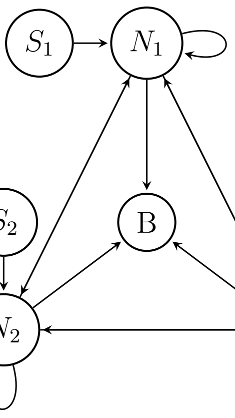
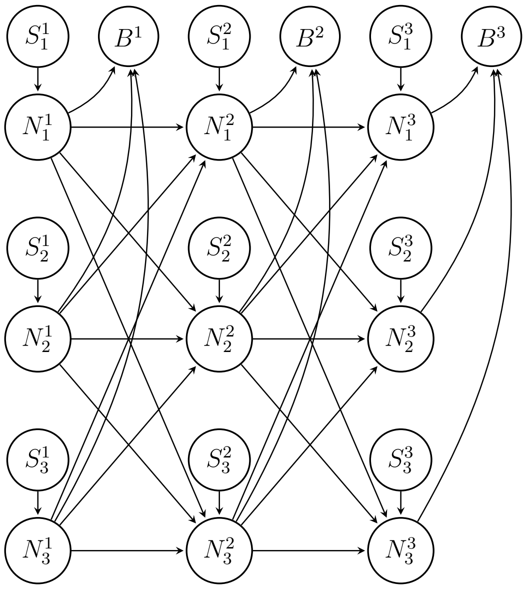
NDSEG Mathematical Appendix

Tyler Benster

1 Introduction



(a) Schematic of three densely-connected neurons (N) with single-unit stimulation (S) and behavior readout (B).



(b) Dynamic bayesian network for time $t \in \{1, 2, 3\}$.

Figure 1

Suppose $\mathcal{N} = \{\forall i N_i\}$ is a set of neurons indexed by $i \in I, I = \{1, ..., n\}$. Let the random variable $N_i^t : \Omega \to [0, 1]$ represent the fluorescence of neuron i at time $t \in \mathbb{N}$ for a sample space Ω, and $\mathcal{N}^t = \{\forall i N_i^t\}$. We denote a single observation of fluorescence by n_i^t.

2 Rate of transmission

See **Figure 2a**. For the sake of simplicity, we only consider binary states $S_i^t \in \{-1, 1\}$ and $N_i^t \in \{0, 1\}$. We achieve the latter by creating an activation function a with activation threshold α:

$$a(n) = \begin{cases} 1, & \text{if } n \geq \alpha \\ 0, & \text{otherwise} \end{cases}$$

Let $\forall i \forall t (A_i^t = a(N_i^t))$ and $\mathcal{A}^t = \{A_1^t, ..., A_n^t\}$. Then, the rate of transmission R between $\mathcal{S}^t$ and $\mathcal{A}^t$ can be calculated with the help of the information entropy equation H:

$$R = H(\mathcal{S}^t) - H(\mathcal{N}^t | \mathcal{S}^t)$$

If we choose a binary stimulus for each neuron independently and uniformly, then

$$H(\mathcal{S}^t) = -\sum_{i=1}^{n} 0.5 \log_2(0.5) + 0.5 \log_2(0.5) = n$$

Thus, the stimulus entropy is n bits. To simplify the calculation of $H(\mathcal{N}^t|\mathcal{S}^t)$, we assume that a maximal optogenetic stimuli dominates neuronal input:

$$(do(S_j^o) = -1 \vee do(S_j^o) = 1) \rightarrow (p(N_j^o|\forall i \forall \tau N_i^{o-\tau}, S_i^o) = p(N_j^o|S_j^o) = do(S_i^o))$$

The calculation for rate of transmission is then straightforward:

$$R = n - \sum_i H(N_i^t|do(S_i^t))$$

If we choose a non-uniform stimulus space, we can calculate the redundancy by one minus the ratio of a source to its maximum entropy.

3 Feedforward entropy

See **Figure 2b**. We use variational inference to calculate a network's feedforward entropy. Variational inference provides a locally-optimal, exact analytic solution to approximate the posterior distribution of feedforward entropy; however, manually deriving the exact set of equations is infeasible for all but the smallest networks. Instead, we use the probabilistic programming language *Pyro* to iteratively update parameters. The rough algorithm for updating the bayesian network after each optogenetic stimulus and recorded neuronal response is as follows:

1. Sample n_i input neurons in a set X and $n - n_i$ output neurons in a set Y such that $X \cap Y = \emptyset$ and $X \cup Y = \mathcal{N}$.
2. Sample an optogenetic stimulus $\mathcal{S}$ where $\mathcal{S}_x = \{S_i | i \in \{j | N_j \in X\}\}$, $S \in \mathcal{S}_x \rightarrow (S = 1 \vee S = -1)$ and $S \notin \mathcal{S}_x \rightarrow S = 0$.
3. Observe the neuronal response (fluorescence) of Y
4. Calculate $H(X, Y)$ and use this observation to update our posterior distribution of belief for feedforward entropy.
5. Repeat 1-4, randomly sampling a new X and Y each time.

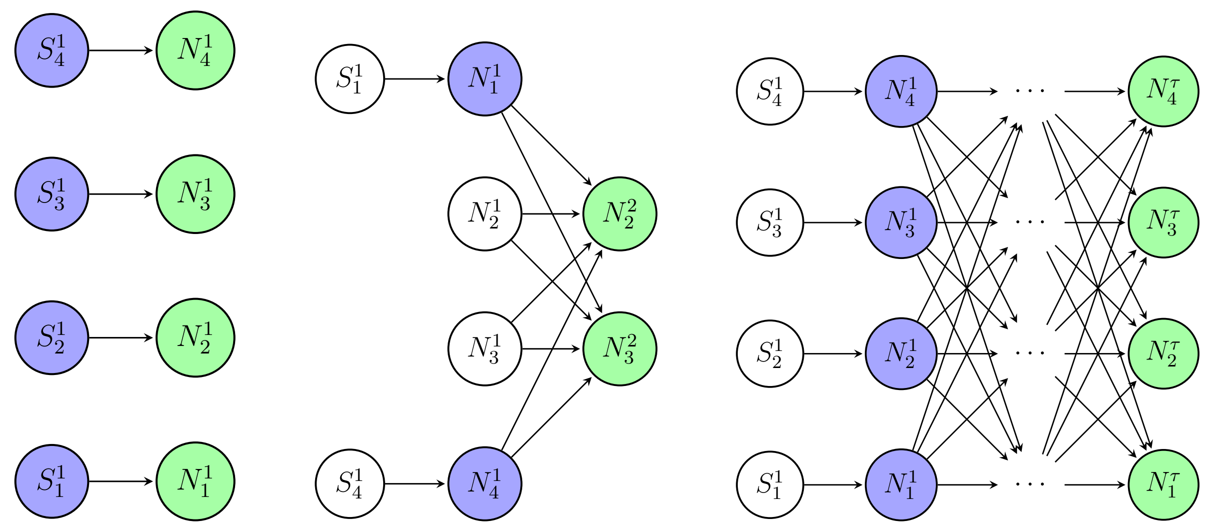
The feedforward entropy limit is obtained at $n_i = n/2$. We can test if this theoretical result holds for an arbitrary neuronal network by repeating the protocol for multiple choices of $n_i \in [1, n-1]$.

4 Entropy production

See **Figure 2c**. To calculate entropy production, we first choose a time t_i to stimulate input, and a time t_o to observe neuronal response where $t_o > t_i$. We define entropy production as the change in entropy per unit of time.

$$\frac{H(\mathcal{N}_i^t) - H(\mathcal{N}_i^t|\mathcal{N}_o^t)}{t_o - t_i}$$

We likewise use variational inference to sample from stimulus space, observe response space, and update our posterior belief of entropy production.



(a) The entropy between the stimulus and recorded neuronal response is the rate of transmission.

(b) In feedforward entropy, neurons receiving stimulation are separate from neurons whose output we observe.

(c) Entropy production is the rate of change in entropy between time 1 and time τ.

Figure 2: Entropy is measured between blue nodes and green nodes.

5 Bayesian Network

We define a first-order bayesian network with a graph inspired from feedforward neural networks (**Figure 1**):

$$p(N_1^t, ..., N_n^t) = p(N_1^{t-1})p(N_2^{t-1}|N_1^{t-1}) \cdots p(N_n^{t-1}|N_1^{t-1}, ..., N_{n-1}^{t-1})$$

Let $j \in I$ and $o \in T$ be arbitrary. By the first-order assumption,

$$p(N_j^o|\forall i \forall \tau N_i^{o-\tau}) = p(N_j^o|\forall i N_i^{o-1})$$

To make the temporal dependencies tractable, we further assume that

$$\forall t \forall \tau p(N_j^t|\forall i N_i^{t-1}) = p(N_j^\tau|\forall i N_i^{\tau-1})$$

However, the spatial dependencies are still untractable: the joint probability distribution requires 2^n calculations for n neurons and is therefore NP-Hard. Thus, we must also reduce complexity by establishing conditional independence of neurons. This can be achieved through many methods, including by using the *PC algorithm* to eliminate edges in the bayesian network and by modeling each neuron's conditional distribution by a finite mixture model $p(N_j^t|\forall i N_i^{t-1}) = \sum_i w_i p(N_j^t|N_i^{t-1})$. The latter strengthens the relationship to deep learning networks and permits the use of automatic differentiation to optimize weights between neurons.

Optogenetic stimulation of neuron i at time t is represented by the factor $S_i^t \in [-1, 1]$, which influences N_i^t. Typically, we exogenously set our stimulation to a particular value. Following the *do*-calculus notation, we denote this by $do(N_i^t) = 1$ for maximal excitation, $do(N_i^t) = 0$ for no stimulation, and $do(N_i^t) = -1$ for maximal inhibition.

6 Long term potentiation & depression

Let i, j be arbitrary. We consider long term potentiation (LTP) as an increase in conditional entropy between N_i^{t-1} and N_j^t, and long term depression (LTD) as a decrease in conditional entropy between N_i^{t-1} and N_j^t. For simplicity, we presume perfect optogenetic control over whether a neuron is spiking or not:

$$R = H(\mathcal{S}) \rightarrow S_i^t = do(N_i^t)$$

We consider a paired stimulus at time $\tau - 1$ and τ, and consider the change in synaptic strength for time periods $t_- \in \{x | x < \tau\}$ and $t_+ \in \{x | x > \tau\}$ We define LTP of synapse (N_i, N_j) as:

$$(do(N_i^{\tau-1}) = 1) \wedge (do(N_j^{\tau}) = 1) \rightarrow \forall t_- \forall t_+ H(N_j^{t_-} | N_i^{t_- -1}) < H(N_j^{t_+} | N_i^{t_+ -1})$$

We define LTD of synapse (N_i, N_j) as:

$$(do(N_i^{\tau-1}) = 1) \wedge (do(N_j^{\tau}) = 0) \rightarrow \forall t_- \forall t_+ H(N_j^{t_-} | N_i^{t_- -1}) > H(N_j^{t_+} | N_i^{t_+ -1})$$

.

7 Connectome programming

To speed up the process of programming a circuit using LTP & LTD, we can modify multiple synapses for neuron N at each timestep. Consider synapses for potentiation $P = \{(N_i, N_j) | \forall N_i \in \mathcal{N}, \text{arbitrary} N_j \in \mathcal{N}\}$ and synapses for depression $D = \{(N_i, N_d) | \forall N_i \in \mathcal{N}, \text{arbitrary} N_j \in \mathcal{N}\}$ where $P \cap D = \emptyset$ and $P \cup D = \{(N_i, N_d) | \forall N_i, N_j \in \mathcal{N}\}$. Then, instead of doing one synapse modification per two time points, we can do multiple in parallel, starting with the neurons that are synapsed by some N_a, and subsequently proceeding to every other neuron. $\forall (N_i, N_p) \in P \forall (N_i, N_d) \in D$ where $N_i = N_a$:

$$(do(N_i^{\tau-1}) = 1) \wedge (do(N_p^{\tau}) = 1) \wedge (do(N_d^{\tau}) = 0)$$

Therefore, we can modify synaptic weights in linear time with respect to the number of neurons. Of course, we may need to undergo multiple cycles of LTP & LTD to adjust conditional entropy to the precise level we desire.

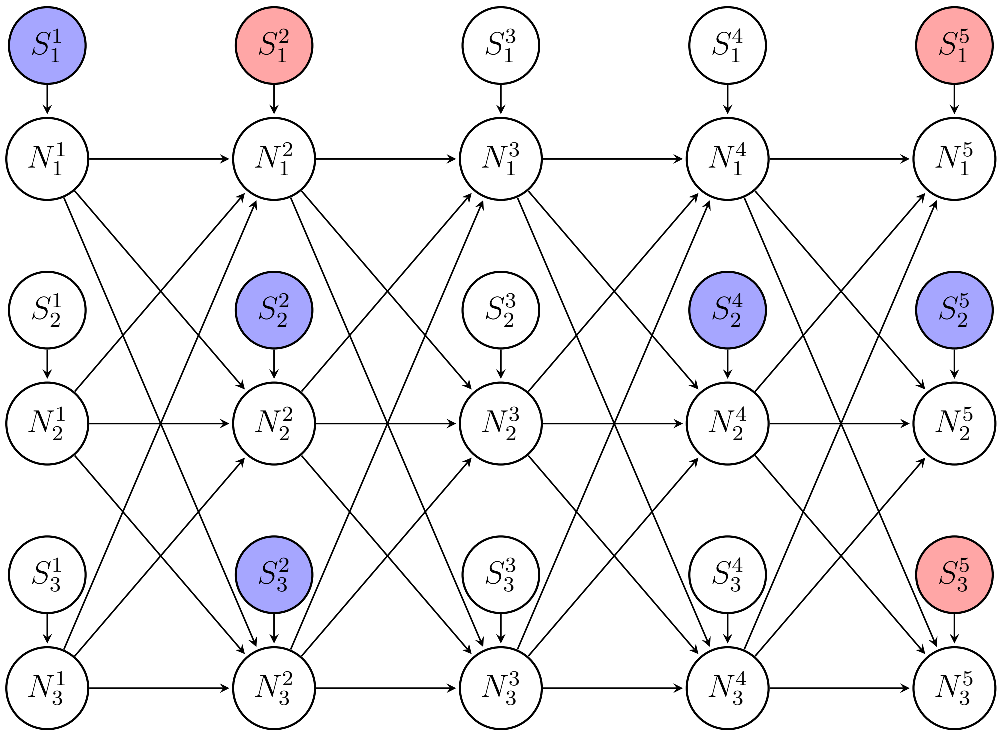


Figure 3: Implementation of LTP & LTD using excitation (blue) and inhibition (red). Starting at time 1, the synapses (N_1, N_2) and (N_1, N_3) undergo LTP while (N_1, N_1) undergoes LTD. At time 4, the synapse (N_2, N_2) undergoes LTP while (N_2, N_1) and (N_2, N_3) undergo LTD.