}{dt} w_2 = (S - w_2 w_1) w_1^\top. \tag{262}$$

Assume that at initialisation, $w_1(0)$ and $w_2(0)$ are diagonal. Let $c_\alpha = (w_1)_{\alpha\alpha}$ and $d_\alpha = (w_2)_{\alpha\alpha}$. Then:

$$\tau \frac{d}{dt} c_\alpha = d_\alpha (S_\alpha - c \alpha d_\alpha) \tag{263}$$

$$\tau \frac{d}{dt} d_\alpha = (S_\alpha - d \alpha c \alpha) c_\alpha. \tag{264}$$

We can then consider the phase portrait describing the dynamics of c_α, d_α. Fixed points occur when $S_\alpha = c_\alpha d_\alpha$ (minima), and $c_\alpha = d_\alpha = 0$ (saddle point – so if we add noise, trajectories dispel). Because the nullclines are hyperbolic, $c_\alpha^2 - d_\alpha^2$ is a constant and fixed by initialisation.[12] Note that the weakest link will change the most over time (this is a general feature of deep linear networks). Assuming the ws are diagonal, the system is decoupled, and balanced.

Now, let $c_\alpha = d_\alpha, a_\alpha = c_\alpha d_\alpha$. Then we can get a separable DE,

$$\tau \frac{d}{dt} a_\alpha = c_\alpha (\dot{d_\alpha}) + a_\alpha (\dot{c_\alpha}) = \tau (c_\alpha^2 + d_\alpha^2)(S - c_\alpha d_\alpha) \tag{265}$$

$$= 2 a_\alpha (s - a_\alpha), \tag{266}$$

which we can solve,

$$t = \frac{\tau}{2S} \log \frac{a_\alpha(t)(S - a_\alpha(0))}{a_\alpha(0)(S - a_\alpha(t))} \tag{267}$$

$$\iff a_\alpha(t) = \frac{S_\alpha e^{2 S_\alpha t/\tau}}{e^{2 S_\alpha t/\tau} - 1 + \frac{s_\alpha}{a_\alpha(0)}}, \tag{268}$$

and so express the dynamics of each singular value.

We can apply these concepts to deeper networks and hierarchical semantic structure selection (that is, picking apart items and features to get independent scalar chains). This gives us rich information about the error landscape with depth (sigmoidal), with learning proportional to the inverse of the singular values.

13.3.3 Semantic cognition

13.4 Student-teacher Formalism

13.5 Neural Tangent Kernel

[12] Standard approach is to start at small weights, i.e. near the saddle.

A Important Constants In Neuroscience

Symbol	Value	Name	Notes
-	10^{11}	number of neurons	in human brain
K	1000	avg connections per neuron	in cortex? (PEL)
V_{rest}	-70mV	resting membrane potential	
V_{th}	-50mV	spiking threshold	in reality not fixed
E_{Na}	50mV	Na^+ reversal potential	
E_K	$-90-70$mV	K^+ reversal potential	
E_{Cl}	$-65-60$mV	Cl^- reversal potential	
E_{Ca}	150mV	Ca^2+ reversal potential	
τ_m	$10-100$ms	membrane time constant	$\tau_m = c_m r_m = C_m R_m$, independent of membrane surface area
r_m	$1\text{M}\Omega\text{mm}^2$	specific membrane resistance	membrane resistance of a neuron with surface area A given by $R_m = \frac{r_m}{A}$, r_m varies with V
c_m	$10\text{nF}/\text{mm}^2$	specific membrane capacitance	membrane capacitance of a neuron with surface area A given by $C_m = c_m A$
A	$.01\text{mm}^2$	neuronal surface area	
r_L	$1\text{k}\Omega\text{mm}$	intracellular resistivity	a property of cell cytoplasm; longitudinal resistance in a neurite of length L and cross-sectional radius a is given by $R_L = \frac{L \times r_L}{\pi a^2}$
a	2μm	cross-sectional radius of a dendrite	velocity of signal propagation in axon, dendrite scales with $a, \sqrt{a}$
λ	1mm	electrotonic length of a dendrite	$\lambda = \sqrt{\frac{r_m a}{2 r_L}}$, sets the scale of spatial decay of a constant current injection (for infinite length dendrite) $\rightarrow$ dendrites can't be much longer than this
R_λ	$\sim\text{M}\Omega$	input resistance	$R_\lambda = \frac{r_m}{\lambda 2\pi a}$, the ratio of equilibrium potential to injected current (for constant current injection in infinite cable)
g_i^{open}	25pS	open channel conductance	
-	1mV	EPSP	
-	1ms	PSP rise time constant	in CA3 pyramidal cell
-	5ms	PSP decay time constant	in CA3 pyramidal cell

B Electrical Circuits

Name	Symbol	Units
Charge	Q	$C = 6.2 \times 10^{18}$ electrons (coulombs)
Current	I	$A = C/s$ (amps = coulombs per second)
Voltage	V	$V = J/C$ (volts = potential energy joules per coulomb)
Resistance	R	$\Omega = V/A$ (ohms = volts per amp)
Capacitance	C	$F = C/V$ (farad = coulomb per volt)

We therefore have:

$$I = \frac{\mathrm{d}Q}{\mathrm{d}t}$$

$$CV = Q \Leftrightarrow C\frac{\mathrm{d}V}{\mathrm{d}t} = I$$

$$R = \frac{I}{V} \Leftrightarrow \Delta V = IR \quad \text{(Ohm's Law)}$$

We can interpret these latter two equations as telling us that

- capacitance C determines how much current I is needed to change the voltage *at a given rate* $\frac{\mathrm{d}V}{\mathrm{d}t}$
- resistance R determines how much current I is needed to change the volltage *by a given amount* ΔV

C Solving Differenetial Equations

C.1 First-Order ODEs: Method of Integrating Factors

Consider a differential equation of the form:

$$\frac{\mathrm{d}y}{\mathrm{d}x} + p(x)y(x) = g(x)$$

Because of the $y(x)$ term isolated from the derivative on the left-hand side, we can't solve this by straightforward integration. We can deal with this, however, via the product rule by multiplying both sides by the *integrating factor*

$$v(x) = \int p(x)\mathrm{d}x \Leftrightarrow \frac{\mathrm{d}v}{\mathrm{d}x} = p(x)$$

which gives us:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x}y(x)e^{v(x)} &= \frac{\mathrm{d}y}{\mathrm{d}x}e^{v(x)} + \frac{\mathrm{d}v}{\mathrm{d}x}y(x)e^{v(x)} \\ &= \left(\frac{\mathrm{d}y}{\mathrm{d}x} + p(x)y(x)\right)e^{v(x)} \\ &= g(x)e^{v(x)}\end{aligned}$$

such that we can now simply integrate both sides to get our solution:

$$\begin{aligned}\int \frac{\mathrm{d}}{\mathrm{d}x}y(x)e^{v(x)}\mathrm{d}x &= \int g(x)e^{v(x)}\mathrm{d}x \\ \Leftrightarrow y(x) &= e^{-v(x)}\int g(x)e^{v(x)}\mathrm{d}x\end{aligned}$$

C.2 Homogenous Second-Order ODEs

Consider a differential equation of the form

$$\frac{\mathrm{d}^2y}{\mathrm{d}x^2} + p\frac{\mathrm{d}y}{\mathrm{d}x} + qy(x) = 0$$

with p, q constant coefficients. Let $y'' = \frac{\mathrm{d}^2y}{\mathrm{d}x^2}, y' = \frac{\mathrm{d}y}{\mathrm{d}x}$. We now note that if we can find a pair a, b such that $p = -(a+b), q = ab$, we can turn this homoegenous second-order ODE into a homogenous first-order ODE:

$$\begin{aligned}y'' + py' + qy &= y'' - (a+b)y' + aby \\ &= (y' - ay)' + b(ay - y') \\ &= 0 \\ \Leftrightarrow (y' - ay)' &= b(y' - ay)\end{aligned}$$

Making the substitution $u = y' - ay$ and solving the resulting first-order ODE we then have:

$$\begin{aligned}u' &= bu \\ \Leftrightarrow u(x) &= Ce^{bx} \\ \Rightarrow y' - ay &= Ce^{bx} \\ \Leftrightarrow (ye^{-ax})' &= Ce^{(b-a)x} \\ \Leftrightarrow y(x) &= c_1e^{ax} + c_2e^{bx}\end{aligned}$$

Unless $a = b$, in which case the fourth line becomes

$$\begin{aligned}(ye^{-ax})' &= C \\ \Leftrightarrow y(x) &= e^{ax}(c_1x + c_2)\end{aligned}$$

So all we need to do to solve a homogenous second-order ODE with constant coefficients p, q is to find a, b such that $p = -(a+b), q = ab$. We can do this easily by noting that a, b are the solutions to the quadratic equation

$$r^2 + pr + q = r^2 - (a+b)r + ab = (r-a)(r-b) = 0$$

We call this equation the *characteristic equation* of the above second-order ODE. Using the quadratic formula, we then have:

$$a, b = \frac{-p \pm \sqrt{p^2 - 4q}}{2}$$

C.3 Nth-order Inhomogenous ODEs: Green's Function

Consider a differential equation of the form

$$\frac{\mathrm{d}^n y}{\mathrm{d}x^n} + \frac{\mathrm{d}^{n-1} y}{\mathrm{d}x^{n-1}} + \ldots + \frac{\mathrm{d}y}{\mathrm{d}x} + y(x) = g(x)$$

Recalling that differentiation is a linear operation, we can define the linear operator L:

$$Ly(x) = \frac{\mathrm{d}^n y}{\mathrm{d}x^n} + \ldots + \frac{\mathrm{d}y}{\mathrm{d}x} + y(x)$$

We now find a function $G(x, s)$ such that

$$LG(x, s) = \delta(s - x)$$

This is called a *Green's function*, which depends on the linear operator L.

Once we have found the Green's function, we can use it to solve the differential equation by noting that

$$\int LG(x, s)g(s)\mathrm{d}s = g(x)$$

Crucially, since L is a linear operator with respect to x (not s), we can pull it out of the integral. We can thus rewrite the differential equation as:

$$\begin{aligned} Ly(x) &= g(x) \\ &= \int LG(x, s)g(s)\mathrm{d}s \\ &= L\int G(x, s)g(s)\mathrm{d}s \\ \Leftrightarrow y(x) &= \int G(x, s)g(s)\mathrm{d}s \end{aligned}$$

which will hopefully be an easy integral if the Green's function $G(x, s)$ is of a nice form.

C.4 Ricatti Equations

NOTE: Thank you Wikipedia.

A *Ricatti equation* is any first-order ODE that is quadratic in the unknown function. In other words, it is of the form

$$\ddot{y}(t) = q_0(t) + q_1(t)y(t) + q_2(t)y^2(t), \tag{269}$$

where $q_0, q_2 \neq 0$. To solve such an equation, we can reduce it to a linear second-order ODE, solvable using the method described above.

Given eq. 269, we can say that wherever q_2 is non-zero and differentiable, $v :== yq_2$ satisfies a Ricatti equation of the form

$$\dot{v} = v^2 + R(t)v + S(t), \tag{270}$$

where $S = q_2 q_0$ and $R = q_1 + \frac{\dot{q}_2}{q_2}$, because

$$\begin{aligned} \dot{v} &= \frac{d}{dt}(yq_2) = \dot{y}q_2 + y\dot{q}_2 \\ &= (q_0 + q_1 y + q_2 y^2)q_2 + v\frac{\dot{q}_2}{q_2} = q_0 q_2 + \left(q_1 + \frac{\dot{q}_2}{q_2}\right)v + v^2. \end{aligned} \tag{271}$$

Substituting $v = -u'/u$, it follows that u satisfies the linear second order ODE given by

$$\ddot{u} - R(t)\dot{u} + S(t)u = 0, \tag{272}$$

since

$$\dot{v} = -\frac{d}{dt}(\dot{u}/u) = -\ddot{u}/u + (\dot{u}/u)^2 = -\frac{\ddot{u}}{u} + v^2, \tag{273}$$

so that

$$\frac{\ddot{u}}{u} = v^2 - \dot{v} = -S + R\frac{\dot{u}}{u} \tag{274}$$

and hence

$$\ddot{u} - R(t)\dot{u} + S(t)u = 0. \tag{275}$$

Therefore, to solve an equation of this form, all we need to do is find S and R using the expressions above, then solve eq. 272 for u. The final result is given by

$$y = -\frac{\dot{u}}{q_2 u}. \tag{276}$$

D Dynamical Systems Analysis

D.1 1D Systems

For 1D systems, we can simply plot the derivative on the phase plane with $\dot{x} = f(x)$ on the y-axis and x on the x-axis. Fixed points then occur wherever $f(x) = 0$. Given a fixed point x^*, we can say that it's stable if $f(x^* + \delta x) < 0$ and unstable if $f(x^* + \delta x) > 0$, for some zero-centered small Gaussian perturbation δx. This is intuitive, as a negative slope at the fixed point implies that the derivative is positive to the left of the point (x is increasing) and it is negative to the right of the point (x is decreasing)—thus the dynamics are converging to the fixed point. The converse is clearly true for a positive slope.

The sign of the slope can be checked by linearizing the derivative around the fixed point via a first order Taylor expansion:

$$f(x^* + \delta x) \approx \underbrace{f(x^*)}_{0} + \delta x \frac{df}{dx}\Big|_{x=x^*}. \tag{277}$$

(More details on linearization below.) Another useful fact to keep in mind is that 1D ODEs cannot express periodic behavior. You can't draw a phase portrait on the real line that oscillates—it will always either tend to $\pm\infty$ or a fixed point.

D.2 2D Systems

Consider a dynamical system of the form

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(x, y)$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = g(x, y)$$

Since the dynamics depend only on the variables x, y themselves and nothing else, we call such a system *autonomous*. To understand this system, we would like to know the behavior of trajectories of $(x(t), y(t))$ over time. We can gain such an understanding qualitatively by plotting the *nullclines* of the system in the $x - y$ plane, given by

$$f(x, y) = 0$$
$$g(x, y) = 0$$

We can then construct the so-called *phase plane* by sketching trajectories in the $x-y$ plane. For simple systems, we can directly calculate trajectories by picking an initial condition and solving the differential equation

$$\frac{\mathrm{d}x}{\mathrm{d}y} = \frac{f(x, y)}{g(x, y)}$$

However, this is usually impossible to do analytically, so we instead turn to the nullclines to guide us via the following rules:

- Trajectories can only cross the $x-$nullcline $f(x, y) = 0$ vertically (i.e. with $\frac{\mathrm{d}x}{\mathrm{d}t} = 0$)
- Trajectories can only cross the $y-$nullcline $g(x, y) = 0$ horizontally (i.e. with $\frac{\mathrm{d}y}{\mathrm{d}t} = 0$)
- Regions enclosed by the nullclines have $\frac{\mathrm{d}x}{\mathrm{d}y}$ with constant sign
- Crossings of the two nullclines are fixed points (stable/unstable) of the system

This last point is of great importance, as often what we are most interested in is the long-run behavior of the system. Thus, we would like to be able to know the behavior of the system near each of the fixed points. We can do so via standard stability analysis. Consider a fixed point (x^*, y^*) of the above system given, found by solving the equation

$$f(x^*, y^*) = g(x^*, y^*) = 0$$

To understand the system's behavior near this point, we analyze the dynamics at a nearby point

$$(\tilde{x}(t), \tilde{y}(t)) = (x^* + \delta x(t), y^* + \delta y(t))$$

to examine where it ends up in the limit of $t \to \infty$. If $(\tilde{x}(t), \tilde{y}(t)) \to (x^*, y^*)$, i.e. $(\delta x(t), \delta y(t)) \to (0, 0)$, as $t \to \infty$ then we know the fixed point (x^*, y^*) is stable.

Assuming $(\delta x(t), \delta y(t))$ to be very small, we can safely approximate the dynamics at $(\tilde{x}, \tilde{y})$ to 1st order:

$$\frac{\mathrm{d}\tilde{x}}{\mathrm{d}t} = \frac{\mathrm{d}\delta x}{\mathrm{d}t} = f(x^* + \delta x, y^* + \delta y) \approx f(x^*, y^*) + f_x(x^*, y^*)\delta x + f_y(x^*, y^*)\delta y$$
$$\frac{\mathrm{d}\tilde{y}}{\mathrm{d}t} = \frac{\mathrm{d}\delta y}{\mathrm{d}t} = g(x^* + \delta x, y^* + \delta y) \approx g(x^*, y^*) + g_x(x^*, y^*)\delta x + g_y(x^*, y^*)\delta y$$

where I have used the notation $f_z(a, b) = \left.\frac{\partial f}{\partial z}\right|_{x=a,y=b}$. Since $f(x^*, y^*) = g(x^*, y^*) = 0$, we can rewrite this approximation in matrix notation as follows:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{J}\mathbf{x}$$

where

$$\mathbf{x} = \begin{bmatrix} \delta x \\ \delta y \end{bmatrix}$$

and

$$\mathbf{J} = \begin{bmatrix} f_x(x^*, y^*) & f_y(x^*, y^*) \\ g_x(x^*, y^*) & g_y(x^*, y^*) \end{bmatrix}$$

is the Jacobian of the vector-valued function $\mathbf{f}(x, y) = \begin{bmatrix} f(x, y) & g(x, y) \end{bmatrix}^{\mathrm{T}}$, evaluated at (x^*, y^*).

We now have a linear dynamical system that we can actually solve. Note what we have done: by picking a point very near to the fixed point and approximating its dynamics to first-order, we have effectively *linearized* the dynamics around this fixed point, giving us a linear system that we can analyze and solve. Specifically, the solution to this linear system is given by

$$\mathbf{x}(t) = c_1 e^{\lambda_1 t}\mathbf{v}_1 + c_2 e^{\lambda_2 t}\mathbf{v}_2$$

where λ_1, λ_2 and $\mathbf{v}_1, \mathbf{v}_2$ are the eigenvalues and eigenvectors of the 2×2 Jacobian matrix $\mathbf{J}$. Thus, if $\mathrm{Re}(\lambda_1), \mathrm{Re}(\lambda_2) < 0$, we know that $e^{\lambda_1 t}, e^{\lambda_2 t} \to 0$ and therefore $(\delta x, \delta y) \to 0$ as $t \to \infty$, so we canc conclude (x^*, y^*) is a stable fixed point. Otherwise, (x^*, y^*) could be either unstable, a saddle node, or a limit cycle (see table ??). We therefore need only calculate the eigenvalues of the Jacobian matrix $\mathbf{J}$ to determine qualitative behavior around the fixed point (x^*, y^*):

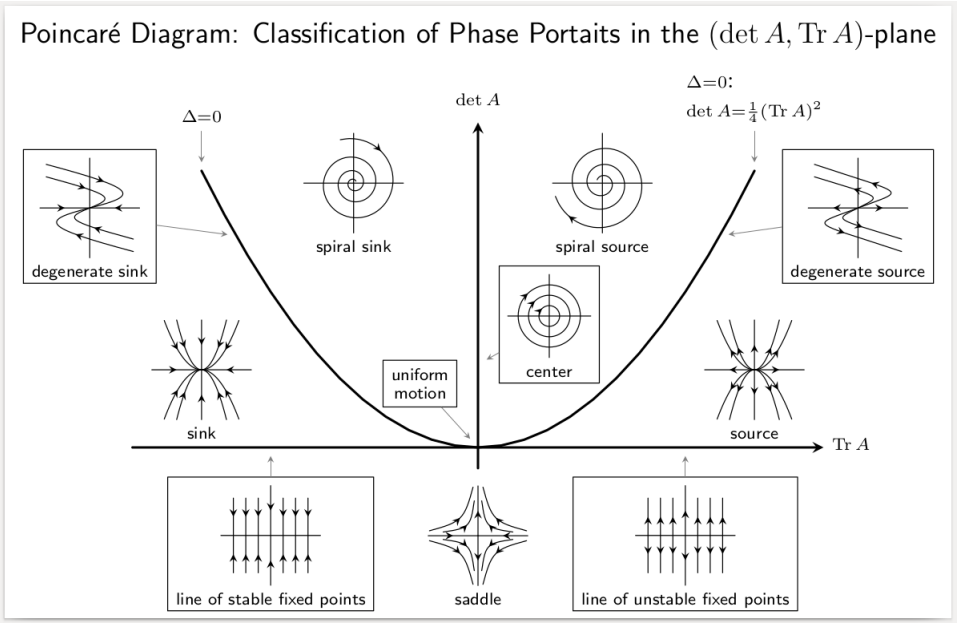
$$\begin{aligned} \mathbf{J}\mathbf{v} &= \lambda\mathbf{v} \\ \Leftrightarrow (\mathbf{J} - \lambda\mathbf{I})\mathbf{v} &= \mathbf{0} \\ \Rightarrow |\mathbf{J} - \lambda\mathbf{I}| &= \mathbf{0} \quad \text{for non-zero } \mathbf{v} \\ \Leftrightarrow \lambda^2 - \underbrace{\mathrm{Tr}[\mathbf{J}]}_{T}\lambda + \underbrace{|\mathbf{J}|}_{D} &= 0 \\ \Rightarrow \lambda_\pm &= \frac{T \pm \sqrt{T^2 - 4D}}{2} \end{aligned}$$

where the third line follows from the fact that, for there to be a non-zero vector $\mathbf{v}$ that satisfies the equation in the second line, the matrix $\mathbf{J} - \lambda\mathbf{I}$ must have a non-zero nullspace and therefore not be full-rank, which implies that its determinant must be 0. We can thus easily derive the following conditions for stability of the fixed point:

$$T < 0$$
$$D > 0$$

The full picture is given by table ?? and figure D.2 below.

Fixed point	Tr[**J**]	Det[**J**]	Real part	Imaginary part
stable node	$T < 0$	$T^2 > 4D > 0$	$\mathrm{Re}(\lambda_\pm) < 0$	$\mathrm{Im}(\lambda_\pm) = 0$
stable spiral	$T < 0$	$4D > T^2 > 0$	$\mathrm{Re}(\lambda_\pm) < 0$	$\mathrm{Im}(\lambda_\pm) \neq 0$
unstable node	$T > 0$	$T^2 > 4D > 0$	$\mathrm{Re}(\lambda_\pm) > 0$	$\mathrm{Im}(\lambda_\pm) = 0$
unstable spiral	$T > 0$	$4D > T^2 > 0$	$\mathrm{Re}(\lambda_\pm) > 0$	$\mathrm{Im}(\lambda_\pm) \neq 0$
center (limit cycle??)	$T = 0$	$D > 0$	$\mathrm{Re}(\lambda_\pm) = 0$	$\mathrm{Im}(\lambda_\pm) \neq 0$
saddle	-	$D < 0$	$\mathrm{Re}(\lambda_+) > 0 > \mathrm{Re}(\lambda_-)$	$\mathrm{Im}(\lambda_\pm) = 0$
star/degenerate node	$T^2 = 4D$	$D \geq 0$	$\mathrm{Re}(\lambda_+) = \mathrm{Re}(\lambda_-)$	$\mathrm{Im}(\lambda_\pm) = 0$



A useful theorem that is often quoted:

Theorem 2 (Poincare-Bendixson Theorem)**.** *If both:*

1. *There is an unstable fixed point*
2. *You can draw a closed loop around the fixed point and the flow points inwards at all points on the loop*

Then there is a stable limit cycle around the unstable fixed point.

It probably has some horrendously complcated proof, but it seems kind of self-evident to me.

E Fourier Transform

Given a function $f(x)$ in space or time (i.e. x in cm or seconds), one can equialently express it in the frequency domain via its Fourier transform $F(\omega)$:

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \omega x} \mathrm{d}x$$
$$f(x) = \int_{-\infty}^{\infty} F(\omega) e^{2\pi i \omega x} \mathrm{d}\omega$$

When the units of ω don't matter to the given derivation, the 2π can be dropped. One must then simply rescale $\omega \to \frac{\omega}{2\pi}$ to interpret it as a frequency in inverse units of x (e.g. Hz for x in seconds).

Some important Fourier transforms to know are given in the table below, where $\omega > 0$ is in inverse units of x:

$f(x)$	$F(\omega)$
$\sin(kx)$	$\delta(\omega - 2\pi k)$
$\delta(x)$	1
$\frac{d^n}{dx^n} g(x)$	$(2\pi i\omega)^n G(\omega)$
$e^{-ax^2}, a > 0$	$\sqrt{\frac{\pi}{a}} e^{-\frac{\pi^2 \omega^2}{a}}$
$\Theta(x) e^{-ax}, a > 0$	$\frac{1}{2\pi i\omega + a}$

A useful property of the Fourier transform is the so-called *Convolution Theorem*, which states that the Fourier transform of the convolution of two functions is equal to the product of their Fourier transforms. Let

$$h(x) = \int f(x') g(x - x') \mathrm{d}x'$$

be the convolution of $f(\cdot)$ and $g(\cdot)$. Then its Fourier transform is:

$$\begin{aligned}
H(\omega) &= \int h(x)e^{-2\pi i\omega x}\mathrm{d}x \\
&= \int\int f(x')g(x-x')\mathrm{d}x'e^{-2\pi i\omega x}\mathrm{d}x \\
&= \int f(x')\int g(x-x')e^{-2\pi i\omega x}\mathrm{d}x\mathrm{d}x' \\
&\overset{y=x-x'}{=} \int f(x')\int g(y)e^{-2\pi i\omega(y+x')}\mathrm{d}y\mathrm{d}x' \\
&= \int f(x')e^{-2\pi i\omega x'}\mathrm{d}x'\int g(y)e^{-2\pi i\omega y}\mathrm{d}y \\
&= F(\omega)G(\omega)
\end{aligned}$$

where in the third line we switched the order of integration[13] and in the fourth line we made the substitution $y = x - x'$.

F Central Limit Theorem

The Central Limit Theorem states that for any set of *independent* 0-mean random variables $X_1, X_2, \ldots, X_n$ with variances given by $\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2$, in the limit of $n \to \infty$

$$P\left(\frac{\sum_{i=1}^n X_i}{\sqrt{n}\sigma} < C\right) = P(Z_n < C) \to P(Z < C), \quad Z \sim \mathcal{N}(0,1)$$

where

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^n \sigma_i^2$$

The arrow here means that the cumulative distribution of the random variable Z_n *converges in distribution* to that of a standard Gaussian. This means that different segments of the distribution may converge to Gaussianity at different rates (e.g. the tails of its distribution will converge more slowly).

We prove it here via the *moment generating function* of a random variable X:

$$M_X(t) \equiv \mathbb{E}[e^{tX}]$$

Noting that this implies

$$M_X(t) = \mathbb{E}\left[1 + tX + \frac{1}{2!}t^2X^2 + \frac{1}{3!}t^3X^3 + ...\right] = 1 + t\mathbb{E}[X] + \frac{1}{2!}t^2\mathbb{E}[X^2] + \frac{1}{3!}t^3\mathbb{E}[X^3] + ...$$

it is easy to see that the following holds:

$$\left.\frac{\mathrm{d}^\ell M_X}{\mathrm{d}t^\ell}\right|_{t=0} = \mathbb{E}[X^\ell]$$

Thus its name.

We then require four facts:

1. The moment generating function of a sum of independent random variables $Z = X + Y$ is the product of their moment generating functions $M_X(t), M_Y(t)$:

$$M_Z(t) = \mathbb{E}[e^{t(X+Y)}] = \mathbb{E}[e^{tX}e^{tY}] = \mathbb{E}[e^{tX}]\mathbb{E}[e^{tY}] = M_X(t)M_Y(t)$$

2. The moment generating function of a linear transformation of a random variable $Z = aX + b$ is given by:

$$M_Z(t) = \mathbb{E}[e^{atX+bt)}] = e^{bt}\mathbb{E}[e^{atX}] = e^{bt}M_X(at)$$

3. If the moment generating functions $M_{X_1}(t), M_{X_2}(t), \ldots$ of a sequence of random variables $X_1, X_2, \ldots$ converge to some moment generating function $M_X(t)$, i.e.

$$\lim_{n\to\infty} M_{X_n}(t) = M_X(t)$$

then their respective cumulative density functions $F_1(x), F_2(x), \ldots$ converge in distribution to the cumulative density function $F(x)$ of X:

$$\lim_{n\to\infty} F_n(x) = F(x)$$

[13] This is allowed under certain relatively soft constraints on $f(x), g(x)$, namely that the integral of their absolute value be finite, I believe (cf. Fubini's Theorem).

4. The moment generating function of a standard Gaussian random variable $X \sim \mathcal{N}(0,1)$ is given by:

$$\begin{aligned} M_X(t) &= \mathbb{E}[e^{tX}] \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{tx - \frac{x^2}{2}} \mathrm{d}x \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{(x^2 - 2tx)}{2}} \mathrm{d}x \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{(x-t)^2 - t^2}{2}} \mathrm{d}x \\ &= \frac{e^{\frac{t^2}{2}}}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{(x-t)^2}{2}} \mathrm{d}x \\ &= e^{\frac{t^2}{2}} \end{aligned}$$

By points 3 and 4, then, to prove the CLT it suffices to show that

$$\lim_{n\to\infty} M_{Z_n}(t) = e^{\frac{t^2}{2}}$$

Using points 1 and 2, we have:

$$M_{Z_n}(t) = \prod_{i=1}^{n} M_{X_i}\left(\frac{t}{\sqrt{n}\sigma}\right)$$

Expanding the individual moment generating functions, we have:

$$M_{X_i}\left(\frac{t}{\sqrt{n}\sigma}\right) = 1 + \frac{t}{\sqrt{n}\sigma}\mathbb{E}[X_i] + \frac{t^2}{2n\sigma^2}\mathbb{E}[X_i^2] + \frac{t^3}{3!n^{3/2}\sigma^3}\mathbb{E}[X_i^3] + \ldots$$

As $n \to \infty$, the latter terms will go to 0 faster than the earlier terms, eventually giving us

$$M_{X_i}\left(\frac{t}{\sqrt{n}\sigma}\right) \to 1 + \frac{t^2\sigma_i^2}{2n\sigma^2}$$

where I have also substituted in $\mathbb{E}[X_i] = 0, \mathbb{E}[X_i^2] = \sigma_i^2$ (true for all i). Assuming the individual variances σ_i^2 are not too different from each other, $\sigma_i^2/\sigma^2 \sim \mathcal{O}(1)$ and we can ignore it. Thus,

$$M_{Z_n}(t) \to \left(1 + \frac{t^2}{2n}\right)^n \to e^{\frac{t^2}{2}}$$

There is a more rigorous way of proving this without any handwaving, but one can see that this definitely holds for the case where the X_i have equal variance (i.e. when they are i.i.d.).

F.1 Rough Size of Sum of Uncorrelated Variables

It is useful to know the scaling with n of a sum of n zero-mean variables:

$$Z = \sum_i X_i$$

X_i are distributed iid according to some distribution with zero mean and finite variance, σ. $\langle Z \rangle = 0$ since each term is zero mean. But the variance of Z is not zero:

$$\langle Z^2 \rangle = \langle \sum_{ij} X_i X_j \rangle = \langle \sum_i X_i^2 \rangle = N\sigma^2$$

Therefore the typical size of Z goes with its standard deviation, which is $\mathcal{O}(\sqrt{n})$.

G Useful Approximations and Maths Facts

- $\frac{d}{dx}\tanh(x) = 1 - \tanh^2(x)$
- $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

- $\log(1+x) \approx x \quad$ for small x
- $(1+x/n)^n \to e^x, \quad$ for $n \to \infty$
- $\frac{d}{dx}\text{arctanh}(x) = \frac{1}{1-x^2}$
- $\int_{-\infty}^{\infty} \delta(t-T)f(t)\,dt = f(T)$
- $\int_{-\infty}^{\infty} \delta(t-T)\,dt = \Theta(t-T) \Leftrightarrow \frac{d}{dt}\Theta(t) = \delta(t)$
- The entropy of a Gaussian is $\frac{1}{2}\log(2\pi e\sigma^2)$
- The Gaussian CDF is $\Phi(z) \coloneqq \int_{-\infty}^{z} e^{-t^2/2}/\sqrt{2\pi}\,dt$. Some useful properties to keep in mind are
  - Probability: $1 - \Phi(r) = P[z > r]$
  - Symmetry: $1 - \Phi(r) = \Phi(-r)$
  - Sigmoidal, with $\Phi(0) = 1/2$
  - In general, draw out the Gaussian and bounds to help convert between probabilities and cdfs
- The parity of a function: f is *even* if $f(x) = f(-x)$, and *odd* if $f(x) = -f(-x) \Leftrightarrow -f(x) = f(-x)$.
- Differentation/integration switch the parity of a function.
- The integral of an even function from $-\infty$ to ∞ is twice the integral from 0 to ∞, the integral of an odd function from $-\infty$ to ∞ is zero.
- The sign$(\cdot)$ function is an odd function.
- $\text{sign}^2(x) = 1$ always.

References

[Dayan and Abbott, 2001] Dayan, P. and Abbott, L. F. (2001). *Theoretical neuroscience*, volume 806. Cambridge, MA: MIT Press.

[Gerstner et al., 2014] Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

[Grabska-Barwińska and Latham, 2014] Grabska-Barwińska, A. and Latham, P. E. (2014). How well do mean field theories of spiking quadratic-integrate-and-fire networks work in realistic parameter regimes? *Journal of computational neuroscience*, 36(3):469–481.

[Graupner and Brunel, 2012] Graupner, M. and Brunel, N. (2012). Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location. *Proceedings of the National Academy of Sciences of the United States of America*, 109(10):3991–6.

[Izhikevich, 2007] Izhikevich, E. M. (2007). *Dynamical systems in neuroscience*. MIT press.

[Latham, 2017] Latham, P. E. (2017). Correlations demystified. *Nature neuroscience*, 20(1):6.

[Latham et al., 2000] Latham, P. E., Richmond, B., Nelson, P., and Nirenberg, S. (2000). Intrinsic dynamics in neuronal networks. i. theory. *Journal of neurophysiology*, 83(2):808–827.

[Laughlin et al., 1981] Laughlin, S. B. et al. (1981). A simple coding procedure enhances a neuron's information capacity. *Z. Naturforsch*, 36(910-912):51.

[Lillicrap et al., 2016] Lillicrap, T. P., Cownden, D., Tweed, D. B., and Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7(1):13276.

[Mastrogiuseppe and Ostojic, 2017] Mastrogiuseppe, F. and Ostojic, S. (2017). Intrinsically-generated fluctuating activity in excitatory-inhibitory networks. *PLoS computational biology*, 13(4):e1005498.

[Menendez, 2018] Menendez, J. A. (2018). Gatsby theoretical neuroscience notes.

[Moskovitz, 2020] Moskovitz, T. (2020). Tn guide.

[Renart et al., 2010] Renart, A., De La Rocha, J., Bartho, P., Hollender, L., Parga, N., Reyes, A., and Harris, K. D. (2010). The asynchronous state in cortical circuits. *science*, 327(5965):587–590.

[Rosenbaum et al., 2017] Rosenbaum, R., Smith, M. A., Kohn, A., Rubin, J. E., and Doiron, B. (2017). The spatial structure of correlated neuronal variability. *Nature neuroscience*, 20(1):107.

[Roudi and Latham, 2007] Roudi, Y. and Latham, P. E. (2007). A balanced memory network. *PLoS computational biology*, 3(9):1679–700.

[Sahani and Dayan, 2003] Sahani, M. and Dayan, P. (2003). Doubly distributional population codes: Simultaneous representation of uncertainty and multiplicity. *Neural Computation*, 15(10):2255–2279.

[Sahani and Latham, 2021] Sahani, M. and Latham, P. (2021).

[Sompolinsky et al., 1988] Sompolinsky, H., Crisanti, A., and Sommers, H.-J. (1988). Chaos in random neural networks. *Physical review letters*, 61(3):259.

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Introduction to reinforcement learning*. MIT Press.

[Tsodyks and Feigel'man, 1988] Tsodyks, M. V. and Feigel'man, M. V. (1988). The enhanced storage capacity in neural networks with low activity level. *Europhysics Letters (EPL)*, 6(2):101–105.

[Van Rossum et al., 2000] Van Rossum, M. C., Bi, G. Q., and Turrigiano, G. G. (2000). Stable hebbian learning from spike timing-dependent plasticity. *The Journal of Neuroscience*, 20(23):8812–8821.

[Vreeswijk and Sompolinsky, 1998] Vreeswijk, C. v. and Sompolinsky, H. (1998). Chaotic balanced state in a model of cortical circuits. *Neural computation*, 10(6):1321–1371.

[Wilson and Cowan, 1972] Wilson, H. R. and Cowan, J. D. (1972). Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical journal*, 12(1):1–24.

[Zhang and Sejnowski, 1999] Zhang, K. and Sejnowski, T. J. (1999). Neuronal tuning: To sharpen or broaden? *Neural Computation*, 11(1):75–84.