CUED - Engineering Tripos Part IIB 2024-2025 **Module Coursework**

Module	4G3	Title of report	4G3 Coursework 1
Date submitted: 28 Feb		Assessment for this module is ☑ 100% / ☐ 25% coursework of which this assignment forms 50 %	

UNDERGRADUATE and POST GRADUATE STUDENTS		
Candidate number:	5641G	☑ Undergraduate ☐ Post graduate

Feedback to the student ☐ **See also comments in the text**		Very good	**Good**	Needs improvmt
CONTENT	**Completeness, quantity of content:** Has the report covered all aspects of the lab? Has the analysis been carried out thoroughly?			
	Correctness, quality of content Is the data correct? Is the analysis of the data correct? Are the conclusions correct?			
	Depth of understanding, quality of discussion Does the report show a good technical understanding? Have all the relevant conclusions been drawn?			
	Comments:			
PRESENTATION	**Attention to detail, typesetting and typographical errors** Is the report free of typographical errors? Are the figures/tables/references presented professionally?			
	Comments:			

Marker: Date:

4G3 Computational Neuroscience Coursework 1

Candidate: 5641G

Declaration This report were developed with the assistance of ChatGPT for code implementation and report text organization. All the prompt information used is provided in the Appendix.

Question I: Reinforcement Learning

1. Lookup Table TD as a Special Case Starting from the function approximation TD update:

$$w_k \leftarrow w_k + \epsilon\Big[r_t + \gamma \hat{V}_\pi(s_{t+1}) - \hat{V}_\pi(s_t)\Big]\frac{\partial f(s_t;\mathbf{w})}{\partial w_k}, \tag{1}$$

if we set $f(s;\mathbf{w}) = w_s$ (i.e. each state s has its own parameter), then

$$\hat{V}_\pi(s_t = k) = f(s_t = k;\mathbf{w}) = w_k, \quad \frac{\partial f(s;\mathbf{w})}{\partial w_k} = \delta_{s,k} \tag{2}$$

Substituting these into the function approximation TD update (equation 1) gives:

$$w_k \leftarrow w_k + \epsilon\Big[r_t + \gamma \hat{V}_\pi(s_{t+1}) - \hat{V}_\pi(s_t)\Big]\delta_{s_t,k}.$$

Since $\delta_{s_t,k} = 1$ only when $s_t = k$, the update simplifies to:

$$w_k \leftarrow w_k + \epsilon\Big[r_t + \gamma \hat{V}_\pi(s_{t+1}) - w_k\Big],$$

which is exactly the lookup table TD update. The parameter w_k represents the value estimate for state k.

2. Linear Function Approximation When using linear function approximation, we define the value estimate as $\hat{V}_\pi(s) = f(s;\mathbf{w}) = \sum_k w_k \phi_k(s)$, where $\phi_k(s)$ are feature detectors. The gradient of the function with respect to w_k is $\frac{\partial f(s;\mathbf{w})}{\partial w_k} = \phi_k(s)$. Substituting these into equation 1 yields the new update rule:

$$w_k \leftarrow w_k + \epsilon\left[r_t + \gamma \sum_j w_j \phi_j(s_{t+1}) - \sum_j w_j \phi_j(s_t)\right]\phi_k(s_t). \tag{3}$$

3. Tapped Delay Line Representation We implement the tapped delay line representation using the linear function approximation update from Equation 3. The stimulus $y(t)$ is defined as a spike at 10 s.
The state $\mathbf{S}$ is represented as a binary 1-by-25 vector corresponding to a memory span of 12 s with a time step of $\Delta t = 0.5$ s (plus the current step). Defining the index $\tau = \frac{k-1}{2}, \quad k \in \{1,\ldots,25\}$, the τ element of state vector is given by

$$\phi_\tau(S_\tau) = S_\tau(t) = y(t-\tau) = \begin{cases} 1, & \text{if } t = \tau + 10 = \frac{k-1}{2} + 10, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The reward is smoothed around 20 s:

$$r(t) = \frac{1}{2}\exp\Big(-\frac{(t-20)^2}{2}\Big). \tag{5}$$

Since only one element of $\mathbf{S}(t)$ is nonzero (specifically, when $k = 2t - 19$), the value estimate is computed as

$$\hat{V}(t) = \sum_{k=1}^{25} w_k\, S_{\frac{k-1}{2}}(t) = \begin{cases} w_{2t-19}, & \text{if } 10 \le t \le 22, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

where w_k represents the estimated value $\hat{V}(t)$ at time $t = \frac{k-1}{2} + 10$.

Because $S_{\frac{k-1}{2}}(t) = 1$ only when $k = 2t - 19$, only the corresponding weight is updated at time t. Accordingly, the TD update from Equation 3 becomes

$$w_k \leftarrow w_k + \epsilon \left(r\left(t = 9 + \frac{k}{2}\right) + w_k - w_{k-1} \right), \tag{7}$$

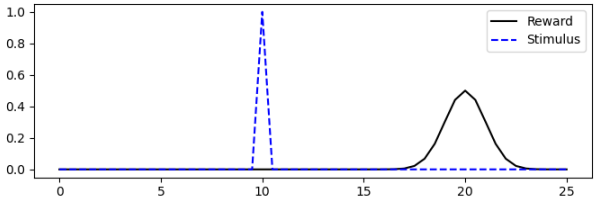


Figure 1: Stimulus and reward profiles.

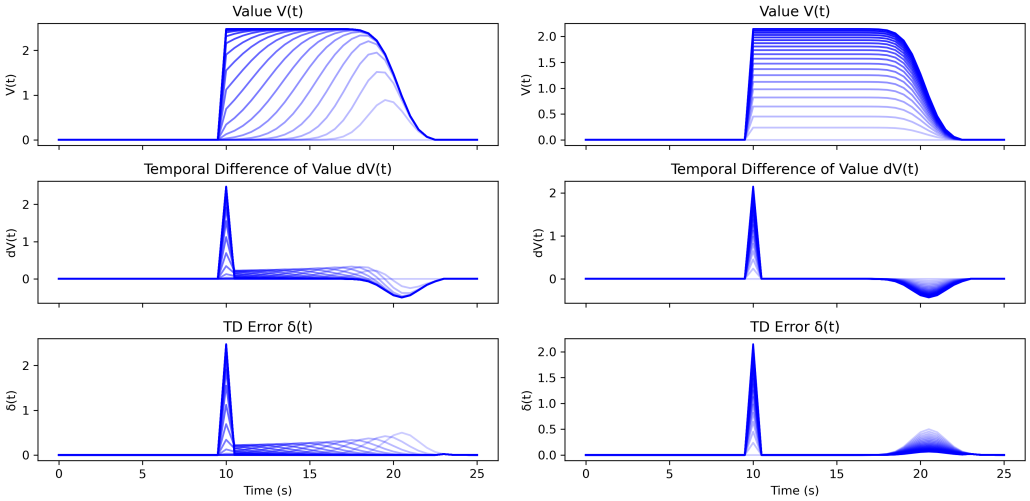
Figure 1 shows the stimulus and reward profiles, and Figure 2a illustrates the evolution of key variables $\hat{V}(t)$, the temporal difference

$$\Delta\hat{V}(t) = \gamma\,\hat{V}(t) - \hat{V}(t - \Delta t), \tag{8}$$

and the TD error

$$\delta(t) = r(t - \Delta t) + \Delta\hat{V}(t), \tag{9}$$

recorded for every 10th trial out of $N = 201$ simulated trials.



(a) Tapped delay line.

(b) Box-car.

Figure 2: (a) Stimulus and reward profiles, (b) Key variables from the tapped delay line implementation, and (c) Value estimates, TD differences, and TD errors for the boxcar representation. The more transparent lines indicate early learning trials, while the solid blue line represents trials at the final stage of learning.

As learning progresses, $\hat{V}(t)$ gradually evolves from mimicking the instantaneous reward (transparent lines in top panel, Figure 2a) to representing the expected sum of future rewards (solid blue line). Notably, during an intermediate stage of learning the value between the stimulus and the reward increases with time, even though it should remain constant. This behavior arises because the feature detectors $\mathbf{S}$ are delta functions (see Equation 4). In particular, at time t the temporal difference becomes

$$\Delta\hat{V}(t) = w_{2t-19} - w_{2t-20}, \tag{10}$$

and the TD update depends solely on $r(t - 0.5)$ and $\Delta\hat{V}(t)$. Consequently, $\Delta\hat{V}(t)$ must be non zero between two events and the proper propagation of the value estimation from the reward time is gradual.

In the TD model, the TD error $\delta(t)$ is assumed to reflect to dopamine activity. However, the tapped delay line model fails to replicate the observation that *no dopamine activation is usually seen between the reward and stimulus times.*

4. Boxcar Representation In the boxcar representation, the feature detectors are defined as the cumulative sum over the past history:

$$\phi_\tau(\mathbf{S}(t)) = \sum_{u=0}^{\tau} S_u(t) = \begin{cases} 1, & \text{if } 10 \le t \le 10+\tau, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the temporal difference in the value estimate becomes

$$\Delta\hat{V}(t) = \mathbf{w}^\top \Big[\mathbf{\Phi}(t) - \mathbf{\Phi}(t-0.5)\Big] = \begin{cases} \mathbf{w}^\top \mathbf{1}, & \text{if } t = 10, \\ -w_\tau, & \text{if } t = 10.5+\tau, \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

so that with the boxcar representation the first and last trials yield results identical to the tapped delay line representation. However, in the intervening trials the TD error $\delta(t)$ remains zero between the stimulus and reward, and the value estimate remains constant during these events (see Figure 2(c)). According to Equation 11, the parameters represent ΔV rather than V. Therefore, the parameters only need to mimic the sharp profile of the reward while maintaining $\Delta V = 0$ between the stimulus and reward. This result closely relates to the observation that *no dopamine activation is usually seen between the reward and stimulus times.*
The learning is adjusted because in the tapped delay line representation the state $\mathbf{S}$ is a delta function so that only one parameter is updated per trial—and only when w_{k-1} has been sufficiently activated. In contrast, with the boxcar representation the step-function features lead to updates across many time steps from the very first trial. Therefore, to prevent overly large updates per trial and ensure learning stability, a smaller learning rate (e.g., $\epsilon = 0.01$) is required.

5. Partial Reinforcement We simulate 1000 trials with partial reinforcement, where only a fraction $p = 0.5$ of trials deliver a reward, using the boxcar representation. For the final 100 trials, we compute the average time courses of three key variables separately for rewarded trials, unrewarded trials, and across all trials. The results are shown in Figure 3a. We find that:

- $\hat{V}(t)$ and $\Delta\hat{V}(t)$ remain essentially identical across trial types because they reflect the expectation of future rewards, which depends primarily on the presence of the stimulus.
- In contrast, $\delta(t)$ varies markedly with the reward outcome: in unrewarded trials, $\delta(t)$ follows $\Delta\hat{V}(t)$, whereas in rewarded trials $\delta(t)$ displays an upward bump due to the addition of the reward $r(t)$. Overall, at the reward time the TD error is nearly canceled out.

Theoretically, let $R = \int_0^\infty r(t)\,dt$ be the total reward. Then, the estimated value is given by

$$\hat{V}(t) = \begin{cases} p\left(R - \int_0^t r(t')\,dt'\right), & t \ge 10\,\text{s}, \\ 0, & t < 10\,\text{s}. \end{cases} \tag{12}$$

The temporal difference is

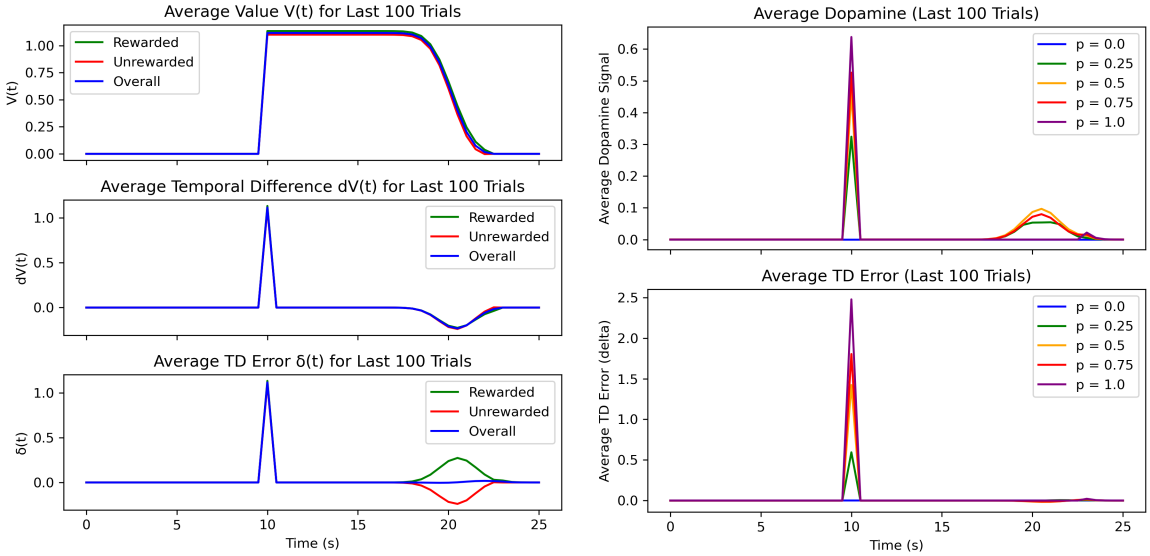
$$\Delta\hat{V}(t) = -pr(t) + \delta_D(t-10)\,pR, \tag{13}$$

where $\delta_D(t-10)$ denotes a delta function at the stimulus. Thus, the TD error becomes

$$\delta(t) = \Delta\hat{V}(t) + r(t) = \begin{cases} \delta_D(t-10)\,pR + (1-p)r(t), & \text{if reward is present,} \\ \delta_D(t-10)\,pR - pr(t), & \text{if reward is absent.} \end{cases} \tag{14}$$

and the overall TD error $\mathbb{E}[\delta(t)]$ is nearly zero at the reward time.

- The peak dopamine at the stimulus increases monotonically with p.
- At the reward time, the dopamine response exhibits a bump that is maximal at $p = 0.5$; for $p = 0$ and $p = 1$ the response is minimal, and for $p = 0.25$ or $p = 0.75$ the bump is modest.



(a) Averages of $\hat{V}(t)$, $\Delta\hat{V}(t)$, and $\delta(t)$ (b) Average dopamine for various p

Figure 3: (a) Averages of $\hat{V}(t)$, $\Delta\hat{V}(t)$, and $\delta(t)$ for the last 100 trials, computed separately for rewarded and unrewarded trials as well as all trials; (b) the average dopamine time course for various reward probabilities;

We then simulate the dopamine (DA) signal by applying a nonlinear function $\mathrm{DA}(x)$ to $x = \delta(t)$. Figure 4 shows that at 10 s the TD error is strongly positive but gets compressed due to saturation, while at 20 s a modest bump emerges because of the asymmetric processing of positive and negative errors.

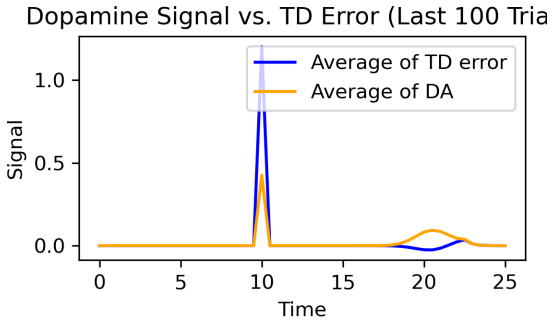


Figure 4: Simulated dopamine signal computed via the nonlinear function.

6. Dopamine Signal for Different Reward Probabilities We conducted similar simulations for reward probabilities $p \in \{0, 0.25, 0.5, 0.75, 1.0\}$. Figure 3b displays the average dopamine time course (over the last 100 trials) for each p, with each probability represented by a different color. We observe that: Although the overall shape of the dopamine signal remains similar across these conditions, the peak dopamine at the stimulus increases with p. In contrast, at the reward time the dopamine response exhibits a bump that is maximal at p = 0.5; for p = 0 or p = 1 there is little to no dopamine response, and for p = 0.25 or p = 0.75 the bump is modest.

7. Peak Dopamine at Stimulus and Reward We further examined the peak dopamine levels as a function of p:

- **At the stimulus:** The dopamine signal is $\mathrm{DA}_{\mathrm{stim}} = \mathrm{DA}(pR)$, so the peak DA increases monotonically with p.
- **At the reward:** The average dopamine response is given by

$$\mathrm{DA}_{\mathrm{reward}} = p\,\mathrm{DA}\big((1-p)r(t)\big) + (1-p)\,\mathrm{DA}\big(-p\,r(t)\big).$$

Under saturation constraints, this reduces approximately to

$$\mathrm{DA}_{\mathrm{reward}} \approx \min\{x^*, (1-p)r(t)\}\,p,$$

implying that the peak DA at the reward increases with p up to a maximum near $p = 0.5$ and then decreases.

Figure 5 illustrates these relationships.

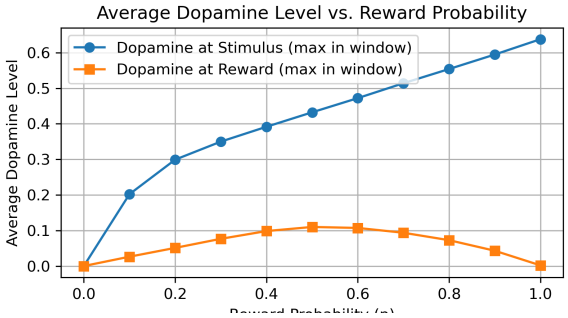


Figure 5: Peak dopamine at the stimulus and reward vs reward probability p.

8. Interpretation of the Averaged Dopamine Signal The DA signal exhibits a pattern—first increasing and then decreasing with p—that appears to reflect uncertainty. In our model, this behavior is related to the binary variance $p(1-p)$ in the reward distribution, and thus it can represent uncertainty. It is important to note that for smaller values of p, the dopamine response is additionally constrained by the saturation parameter x^* and hence remains more linear.
However, because the dopamine response to positive prediction errors (i.e., when a reward is delivered) is much stronger than the response to negative errors (when a reward is omitted), the positive signal is not completely canceled out when averaging over multiple trials. This imbalance results in an apparent ramping (gradual increase) of the DA signal over time and might look like encoding uncertainty, even though it is fundamentally driven by prediction errors.

Question II: Representational Learning

1. Empirical Estimation of $p(x_k)$ We consider a dataset of N image patches, each represented by a D-dimensional row vector $\mathbf{y}_n^\top \in \mathbb{R}^D$. Arranging these vectors into the matrix

$$\mathbf{Y} = \begin{pmatrix} \mathbf{y}_1^\top \\ \mathbf{y}_2^\top \\ \vdots \\ \mathbf{y}_N^\top \end{pmatrix} \in \mathbb{R}^{N\times D},$$

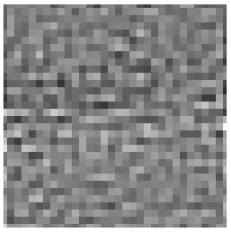
we define a feed-forward weight matrix $\mathbf{R} \in \mathbb{R}^{D\times K}$ and a generative weight matrix $\mathbf{W} \in \mathbb{R}^{D\times K}$, where K is the number of latent components.
The k-th latent variable of the n-th image is computed as

$$x_{n,k} = \mathbf{y}_n^\top \mathbf{r}_k,$$

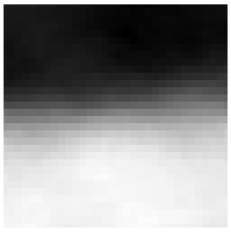
where $\mathbf{r}_k$ is the k-th column of $\mathbf{R}$. Collecting all latent representations into

$$\mathbf{X} = \mathbf{Y}\,\mathbf{R} \in \mathbb{R}^{N\times K},$$

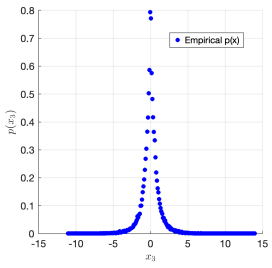
the n-th row of $\mathbf{X}$ is $\mathbf{x}_n^\top = [x_{n,1}, x_{n,2}, \ldots, x_{n,K}]$, and the k-th column provides an empirical basis to estimate $p(x_k)$.



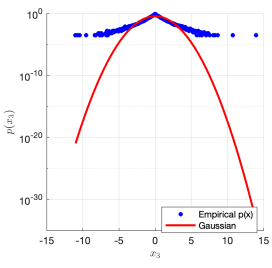
(a) $R(:,3)$



(b) $W(:,3)$

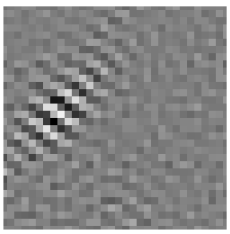


(c) Histogram (linear)

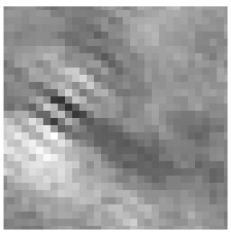


(d) Histogram (log)

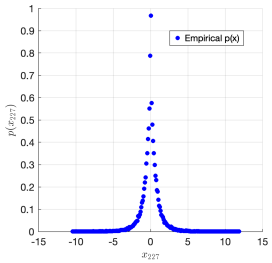
Figure 6: Empirical results for latent variable $k = 3$.



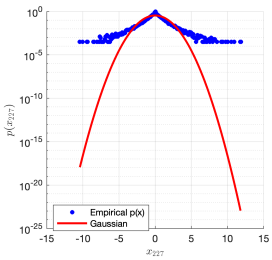
(a) $R(:, 227)$



(b) $W(:, 227)$



(c) Histogram (linear)

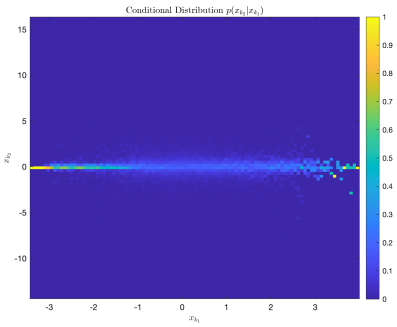


(d) Histogram (log)

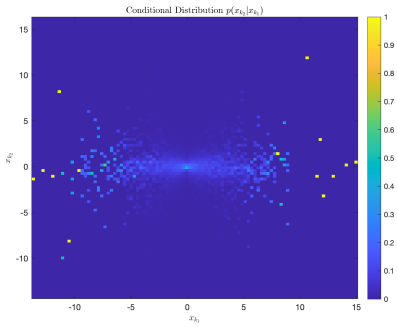
Figure 7: Empirical results for latent variable $k = 227$.

Figures 6 and 7 show, for selected $k \in \{3, 227\}$, the feed-forward weights $R(:, k)$ and generative weights $W(:, k)$ (left subfigures) along with the corresponding histograms of $p(x_k)$ on linear and logarithmic scales (right subfigures). A Gaussian is overlaid for comparison. It can be observed that, for components with different characteristic frequencies, the resulting distributions are sparse.

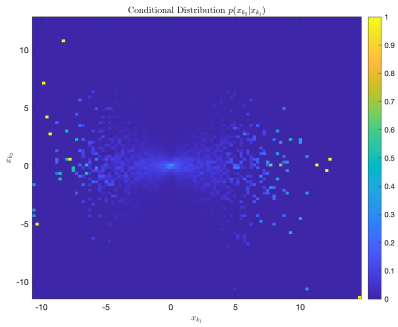
We then examined the independence between pairs of latent components. Figures 8a–8c show conditional histograms, where the horizontal axis corresponds to x_{k_1} (the conditioning variable) and the color represents the distribution of x_{k_2}. If x_{k_1} and x_{k_2} were independent, the color distribution would remain invariant with respect to x_{k_1}. Instead, we observe that as $|x_{k_1}|$ increases, the standard deviation of x_{k_2} increases, indicating that x_{k_1} modulates the uncertainty of x_{k_2} and suggesting dependency. Notably, components 1 and 51 show less modulation, implying higher independence.



(a) $p(x_{51} \mid x_1)$



(b) $p(x_{51} \mid x_{50})$



(c) $p(x_{109} \mid x_{108})$

Figure 8: Conditional histograms of latent variables.

2. Conditional Variance Model and Parameter Estimation For a dataset $\mathbf{X}$ with N samples and K components, the conditional log-likelihood is

$$\mathcal{L}(\mathbf{X}; \theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \Big[-\frac{1}{2} \log(2\pi) - \frac{1}{2} \log \sigma_{n,k}^2 - \frac{x_{n,k}^2}{2\sigma_{n,k}^2} \Big], \tag{15}$$

where $\sigma_{n,k}^2 = \sigma_k^2(\mathbf{x}_{n,\neq k}; \theta)$.

Differentiation yields, for $j \neq k$,

$$\frac{\partial \mathcal{L}}{\partial a_{k,j}} = \sum_{n=1}^{N} \frac{x_{n,j}^2}{2} \left(\frac{x_{n,k}^2}{(\sigma_{n,k}^2)^2} - \frac{1}{\sigma_{n,k}^2} \right), \tag{16}$$

and for b_k,

$$\frac{\partial \mathcal{L}}{\partial b_k} = \sum_{n=1}^{N} \frac{1}{2} \left(\frac{x_{n,k}^2}{(\sigma_{n,k}^2)^2} - \frac{1}{\sigma_{n,k}^2} \right). \tag{17}$$

Optimizing the log-parameters gives

$$\frac{\partial \mathcal{L}}{\partial \log a_{k,j}} = a_{k,j} \frac{\partial \mathcal{L}}{\partial a_{k,j}}, \quad \frac{\partial \mathcal{L}}{\partial \log b_k} = b_k \frac{\partial \mathcal{L}}{\partial b_k}. \tag{18}$$

For computational efficiency, define

$$\mathbf{\Sigma} = \mathbf{X}^{\odot 2}\mathbf{A}^\top + \mathbf{1}_N\,\mathbf{b}^\top,\ \mathbf{U} = \frac{1}{2}\Big(\mathbf{X}^{\odot 2} - \mathbf{\Sigma}\Big) \odot \mathbf{\Sigma}^{\odot -2}.$$

Then the loss becomes

$$\text{Loss} = \frac{NK}{2}\log(2\pi) + \frac{1}{2}\mathbf{1}_N^\top \log(\mathbf{\Sigma})\,\mathbf{1}_K + \frac{1}{2}\sum_{n,k}\frac{x_{n,k}^2}{\sigma_{n,k}^2}, \tag{19}$$

with gradients

$$\nabla_{\log(\mathbf{A})}\text{Loss} = -\Big(\mathbf{U}^\top\mathbf{X}^{\odot 2}\Big) \odot \mathbf{A}, \tag{20}$$

$$\nabla_{\log(\mathbf{b})}\text{Loss} = -\Big(\mathbf{U}^\top\mathbf{1}_N\Big) \odot \mathbf{b}. \tag{21}$$

The gradients were verified using the `checkgrad` function on a small subset of the data (e.g., $X_{\text{subset}} = X(1{:}10, 1{:}50)$), yielding a relative error of approximately 2.63×10^{-7}.
Subsequently, the model was trained on 80% of the dataset (25,600 images). Figure 9 shows the loss function value returned by the `minimize` function over iterations, verifying convergence of the parameter estimation.

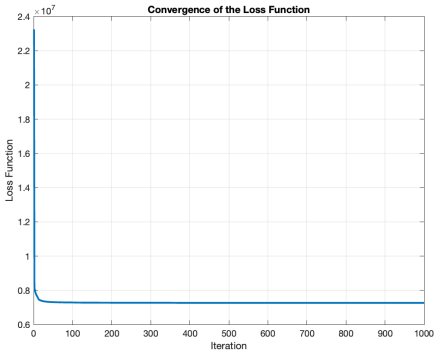


Figure 9: Convergence of the loss function during parameter estimation.

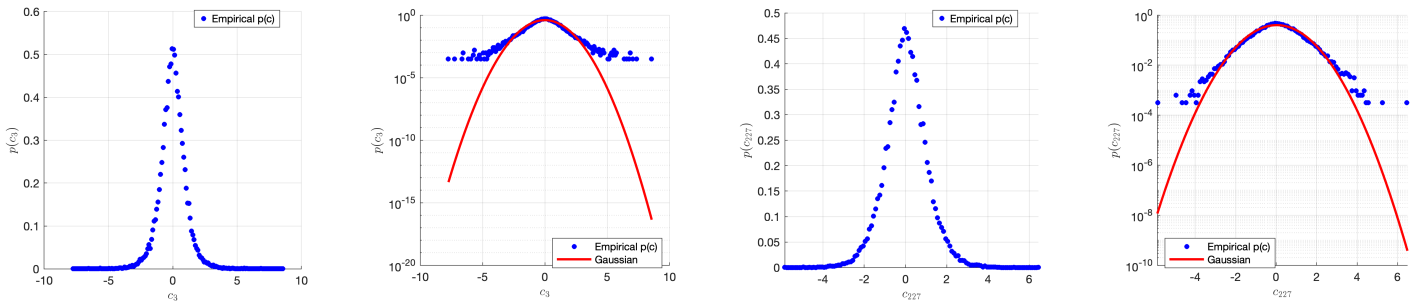
3. Normalized Variables After training, we define the normalized latent variables as

$$c_k = \frac{x_k}{\sigma_k(x_{\setminus k};\theta)}. \tag{22}$$

Excess kurtosis is defined by

$$\text{Excess Kurtosis} = \kappa - 3,$$

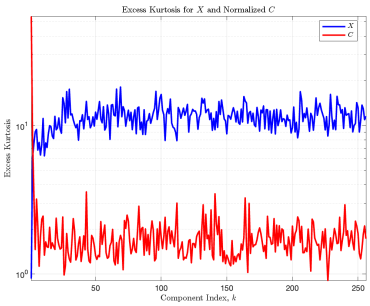
where κ is the kurtosis. Figures 10a–10d display the empirical histograms of $p(c_k)$ for $k = 1$, 3, 30, and 227 on both linear and logarithmic scales. Figure 11a shows that $p(c_k)$ has excess kurtosis closer to 0 than $p(x_k)$, indicating a more Gaussian, less sparse distribution.



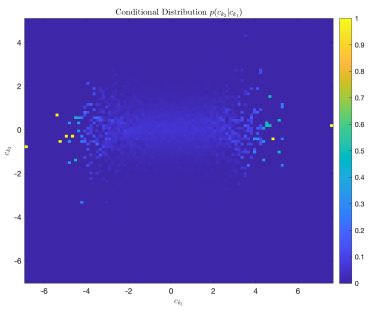
(a) $p(c_3)$ (linear) (b) $p(c_3)$ (log) (c) $p(c_{227})$ (linear) (d) $p(c_{227})$ (log)

Figure 10: Normalized marginal distributions.

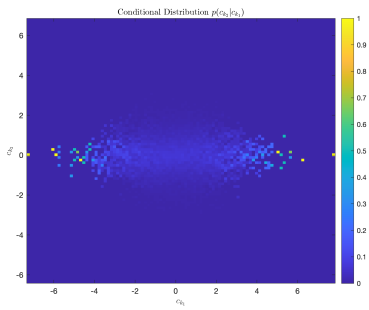
Finally, Figures 11b and 11c compare the conditional distributions $p(c_{k_2} \mid c_{k_1})$ and $p(x_{k_2} \mid x_{k_1})$. The normalized variables c_k exhibit greater independence, consistent with the model assumption $c_k \sim \mathcal{N}(0,1)$.



(a) Excess kurtosis comparison between $p(x_k)$ and $p(c_k)$.



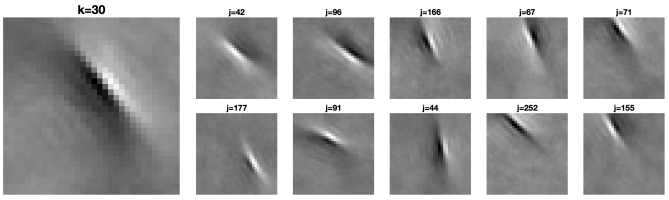
(b) $p(c_{51} \mid c_{50})$



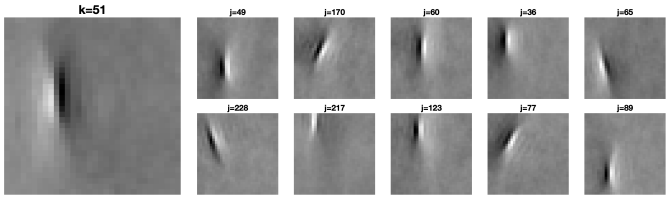
(c) $p(c_{109} \mid c_{108})$

Figure 11: Excess kurtosis comparison and Conditional distributions of normalized latent variables.

4. Correlated Generative Weights The parameter $a_{k,j}$ represents the dependency of component k on component j. By plotting the generative weights with `plotIm(W(:,k))` (see Figures 12a and 12b), we observe that components with larger $a_{k,j}$ values tend to have similar orientations and spatial locations as component k (sometimes with reversed edge polarity). This suggests that natural images exhibit local statistical dependencies, where components co-occurring in the same region are coordinated in position and orientation, often forming continuous edges.



(a) $k = 30$



(b) $k = 51$

Figure 12: Combined generative weights (top 10) for components $k = 30$ and $k = 51$.

5. V1 Data Interpretation In our computational framework, we identify the response of neuron k with the normalized variable

$$c_k = \frac{x_k}{\sigma_k}, \tag{23}$$

where x_k represents the signal strength of the kth component and σ_k is the normalization factor.
The computational model indicates that contrast primarily affects the magnitude of x_k. For neurons that are selective for a particular orientation and location (corresponding to component k), **the neural response is given by** c_k. When a surround masker is presented, neurons surrounding the k-selective neuron increase their activity. Although the raw signal x_k remains largely unaltered, the normalization factor σ_k increases because the response of nearby, highly correlated neurons (those with similar edge features) is enhanced. As a result, the response c_k decreases, leading to a diminished neural response for the k-selective neuron.
In contrast, for an *orthogonal mask* the coupling parameter $a_{k,j}$ is very small between k-selective neuron and neurons selective for orthogonal orientations. Thus, when an orthogonal mask is applied, even if surrounding neural responses increase due to enhanced contrast, both the normalization σ_k and raw siganl strength x_k are unchanged ,leaving the response c_k largely unchanged.
In summary, the model predicts that surround parallel masking decreases the neural response by increasing the normalization factor σ_k (while leaving x_k relatively unaltered), whereas orthogonal masking exerts minimal influence on the response because the corresponding coupling parameters $a_{k,j}$ are small. This computational interpretation is consistent with the divisive normalization observed in the neural processing of natural images.

Appendix

Code

All code used in this report is available at: `https://github.com/derek1909/4G3-CW1.git`

ChatGPT Conversation Log

The ChatGPT conversation log related to this assignment can be found at: `https://chatgpt.com/share/67c1dbf1-3e60-800d-bf80-b0b3b8bcb817`