

Neuroscience of Learning, Memory, Cognition

Theory Homework 2

Due date: Announced for each question seperately.

Instructor: Prof. Aghajan

Sharif University of Technology, EE Department



Notes

- This assignment involves computational tasks that may be completed using programming languages such as Python or MATLAB.
- The use of AI tools is strictly prohibited.
- Include the results of your code in your answer document (PDF), and upload your source code files to CW together with the PDF in a single ZIP archive.
- If you have any questions or ambiguities regarding the assignments, please contact the teaching assistants through Telegram [Kiana Kalantari](#) (for questions 1 and 4), [Parham Talebkhamneh](#) (for questions 2 and 3).

Question 1: Linear Filter Model of a Synapse [30 pts]

This question is due Aban 20th.

A synapse can be modeled as a linear filter, where the input spike train from a presynaptic neuron influences the postsynaptic response. In this question, you will become familiar with mathematical background of the concept and then, work on some numerical examples.

1.1 Background and introduction

Spikes from the presynaptic neuron occur at discrete times t_i , and we model them as delta functions $\delta(t - t_i)$. The full spike train from synapse b , denoted $\rho_b(t)$, can be expressed as:

$$\rho_b(t) = \sum_i \delta(t - t_i)$$

This spike train is passed through a synaptic filter with kernel $K(t)$, which determines how a spike at time t_i affects the postsynaptic response at time $t > t_i$. $P_s(t)$ represents the probability that postsynaptic ion channels are open at time t , which is influenced by incoming spikes from the presynaptic neuron. The postsynaptic effect is modeled as:

$$P_s(t) = \sum_{t > t_i} K(t - t_i)$$

In this case, we assume an exponential decay kernel:

$$K(t) = e^{-t/\tau_s}$$

where τ_s is the synaptic time constant.

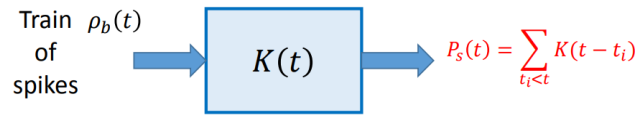


Figure 1: Linear filter model block diagram.

Note: In the following parts of this question, for simplicity, you may assume that all time variables (t , t_i) are expressed in milliseconds.

(a) [3 pts] Write the expression for the spike train $\rho_b(t)$ using delta functions. Assume you are given the spike times from synapse b :

$$t_i = \{1 \text{ ms}, 3 \text{ ms}, 7 \text{ ms}, 9 \text{ ms}, 18 \text{ ms}\}$$

(b) [3 pts] Assume $\tau_s = 5 \text{ ms}$. Compute the value of $K(t - t_i)$ at $t = 10 \text{ ms}$ for each spike time before 10 ms ($t_i < 10 \text{ ms}$), and the total postsynaptic response $P_s(t)$ at $t = 10 \text{ ms}$.

(c) [4 pts] Briefly interpret what a higher or lower value of τ_s would mean in the context of the synaptic response.

1.2 Sifting property and convolution interpretation

In signal processing and neural modeling, the Dirac delta function plays a crucial role because of its sifting property:

$$h(t) = \int_{-\infty}^{\infty} h(\tau) \delta(t - \tau) d\tau$$

This identity tells us that the delta function “picks out” the value of $h(\tau)$ at $\tau = t$, effectively sampling the function.

When modeling spike trains and postsynaptic filtering, we previously wrote the input as:

$$\rho_b(t) = \sum_i \delta(t - t_i)$$

We can use this form to derive the total postsynaptic response using convolution:

$$\sum_{t > t_i} h(t - t_i) = \int_{-\infty}^t h(t - \tau) \rho(\tau) d\tau$$

This equation shows that summing over past spike contributions is equivalent to convolving the filter $h(t)$ with the spike train $\rho(t)$.

(d) [6 pts] Assume the spike train input $\rho_b(t)$ passes through a synaptic kernel (filter) with impulse response $h(t)$. Given $\rho_b(t)$ and $h(t)$ as follows, compute the filtered output and express the result as a piecewise function of time.

$$\rho_b(t) = \delta(t - 1) + \delta(t - 2) + \delta(t - 3)$$

$$h(t) = e^{-t} u(t)$$

where $u(t)$ is the Heaviside unit step function.

1.3 Calculating the system's response to a spike train

Previously, you learned that the response of a synapse to an incoming spike train $\rho_b(t)$ can be modeled using a linear filter with kernel $K(t)$. This produces a postsynaptic response:

$$P_s(t) = \sum_{t > t_i} K(t - t_i) = \int_{-\infty}^t K(t - \tau) \rho_b(\tau) d\tau$$

For a specific synapse b from neuron n , we can define a time-varying synaptic conductance as follows. This conductance will be used to calculate the total input current into a neuron:

$$g_b(t) = g_{sb, \max} \sum_{t > t_i} K(t - t_i) = g_{sb, \max} \int_{-\infty}^t K(t - \tau) \rho_b(\tau) d\tau$$

where $g_{sb, \max}$ is the maximum possible conductance of the synapse from neuron b . It reflects the peak strength of the synaptic connection.

From different studies, we know that $I_b \propto g_b$; using the total conductance from a synapse g_b derived above, we can write the input current from synapse b into neuron n as:

$$I_b(t) = g_{sb, \max} P_{\text{rel}, b} \int_{-\infty}^t K(t - \tau) \rho_b(\tau) d\tau$$

where $P_{\text{rel}, b}$ is the release probability of neurotransmitter vesicles at synapse b . The term $g_{sb, \max} P_{\text{rel}, b}$ is a constant factor for any synapse in any neuron, which we call the *synaptic weight* w_b . Substituting w_b into the above equation gives:

$$I_b(t) = w_b \int_{-\infty}^t K(t - \tau) \rho_b(\tau) d\tau$$

In the case of multiple presynaptic neurons (indexed by $b = 1, \dots, N$), each with a synaptic weight w_b , the total synaptic input current $I_s(t)$ is obtained by summing the contributions from all individual synapses:

$$I_s(t) = \sum_{b=1}^N w_b \int_{-\infty}^t K(t - \tau) \rho_b(\tau) d\tau$$

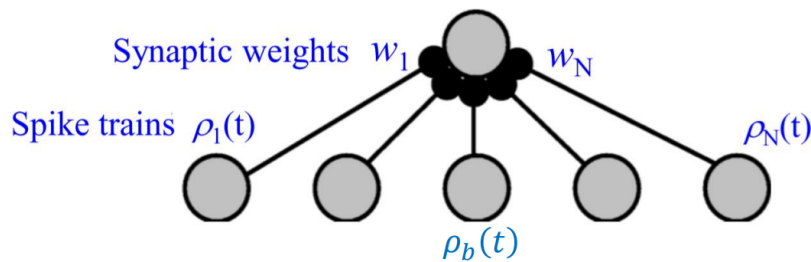


Figure 2: A neuron with multiple input synapses.

(f) [6 pts] Compute the total input current $I_s(t)$ at $t = 15$ ms for a neuron with two synapses with the following properties:

Synapse 1: $t_i^{(1)} = \{1 \text{ ms}, 6 \text{ ms}\}$, synaptic weight $w_1 = 1.0$

Synapse 2: $t_i^{(2)} = \{2 \text{ ms}, 8 \text{ ms}, 11 \text{ ms}\}$, synaptic weight $w_2 = 0.5$

You may assume the kernel as:

$$K(t) = e^{(-t/\tau_s)}, \tau_s = 5 \text{ ms}$$

(g) [8 pts] For the neuron in part (f), draw the following variables between times 0 and 25ms ($0 < t < 25ms$) using a software simulation program of your choice (suggested options: python or MATLAB).

- $\rho_1(t), \rho_2(t)$: Synapses' input train
- $P_1(t), P_2(t)$: Synapses' postsynaptic response
- I_s : Neuron's total input current.

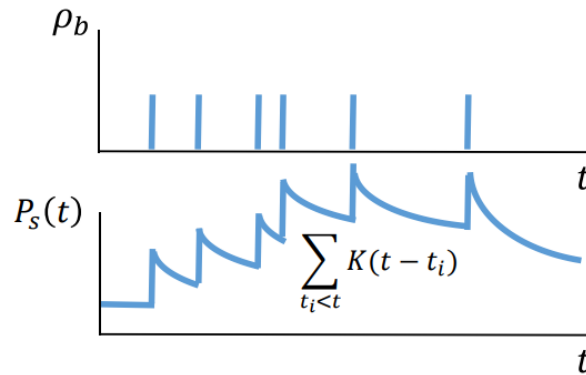


Figure 3: Sample ρ_b and P_s plot.

Question 2: Linear Recurrent Network [28 pts]

This question is due Aban 22nd.

In this question, you will become familiar with the mathematical background of a linear recurrent network and work on a numerical example at the end.

A **recurrent neural network** is a network in which neurons not only send signals forward but also send signals back to one another. This means that the current activity of the network depends on both the external input and the recurrent signals among the output neurons, as shown in Fig. 4. In this question, we aim to examine the mathematical structure of a recurrent neural network and express its dynamics in a form that is easier to analyze.

The dynamics of the network are described by the differential equation:

$$\tau \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{W}\mathbf{u} + \mathbf{M}\mathbf{v},$$

where $\mathbf{v} \in \mathbb{R}^N$ is the vector of neuron activities, $\mathbf{u} \in \mathbb{R}^K$ is a constant input vector, $\mathbf{W} \in \mathbb{R}^{N \times K}$ maps the input to the network, and $\mathbf{M} \in \mathbb{R}^{N \times N}$ contains the recurrent connections among the output neurons. The constant $\tau > 0$ is a time-scale parameter, and $\mathbf{h} := \mathbf{W}\mathbf{u} \in \mathbb{R}^N$ denotes the resulting constant drive to the network.

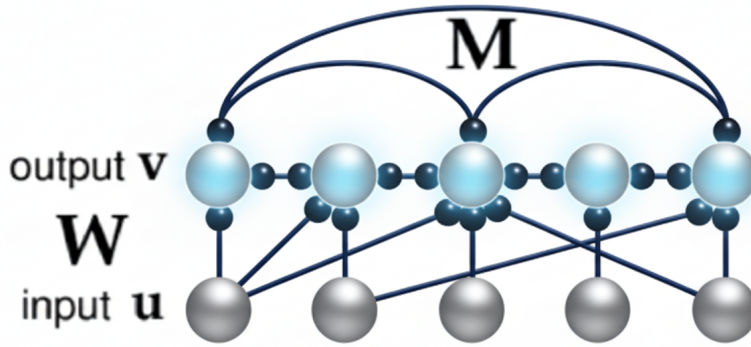


Figure 4: Overview of a linear recurrent network.

(a) [3 pts] Prove that any output state $\mathbf{v}(t)$ can be written in the form:

$$\mathbf{v}(t) = \sum_{j=1}^N c_j(t) \mathbf{e}_j,$$

when \mathbf{M} is symmetric with eigenpairs $(\lambda_i, \mathbf{e}_i)$ and orthonormal eigenvectors $\{\mathbf{e}_i\}_{i=1}^N$ satisfying $\mathbf{M}\mathbf{e}_i = \lambda_i \mathbf{e}_i$, and where $c_j(t)$, $j = 1, 2, \dots, N$, are time-dependent coefficients.

(b) [3 pts] Project the dynamics onto \mathbf{e}_i :

$$\tau \frac{d\mathbf{v}}{dt} \cdot \mathbf{e}_i = -\mathbf{v} \cdot \mathbf{e}_i + \mathbf{h} \cdot \mathbf{e}_i + (\mathbf{M}\mathbf{v}) \cdot \mathbf{e}_i.$$

Then, using the orthonormality of the eigenvectors and the definition of the eigenpair ($\mathbf{M}\mathbf{e}_i = \lambda_i \mathbf{e}_i$), simplify the equation above completely to obtain a scalar first-order ODE for $c_i(t)$.

(c) [5 pts] Solve the ODE derived in part (b) and express $c_i(t)$ explicitly in terms of $c_i(0)$, λ_i , τ , and $\mathbf{h} \cdot \mathbf{e}_i$.

(d) [5 pts] Discuss how different values of λ_i affect $c_i(t)$ and the network stability for the following cases: $\lambda_i < 1$, $\lambda_i = 1$, $\lambda_i > 1$, and complex λ_i .

(e) [4 pts] Assume all $\lambda_i < 1$ and that one eigenvalue (say, λ_1) is close to 1 while the others are much smaller. Prove that the steady state satisfies the following:

$$\mathbf{v}_{ss} \approx \frac{\mathbf{h} \cdot \mathbf{e}_1}{1 - \lambda_1} \mathbf{e}_1.$$

(f) [8 pts] (Computation on a 3-node network)

For the network described below, find $c_i(t)$, assemble $\mathbf{v}(t)$, and report the steady state \mathbf{v}_{ss} . Identify the largest eigenvalue and discuss how it influences \mathbf{v}_{ss} in light of part (e), and finally comment on stability.

$$\mathbf{M} = \begin{bmatrix} 0.2 & 0.1 & 0 \\ 0.1 & 0.6 & 0.1 \\ 0 & 0.1 & 0.3 \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \tau = 10 \text{ ms}, \quad \mathbf{v}(0) = \mathbf{0}.$$

Question 3: Working Memory as a Neural Integrator [20 pts]

This question is due Aban 22nd.

In this question, you will explore an example of a linear recurrent network. First, you will learn the mathematics underlying working memory, and then apply the concept to a practical example — the *eye-position integrator*.

3.1 Working memory

Working memory is the brain’s short-term “scratchpad” that keeps recently relevant information active so it can guide behavior (e.g., remembering a location during a delay). A simple recurrent-rate model captures this persistence: when one eigenvalue of the recurrent connectivity, λ_1 , is tuned to 1 while all others satisfy $\lambda_i < 1$, the network has a slow (ideally lossless) mode that *integrates* inputs along the direction of its first eigenvector, \mathbf{e}_1 . This makes the population activity behave like a running total of recent evidence—an elementary mechanism for working memory.

Consider the same linear recurrent network as in Question 2, described by the following equations:

$$\tau \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{h}(t) + \mathbf{M}\mathbf{v}, \quad \mathbf{v}(t) = \sum_{i=1}^N c_i(t) \mathbf{e}_i, \quad \mathbf{M}\mathbf{e}_i = \lambda_i \mathbf{e}_i,$$

where the eigenvectors $\{\mathbf{e}_i\}$ are orthonormal, and assume $\lambda_1 = 1$ while $\lambda_i < 1$ for all $i \geq 2$.

(a) [5 pts] Using the scalar ODE for $c_i(t)$ derived in Question 2, write the explicit differential equation and closed-form expression for $c_1(t)$ when $\lambda_1 = 1$ (assume $\mathbf{h}(t)$ is arbitrary and \mathbf{e}_1 is unit norm). Express your answer in terms of $c_1(0)$, τ , and the projection $\mathbf{h}(t) \cdot \mathbf{e}_1$.

(b) [4 pts] Working-memory interpretation.

Approximate the network state by the dominant mode, $\mathbf{v}(t) \approx c_1(t) \mathbf{e}_1$. Show from your result in part (a) that the network activity keeps a *running integral* of the input projection onto \mathbf{e}_1 (or along \mathbf{e}_1 when \mathbf{h} is constant). State precisely the condition under which this approximation is valid.

(c) [3 pts] Pulse input and memory

Assume $\lambda_1 = 1$, $c_1(0) = 0$, and that the input projection onto \mathbf{e}_1 is a temporary pulse given by:

$$\mathbf{h}(t) \cdot \mathbf{e}_1 = \begin{cases} 1, & 0 \leq t \leq T, \\ 0, & \text{otherwise.} \end{cases}$$

Compute $c_1(t)$ for all t and determine $\mathbf{v}(t)$. Then, sketch $c_1(t)$ as a function of time.

3.2 Eye-position integrator: response to ON/OFF burst impulses

As discussed in class, when the eyes move, **ON-direction burst neurons** produce a brief burst of spikes that shifts the eyes to a new position. After the burst ends, the eyes remain at the new location. Later, **OFF-direction burst neurons** fire to move the eyes back toward the center. Neurons in the *medial vestibular nucleus* function as a **neural integrator**, maintaining a **memory of eye position**.

For the $\lambda_1 = 1$ mode of the recurrent network, the dynamics reduce to:

$$\tau \frac{dc_1(t)}{dt} = h(t), \quad \text{eye position} \approx c_1(t).$$

We model ON and OFF bursts as narrow rectangular pulses (approximating impulse inputs), each having:

$$\text{width } w = 10 \text{ ms}, \quad \text{area } A = \tau, \quad \tau = 20 \text{ ms}.$$

This makes each ON pulse increase $c_1(t)$ by $+1$ and each OFF pulse decrease it by -1 . Thus, the input is defined as:

$$h(t) = \begin{cases} +\frac{\tau}{w}, & \text{during each of the three ON bursts,} \\ -\frac{\tau}{w}, & \text{during each of the three OFF bursts,} \\ 0, & \text{otherwise.} \end{cases}$$

(d) [8 pts] Plot $h(t)$ and $c_1(t)$ for the following input pattern, using a software simulation program of your choice (suggested options: python or MATLAB):

- Three ON bursts at $t = 0.40, 0.55, 0.70$ s,
- Three OFF bursts at $t = 1.10, 1.25, 1.40$ s.

Plot:

- $h(t)$: showing three positive pulses (ON bursts) followed by three negative pulses (OFF bursts),
- $c_1(t)$: representing the resulting eye-position signal.

This exercise demonstrates how the network maintains a **memory of eye position** in the absence of ongoing input. An overview of this process is shown in Fig. 5, and your results should qualitatively resemble the figure below.

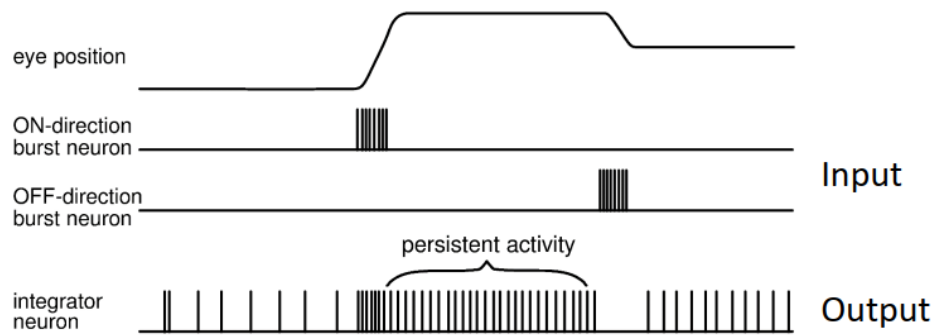


Figure 5: Overview of the eye-position integrator.

Question 4: Nonlinear Recurrent Networks [22 pts]

You shouldn't upload this question yet the deadline of this question will be announced later.