Bayesian Computation and Optimal Decision Making in Primate Brains

Yanping Huang

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2015

Reading Committee:

Rajesh P.N. Rao, Chair

Eric Shea-Brown

Adrienne L. Fairhall

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Bayesian Computation and Optimal Decision Making in Primate Brains

Yanping Huang

Chair of the Supervisory Committee:
Dr. Rajesh P.N. Rao
Computer Science and Engineering

This dissertation investigates the computational principles underlying the brains' remarkable capacity to perceive, learn and act in environments of constantly varying uncertainty. Bayesian probability theory has suggested that optimal perception, learning and action rely on computing probability distributions over task-relevant world variables.This suggests the nervous system may maintain internal probabilistic generative models for what caused its sensory input. In this dissertation, we examine many aspects of primate perceptual and motor behaviors and model them under the framework of Bayesian inference and optimality principle.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number Page

ACKNOWLEDGMENTS

First and for most, I would like to express my sincerest appreciation to my advisor, Dr. Rajesh Rao. He has constantly helped me during the whole process of our research, from the big picture to the small details. Without his incisive comments, wise direction and kind encouragement, the completion of this dissertation would not have been possible. More importantly, Dr. Rajesh Rao has taught me to approach economics as a science, always being precise and clear about my hypotheses and methods. I am sure I will benefit so much from this in the future.

I would also like to give my thanks to Dr. Adrienne Fairhall, Dr. Eric Shea-Brown, Dr. Luke Zuttlemore and Radha Poovendran for enhancing this dissertation, offering valuable suggestions, and serving on my committee.

In addition, I would like to express my love to my parents, Yongquan Huang and Shugui Guo, without whom none of this would have been possible. They have given me so much and every single day I feel blessed to be their daughter. There is nothing in the world I would trade for parents like mine.

DEDICATION

To my beloved parents, Yongquan Huang and Shugui Guo for their everlasting love, trust and encouragement. . .

Chapter 1

INTRODUCTION

The main goal of the present study is to understand the mechanisms in the brain that enable humans and other mammals to select and achieve non-immediate goals and thereby survive. Every day, mammals, including humans, must make countless decisions both large and small. In particular, they need to make choices that lead to better long-term outcomes in order to survive. For one example, many species of mammals gather and store food for the winter instead of eating it right away. For another, some mammal species migrate seasonally thousands of miles to a better place to find food and breed. The benefits of long-distance migration outweigh its cost and the benefits of shorter distances. These mammals are therefore willing to take some short-term loss in order to gain more long term goals.

So, in order to survive, mammals need to make decisions oriented toward long-term outcomes. This requires them to compute expected long-term outcomes and compare outcomes for different course of action. It follows that in order to compute any expected future outcome, they should able to make predictions based on past experience.

Two major challenges confront us in attempting to understand how mammals make decisions:

- Animals are not computers. They cannot perform arithmetical computations. But they do have neurons, literally astronomical numbers of neurons. How do they make use of those myriad neurons to compute information (in the broader sense) and make decisions? Multiple factors enter animal decision making: current sensory perception, perceptual filtering and sorting mechanisms, prior experience, which are rules or generalizations derived from this experience, situation analysis, predictions about the future. How do animal brains, specifically primate brains, synthesize these factors to make a decision?

- The world is stochastic and ambiguous, and classificatory schemes and rules, whether neuronally hardwired or developed from experience, are never precise maps of reality. Is that dress blue and purple, or aqua and magenta? How do we determine where we are or have already been in a big maze? And especially for humans, much of the data needed for prediction and decision making is second-hand or third-hand, summarized and organized by others: where is the enemy in the (metaphorical) fog of war? Even our eyes are unreliable sensors. Given that the information collected through sensory organs is likewise stochastic and ambiguous, how do brains and the neuronal networks perform estimation and prediction? For example, when we play tennis, how do we estimate and predict the trajectory of the incoming ball and make the best return stroke?

Given those challenges, the present dissertation offers some explanations and theories. The main contributions can be summarized as follows:

First, we propose a process whereby mammals compute estimations and make predictions about the future based on "noisy" experience in neural networks. In our model, mammals don't need to know anything about the situation beforehand. They can merely observe the unreliable flow of sensory data; our proposed neural network is able to perform the estimation and make the predictions based on this information. Our model is novel because the solutions suggested in the literature have two main sorts of shortcomings: they either assume the observations from sensors are reliable and accurate, or else they assume the mammals already know the dynamics of the world. Our model requires neither assumption.

Second, we propose a decision-making framework that optimally combines three principal factors: the prior (existing weights and connections in related neural networks, whether inborn or modified through experience); current evidence (the observation); and the predicted reward for the actions in the short term. Combining these three factors, our proposed framework will be able to compute the action that leads to maximum long-term reward. The novelty here is that we propose this framework from first principles. In addition, we can explain many behavioral experiments on mammals using this framework.

To resume: mammals are routinely faced with the problem of estimating unknown world-states from ambiguous and noisy stimuli. Moreover, the world is dynamic, putting a premium on the ability to actively explain and predict upcoming events by learning the temporal dynamics of relevant states of the world. Mathematically, this suggests the brain's ability to perform both Bayesian inference and learning for hidden Markov models.

In chapter 2, we present results that illustrate the ability of the model to explain neurobiological data such as history-dependent adaptation to light intensity in visual neurons and changes in the receptive fields of hippocampal place cells after learning. We explore a novel neural implementation of HMMs in networks of spiking neurons that perform approximate Bayesian inference similar to the Monte Carlo method of particle filtering. We propose that a recurring neuronal network with synaptic plasticity can implement a form of Bayesian inference similar to Monte Carlo methods such as particle filtering. In this process, a lower layer of sensory neurons receives noisy measurements of hidden world states. A higher layer of neurons then infers a posterior distribution over world states via Bayesian inference from inputs generated by sensory neurons. Each spike in the population of inference neurons represents a sample of a particular hidden world state. The spiking activity across the neural population approximates the posterior distribution of hidden states. We show how the time-varying posterior distribution $P(Xt|Z_{1:t})$ can be directly represented by mean spike counts in populations of Poisson neurons. In addition, we demonstrate how a spike-timing based Hebbian learning rule in our network can implement an online version of the Expectation-Maximization(EM) algorithm to learn the emission and transition matrices of HMMs. The major novelty is that the stochasticity of synaptic transmission is directly involved in the implementation of stochasticity necessary for Monte Carlo sampling. And this work was published in Advances in Neural Information Processing Systems 27 (NIPS 2014) [82].

As earlier noted, a key problem in neuroscience is understanding how the brain makes decisions under uncertainty. Important insights have been gained using tasks such as the "random dots motion" discrimination task, in which the subject makes decisions based on noisy stimuli. A descriptive model known as the drift diffusion model has previously been used to explain psychometric and reaction-time data from such tasks. But to fully explain the data, one is forced to make

ad-hoc assumptions such as a time-dependent collapsing decision boundary. In Chapter 3, we show that such assumptions are unnecessary when decision making is viewed within the framework of partially observable Markov decision processes (POMDPs). We propose an alternative model for decision making based on POMDPs. Specifically, we show that the motion-discrimination task can be reduced to the problems of

1. Computing beliefs (posterior distributions) over unknown direction and motion strength from noisy observations in a Bayesian manner.

2. Selecting actions based on these beliefs to maximize the expected sum of future rewards.

The resulting optimal policy (belief-to-action mapping) is shown to be equivalent to a collapsing decision threshold that governs the switch from evidence accumulation to a discrimination decision. We show that the model accounts for both accuracy and reaction time as functions of stimulus strength as well as different speed-accuracy conditions in the "random dots" task. This piece of work was published in Plos One 2013 [81].

It is challenging to make decisions to achieve distant goals. Yet mammalian brains are extremely adept at performing this task. How does the mammalian brain combine prior knowledge with sensory evidence when making decisions under uncertainty? Two competing descriptive models based on experimental data have been proposed. The first posits an additive offset to a decision variable, implying a static effect of the prior. However, this model is inconsistent with recent data from a motion discrimination task involving temporal integration of uncertain sensory evidence. To explain this data, a second model has been proposed that assumes the prior's influence varies over time. In Chapter 4 we present a normative model of decision making that incorporates prior knowledge in a principled way. We show that the additive-offset model and the time-varying prior model emerge naturally when decision making is viewed within the framework of partially observable Markov decision processes (POMDPs). Thus, decisions are made so as to maximize cumulative expected rewards over the course of sequential decision making. We show that such a model explains behavioral data from several sensory decision-making tasks in humans and monkeys, while

also providing a normative explanation for otherwise conflicting neurophysiological data from cortical area LIP. Our model describes a novel optimum decision theoretic formulation of evidence accumulation under uncertainty. It explains psychophysical data without invoking extra free parameters in contrast to existing models (using static priors or a time varying prior). This piece of work was published in Advances in Neural Information Processing Systems 25 (NIPS 2012) [80].

Markov decision processes (MDPs) are a well-studied framework for solving sequential decision-making problems under uncertainty. Exact methods for solving MDPs based on dynamic programming, such as policy iteration and value iteration, are effective for small problems. In problems with a large discrete state space or with continuous state spaces, a compact representation is essential for providing an efficient approximation solution to MDPs. Commonly used approximation algorithms involve constructing basis functions for projecting the value function onto a low-dimensional subspace, and building a factored or hierarchical graphical model to decompose the transition and reward functions. However, hand-coding a good compact representation for a given reinforcement learning (RL) task can be quite difficult and time-consuming. Recent approaches have attempted to automatically discover efficient representations for RL. In chapter 5, we discuss the problems of automatically constructing structured kernels for kernel-based RL, a popular approach to learning non-parametric approximations for value function. We explore a space of kernel structures that are built compositionally from base kernels using a context-free grammar. We examine a greedy algorithm for searching over the structure space.

Chapter 2

NEURONS AS MONTE CARLO SAMPLERS: BAYESIAN INFERENCE AND LEARNING IN SPIKING NETWORKS

Abstract

We propose a two-layer recurrent Poisson neuronal network capable of performing both approximate inference and learning for any hidden Markov model. The lower layer sensory neurons receive noisy measurements of hidden world states. The higher layer neurons infer a posterior distribution over world states via Bayesian inference from inputs generated by sensory neurons. We show how such a neuronal network with synaptic plasticity can implement a form of Bayesian inference similar to Monte Carlo methods such as particle filtering. Each spike in the population of inference neurons represents a sample of a particular hidden world state. The spiking activity across the neural population approximates the posterior distribution of hidden state. Uncertainties in spike numbers provide the necessary variability for sampling during inference. Unlike previous models, the hidden world state is not observed by the sensory neurons, and the temporal dynamics of the hidden state is unknown. We demonstrate how the network can learn the likelihood model as well as the transition probabilities underlying the dynamics using a spike-timing dependent Hebbian learning rule. We present results illustrating the ability of the network to perform filtering and learning of arbitrary Hidden Markov Models.

1 Introduction

Animals are routinely faced with the problem of estimating unknown world states from ambiguous and noisy stimuli. For example, when inferring 3D structure from a 2D image, the neural system must choose one among many possible interpretations that are consistent with the projected 2D image. A mouse in a maze must estimate its current location indirectly from noisy sensory evidence

such as whisker deflections, sight, and odor. In such situations, the brain needs to combine noisy sensory information with incomplete knowledge of the environment. Furthermore, the world is dynamic, putting a premium on the ability to actively anticipate upcoming events by learning the temporal dynamics of relevant states of the world. For example, when facing an approaching tennis ball, a player must not only estimate the current position of the ball, but also predict its trajectory by inferring the ball's velocity and acceleration before deciding on the next stroke. The relevant hidden variables (e.g., velocity, acceleration) are not directly observable but must be estimated from retinal images. Tasks such as these can be modeled using a hidden Markov model [122], where the relevant states of the world are latent variables related to sensory observations via a likelihood model (determined by the *emission probability matrix*). The states themselves evolve over time in a Markovian manner, the dynamics being governed by a *transition probability matrix*. In these tasks, the optimal way of combining such noisy sensory information is to use Bayesian inference, where the level of uncertainty for each possible state is represented as a probability distribution [166]. Behavioral and neuropsychophysical experiments [87, 88, 52] have suggested that the brain may indeed maintain such a probablistic representation of the world state and employ Bayesian inference and learning in a great variety of tasks in perception, sensori-motor integration, and sensory adaptation. However, it remains an open question how the brain can sequentially infer the hidden state and learn the environment from the noisy sensory observations.

There have also been several neural models using populations of neurons to represent probability distribution [168, 165, 162, 98]. These models assume a static world state X. To get around this limitation, firing-rate models [124, 8] has been proposed to used responses in populations of neurons to represent the time-varying posterior distributions of arbitrary hidden Markov models with discrete states. For the continuous state space, similar models based on line attractor networks [160] have been proposed for implementing the Kalman filter, which assumes all distributions are Gaussian and the dynamics is linear. More recently, Bobrowski et al. [17] proposed a spiking network model that can compute the optimal posterior distribution in continuous time. The limitation of these models is that model parameters (the emission and transition probabilities) are assumed to be known a priori. Deneve [44, 45] proposed a model for inference and learning based

on the dynamics of a single neuron. However, the number of states in her model is limited to 2.

In this article, we propose a new model of Bayesian computation in networks of Poisson neurons. We show how the time-varying posterior probability distribution for a hidden Markov model can be directly represented by mean spike counts of neurons, without invoking complicated decoding methods. Each spike in the posterior population is viewed as a Monte Carlo sample of a particular world state. The probability that a neuron's membrane potential exceeds spiking threshold is shown to approximate the posterior probability of the preferred state encoded by the neuron. The resulting responses of model neurons exhibit a characteristic property of cortical neurons, namely, that the variance of the spike count is proportional to the mean. In this model, variability in spiking is not regarded as a nuisance but an integral feature that provides the variability necessary for sampling during inference. The model thus provides a concrete neural implementation of sampling ideas previously suggested in [79, 119, 26, 11]. In addition, we demonstrate how a spike-timing based Hebbian learning rule in our network can implement an online version of the Expectation-Maximization(EM) algorithm to learn the emission and transition matrices of HMMs.

2 Neural Network Model

2.1 Review of Hidden Markov Models

We begin by considering a discrete-time `grid-based` hidden Markov process $\{X_k, k = 1, \ldots, \mathbb{K}\}$ such that

$$X_{k+1} \mid (X_k = x') \sim f(x|x'), \quad x, x' \in x^1, \ldots, x^{\mathbb{X}}.$$

where $f(x|x')$ is the transition probability density, $\mathbb{X}$ is the number of states of X_k, $\mathbb{K}$ is the number of time steps, and "$\sim$" denotes distributed according to. The hidden world state X_k could correspond to an attribute of the real world, such as the mean light intensity of the visual stimulus, or the location of a rat in a maze. We assume animals are interested in estimating X_k by constructing its probability mass function, also called the "belief state," based only on noisy measurements or observations $\{Z_k\}$. The $\{Z_k\}$ are assumed to be conditional independent given $\{X_k\}$ and are

governed by a likelihood function g:

$$Z_k \mid (X_k = x) \sim g(z|x), \quad z \in \mathcal{Z}.$$

It is not necessary for the animal to remember the complete history of observations $\{Z_k\}$ to calculate the belief state. Instead, the belief state can be updated sequentially every time the sensory organs receive new measurements. This procedure is called "filtering" in the engineering subjects. From a Bayesian perspective, filtering corresponds to recursively calculating the belief state of X_k given the observation sequence $Z_{1:k}$ up to time k. When both $f(x|x')$ and $g(z|x)$ are given, the posterior pdf $P(X_k|Z_{1:k})$ may be obtained recursively in two steps: a prediction step (Equation 2.1) and a measurement update (or correction) step (Equation 2.2):

$$P(X_{k+1} = x^i \mid Z_{1:k}) = \quad \omega^i_{k+1|k} = \textstyle\sum_{j=1}^{\mathbb{X}} \omega^j_{k|k} f(x^i|x^j), \tag{2.1}$$

$$P(X_{k+1} = x^i \mid Z_{1:k+1}) = \quad \omega^i_{k+1|k+1} = \frac{\omega^i_{k+1|k} g(Z_{k+1}|x^i)}{\sum_{j=1}^{\mathbb{X}} \omega^j_{k+1|k} g(Z_{k+1}|x^j)}. \tag{2.2}$$

The prediction equation (equation 2.1) uses the previous belief state $P(X_k|Z_{1:k})$ from time step k to produce a prior distribution of the state at time $k+1$. When a new measurement Z_{k+1} becomes available, the update equation (equation 2.2) modifies this prior density via Bayes' rule to obtain the posterior distribution $P(X_{k+1}|Z_{1:k+1})$. This process is repeated for each time step. These two recursive equations above are the foundation for any exact or approximate solution to Bayesian filtering, including well-known examples such as Kalman filtering when the original continuous state space is divided into $\mathbb{X}$ bins.

2.2 Network architecture

We now show that the framework of grid-based filtering can be implemented in a two-layer spiking neural network as shown in center panel of Figure 2.1. The lower layer consists of an array of $\mathbb{Z}$ `sensory` neurons, each of which will be atctivated at time step k if some noisy observation Z_k is in the receptive field. The higher layer consists of an array of $\mathbb{X}$ `inference` neurons, each of which is a modulated Poisson neuron [69] whose spike count N^i_k at time step k follows:

$$P(N^i_k|\mu^i_k) = \frac{(\mu^i_k)^{N^i_k}}{N^i_k!} \exp(-\mu^i_k)\mu^i_k = \rho^i(Z_k)\gamma^i_k$$

where $i = 1, \ldots, \mathbb{X}$, and the rate μ_k^i is the product of the drive $\rho^i(Z_k)$, which is some reproducible response to the stimulus, and the gain γ_k^i, which represents stimuli-independent modulations. The inference neurons are connected with sensory neurons via feedforward weight matrix M, and are mutally connected via recurrent weight matrix W. As a result of this architecture, the drive $\rho(Z_k)$ is the sum of feedforward inputs from sensory neurons when stimulus Z_k is observed. The gain γ_k^i represents inference neurons' recurrent inputs, which are determined by the recurrent weight matrix W and $\{N_{k-1}^i\}$ from previous time step. How can Bayesian inference be achieved using the above network architecture? We approach this problem by first showing how this neural network can represent and maintain probability distributions.

Similar to the idea of grid-based filtering, we have each inference neuron representing each of of $\mathbb{X}$ preferred states. A spike, generated by a neuron whose preferred world state is x^i within timestep k, represents an independent Monte Carlo sample (particle) from the posterior probability $P(X_k = x^i|Z_k)$. As depicted in the raster plot of Figure 2.1, the distribution of spikes across the entire inference layer population is a Monte-Carlo approximation to the current posterior distribution. At time k let the population responses in inference layer $\{N_k^i\}$ be proportional to conditional probabilities $P(X_k = x^i|Z_{1:k})$:

$$N_k^i = P(X_k = x^i|Z_{1:k}) \times N_k = \omega_{k|k}^i \times N_k, \tag{2.3}$$

where $N_k = \sum_i N_k^i$. We next show how our network architecture with appropriate chosen synaptic connections ensure that this population responses at subsequent time steps will be still proportional to updated posterior distributions.

Bayesian Inference with Stochastic Synaptic Transmission

To implement the prediction equation 2.1, we require that the recurrent weights between the inference neurons encode the transition probabilities: $W_{ij} = f(x^j|x^i)/C_W$, where C_W is a scaling constant. We define the recurrent weight W_{ij} to be the synaptic release probability between the i-th neuron and the j-th neuron in the inference layer. With the population response follows equa-

tion 2.3, the gain function γ_{k+1}^j for the j inference neuron at time $k+1$ is then:

$$\gamma_{k+1}^j = \sum_i W_{ij} N_k^i = \frac{N_k}{C_W} \sum_i f(x^j \mid x^i) \omega_{k|k}^i = \frac{N_k}{C_W} \omega_{k+1|k}^i \tag{2.4}$$

Thus, the prediction probability in equation 2.1 is encoded as the gain modulation in equation 2.4.

The noisy measurement Z_{k+1} is not directly observed by the inference neurons, but sensed through an array of $\mathcal{Z}$ sensory neurons, whose receptive fields are centered at $z^i \in \mathbb{Z}, i = 1, \ldots, \mathcal{Z}$. We assume for simplicity that receptive fields of sensory neurons do not overlap with each other (In appendix we will discuss the more general overlapping case). Again we define the feedforward weight M_{ij} to be the synaptic release probability between i-th sensory neuron i and j-th inference neuron. Let $M_{ij} = g(z^i|x^j)/C_M$, we have the drive $\rho^j(Z_{k+1}) = \frac{g(Z_{k+1}=z^i|x^j)}{C_M}$ and firing rate for the j-th neuron at time $k+1$:

$$\begin{aligned} \mu_{k+1}^j &= \frac{1}{C_W C_M} g(Z_{k+1}|x^j) \sum_i f(x^j \mid x^i) N_k^i \\ &= \frac{P(Z_{k+1}|Z_{1:k})}{C_W C_M} \omega_{k+1|k+1}^j N_k \end{aligned} \tag{2.5}$$

Let N_{k+1}^j be the number of spikes in j-th sub-population at time $k+1$, its mean and variance follows:

$$E[N_{k+1}^j|\{N_k^i\}] = \mathrm{Var}[N_{k+1}^j|\{N_k^i\}] = \mu_{k+1}^j$$

Equation 2.5 ensures that the expected response distribution at time $k+1$ is a Monte Carlo approximation to the updated posterior probability $P(X_{k+1}|Z_{1:k+1})$.

2.3 Convergence results

In this section, we briefly discuss some convergence results for Bayesian filtering using the proposed Poisson network. Let $\hat{P}_k^j = \frac{N_k^j}{N_k} \mid N_k$ be the network estimator of the posterior probability $P(X_k = x^j|Z_{1:k})$ conditioned on the total spike count N_k. Suppose the true distribution is known only at initial time 1: $N_1^j = N_1 \omega_{1|1}^i$, we would like to investigate how the mean and variance of $\hat{P}_k^j$ vary over time. Given the previous distribution $\{\hat{P}_k^j\}$, the population response in the network follows a multi-nomial distribution:

$$
\begin{aligned}
N_{k+1}^{j} \mid N_{k+1}, \{P_k^i\} &\sim \text{Multinomial}(N_{k+1}, \frac{g(Z_{k+1}|x^j)}{P(Z_{k+1}|Z_{1:k})} \sum_i f(x^j|x^i)\hat{P}_k^i) \\
E[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}] &= \frac{g(Z_{k+1}|x^j)}{P(Z_{k+1}|Z_{1:k})} \sum_i f(x^j|x^i)\hat{P}_k^i \\
\text{Var}[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}] &= \frac{E[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}] - E^2[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}]}{N_{k+1}}
\end{aligned}
$$

Marginalizing over $\{\hat{P}_k^j\}$ we obtain the recursive upate equations for $E[\hat{P}_{k+1}^j]$ and $\text{Var}[\hat{P}_{k+1}^j]$ using the laws of total expectation and variance:

$$
\begin{aligned}
E[\hat{P}_{k+1}^j] &= \frac{g(Z_{k+1}|x^j)}{P(Z_{k+1}|Z_{1:k})} \sum_{i=1}^{\mathcal{X}} f(x^j|x^i)E[\hat{P}_k^i] \qquad (2.6) \\
\text{Var}[\hat{P}_{k+1}^j] &= E[\text{Var}[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}]] + \text{Var}[E[\hat{P}_{k+1}^j|\{\hat{P}_k^j\}]] \\
&= \frac{E[\hat{P}_{k+1}^j] - E^2[\hat{P}_{k+1}^j]}{N_{k+1}} + \frac{g^2(Z_{k+1}|x^j)}{P^2(Z_{k+1}|Z_{1:k})} \times \text{Var}[\sum_i f(x^j|x^i)\hat{P}_k^i]) \qquad (2.7)
\end{aligned}
$$

where $\eta_k^j = g^2(Z_{k+1}|x^j)/P^2(Z_{k+1}|Z_{1:k})$. The variance $\text{Var}[\hat{P}_{k+1}^j]$ can be partitioned into two parts. The first part represents the variance from current time step. The second part represents the variance from previous time step, but weighted by the coefficient $\frac{g^2(Z_{k+1}|x^j)}{P^2(Z_{k+1}|Z_{1:k})}$.

Since the initial distribution $\omega_{1|1}^j$ is known, the solution to equation 2.6 is easy to obtain:

$$
E[\hat{P}_k^j] = \omega_{k|k}^j \qquad (2.8)
$$

Thus, $\hat{P}_k^j$ is an unbiased estimator of true posterior probability $\omega_{k|k}^j$. However, the closed-form solution for the variance update equation 2.7 is generally intractable, except for some special forms of f. For example, consider a uniform transition model where $f(x^j|x^i) = 1/\mathcal{X}$. Since the $\{\hat{P}_k^i\}$ are negatively correlated, we have:

$$
\begin{aligned}
\text{Var}[\hat{P}_2^j] &= \frac{1}{N_2}(E[\hat{P}_2^j] - E^2[\hat{P}_2^j]) + 0 \\
\text{Var}[\hat{P}_3^j] &\leq \frac{1}{N_3}\{(E[\hat{P}_3^j] - E^2[\hat{P}_3^j]) + \frac{g^2(Z_3|x^j)}{P^2(Z_3|Z_{1:2})\mathcal{X}^2 N_3}(1 - \sum_i E^2[\hat{P}_2^i])\} \\
&\cdots \\
\text{Var}[\hat{P}_k^j] &\leq \frac{1}{N_k}\{(E[\hat{P}_k^j] - E^2[\hat{P}_k^j]) + \frac{g^2(Z_k|x^j)}{P^2(Z_k|Z_{1:k-1})\mathcal{X}^2 N_3}(1 - \sum_i E^2[\hat{P}_{k-1}^i]) + O(\frac{1}{\mathcal{X}^4})\} \qquad (2.9)
\end{aligned}
$$

Equation 2.9 has several implications. First, when the state space $\mathbb{X}$ is large, we can ignore the higher order terms of $\mathcal{X}$. The variance of the estimator becomes:

$$\mathrm{Var}[\hat{P}_k^j] \approx \frac{1}{N_K}(E[\hat{P}_k^j] - E^2[\hat{P}_k^j])$$

The variance of network estimator $\mathrm{Var}[\hat{P}_k^j] \propto 1/N_K$. Therefore $\mathrm{Var}[\hat{P}_k^j] \to 0$ as there are enough spikes in the network $N_k \to \infty$, showing that $\hat{P}_k^j$ is a consistent estimator of $\omega_{k|k}^j$.

In general, when the transition model is arbitrary, numerical methods are needed to study the relationship between the variance of $\hat{P}_k^j$ and k. In Figure 2.3, we test whether the above two implications still hold for random transition models. The state space is finite, $Z_k \sim N(X_k, 5)$. A form of divisive inhibition [32] mechanism was also employed to keep the number of spikes in the network roughly constant over time, e.g., $C_W C_M = 10N_k/N_1$. If the overall neural activity is weak at time k, then the global inhibition regulating the network weights is decreased to allow more spikes at time k + 1.

For the experiments, elements in the transition matrix $f(x^j|x^i)$ were first uniformly drawn from $[0, 1]$, and then normalized to ensure $\sum_j f(x^j|x^i) = 1$. In Figure 2.3(a-c), we examine equation 2.10 for different initial spike count values: $N_1 = 10^2, 10^3$ and 10^4. Each data point represents $\mathrm{Var}[\hat{P}_k^j]$ along the vertical axis and $E[\hat{P}_k^j] - E^2[\hat{P}_k^j]$ along the horizontal axis, calculated over 100 trials with the same random transition matrix f, and $k = 1, \ldots 10, j = 1, \ldots 20$. The solid lines represent a least squares power law fit to the data: $\mathrm{Var}[\hat{P}_k^j] = C_V * (E[\hat{P}_k^j] - E^2[\hat{P}_k^j])^{C_E}$. For 100 different random transition matrices f, the means of the exponential term C_E were $1.2863, 1.13$, and 1.037, with standard deviations $0.13, 0.08$, and 0.03 respectively, for $N_1 = 100$ and $\mathcal{X} = 4, 20$, and 100. The mean of C_E continues to approach 1 when $\mathcal{X}$ is increased, as shown in figure 2.3(d). Since $\mathrm{Var}[\hat{P}_k^j] \propto (E[\hat{P}_k^j] - E^2[\hat{P}_k^j])$ implies $\mathrm{Var}[n_{k|k}^j] \propto E[n_{k|k}^j]$, these results suggest that arbitrary transition models still preserve the Poisson variability.

The term C_V represents the scaling constant for the variance. Figure 2.3(e) shows that the mean of C_V over 100 different transition matrices f (over 100 different trials with the same f) is inversely proportional to initial spike count N_1, with power law fit $C_V = 1.77 N_1^{-0.9245}$. This indicates that the relation $\mathrm{Var}[\hat{P}_k^j] \propto 1/N_1$ (equation 2.9) still approximately holds no matter what the dynamics

model f is. The bias between estimated and true posterior probability can be calculated as:

$$bias(f) = \frac{1}{\mathcal{X}} \sum_{i=1}^{\mathcal{X}} (E[\hat{P}_k^i] - \omega_{k|k}^i)^2$$

The relationship between the mean of the bias (over 100 different f) versus initial count N_1 is shown in figure 2.3(f). Since the precision of the estimator $\hat{P}_k^j$ is limited by N_1, we also have an inverse proportionality between bias and N_1. Therefore, as the figure shows, for arbitrary f, the estimator $\hat{P}_k^j$ remains a consistent estimator of $\omega_{k|k}^j$.

In summary, we have proposed a spiking network model that approximates Bayesian filtering using spikes as Monte Carlo samples of probability distributions. We assume that the transition (dynamics) and emission (obsevation) models are known and encoded in the recurrent weights W and feedforward weights M, respectively. The model does not put any contraints on the particular form of the probability density over hidden world state. When the state space $\mathbb{X}$ is discrete, the spiking network provides the optimal Bayesian solution when the above assumptions hold.

2.4 Sequential filtering examples

We tested the filtering results of the proposed neural network with two other example HMMs. The first example is the classic stochastic volatility model. The transition model of the hidden volatility variable $f(X_{k+1}|X_k) = \mathcal{N}(0.91X_k, 1.0)$, and the emission model of the observed price given volatility is $g(Z_k|X_k) = \mathcal{N}(0, 0.25\exp(X_k))$. The posterior distribution of this model is uni-modal. In simulation we divided $\mathbb{X}$ into 100 bins, and initial spikes $N_1 = 1000$. We plotted the expected volatility with estimated standard deviation from the population posterior distribution in Figure 2.2(a). We found that the neural network does indeed produce a reasonable estimate of volatility and plausible confidence interval.

The second example tests the network's ability to approximate bi-modal posterior distributions. In a vertical symmetric maze shown in figure 1 (left), one cannot infer the hidden vertical coordinate $X \in \{1, \ldots, \mathcal{X}\}$ by using only observations about the surrounding environment. Let the observation Z be the arbitrary observation, for the demonstration purpose, the emission probability of this vertical symmetric maze can have the form: $g(z|x) = g(z|\mathcal{X} + 1 - x)$ and

$g(z|x) = \mathcal{N}(x, \sigma_z) + \mathcal{N}(x - \mathcal{X} - 1, \sigma_z)$. Evidence alone only tells information about the relative distance to the upper or lower boundaries. Suppose the agent in the maze can only move forward in the horizontal direction. The transition probability $f(x'|x) = \frac{x-x'-2}{6} I_{|x'-x|\leq 1}$. Suppose the prior distribution over the vertical coordinate X is uniform initially. Figures 3b compares the time varying population posterior distribution with the true one using heat maps. The vertical axis represents the hidden state and the horizontal axis represents time steps. The magnitude of the probability is represented by the intensity of the pixel. In this example, $\mathbb{X} = \{1, \ldots, 8\}$ and there are 20 time steps.

3 On-line parameter learning

In the previous section, we assumed that the model parameters, i.e., the transition probabilities $f(X_{k+1}|X_k)$ and the emission probabilities $h(Z_k|X_k)$, are known. In this section, we describe how these parameters $\theta = \{f, g\}$ can be learned from noisy observations $\{Z_k\}$. Traditional methods to estimate model parameters are based on the Expectation-Maximization (EM) algorithm [43], which maximizes the (log) likelihood of the unknown parameters $\log P_\theta(Z_{1:k})$ given a set of observations collected previously. However, such an "off-line" approach is biologically implausible because (1) it requires animals to store all of the observations before learning, and (2) evolutionary pressures dictate that animals update their belief over θ sequentially any time a new measurement becomes available.

We therefore propose an on-line estimation method where observations are used for updating parameters as they become available and then discarded. Our approach is based on recursively calculating the sufficient statistics of θ using stochastic approximation algorithms and the Monte Carlo method. We explore how animals can implement this on-line learning algorithm in a spiking network, where changes in synaptic weights are subject to Hebbian learning rules.

Here we describe a general framework for on-line learning of parameters in non-linear non-Gaussian state space models, following [3, 28]. Again, $\{X_k,\, k \in \mathbb{N}\}$ is a hidden Markov process, with the additional assumption that it is stationary and ergodic: as before, $X_{k+1}|X_k \sim f_\theta(X_{k+1}|X_k)$ and $\{Z_k\}$ are the observations with emission probabilities $Z_k|X_k \sim g_\theta(Z_k|X_k)$. We

would like to find the parameters θ that maximize the log likelihood: $\log P_\theta(Z_{1:k}) = \sum_{t=1}^{k} \log P_\theta(Z_t|Z_{t-1})$.

We first show how the traditional off-line EM algorithm [43] accomplishes this goal in an iterative manner. In iteration k, the EM algorithm approximates the joint log-likelihood $\log P_\theta(X, Z)$ using the *expected* value over the hidden data X based on the current estimate of parameters θ_k (E-step):

$$Q(\theta, \theta_k) \quad = \quad E_{\theta_k}[\log P_\theta(X, Z)|Z]$$

Then the value of θ that *maximizes* $Q(\theta, \theta_k)$ is found in the M-step. This gives rise to the new estimate:

$$\theta_{k+1} = \arg\max Q(\theta, \theta_k).$$

Maximizing $Q(\theta, \theta_k)$ is equivalent to increasing the marginalized log likelihood $\log P_\theta(Z_{1:k})$, since their gradient terms coincide [43]:

$$E_\theta \nabla_\theta \log P_\theta(X, Z)|Z] = \nabla_\theta \log P_\theta(Z),$$

Starting from an initial guess θ_0, the EM algorithm generates a sequence of estimates $\{\theta_k\}$, which converge to the true parameter θ^* under some regularity conditions [161].

To perform on-line parameter estimation, we aim to produce a new estimate θ_{k+1} when the observation Z_k becomes available, where θ_{k+1} maximizes the function

$$\begin{aligned} Q(\theta, \theta_k) &= E_{\theta_k}[\log P_\theta(X_{1:k}, Z_{1:k})|Z_{1:k}] \\ &= E_{\theta_k}[\sum_{t=1}^{k} \log P_\theta(X_t, Z_t|X_{t-1})|Z_{1:k}] \\ &= E_{\theta_k}[\sum_{t=1}^{k} \log(f_\theta(X_t|X_{t-1}) g_\theta(Z_t|X_t))|Z_{1:k}] \end{aligned} \tag{2.10}$$

In general, $Q(\theta, \theta_k)$ and its derivative $\nabla_\theta Q(\theta, \theta_k)$ are difficult to estimate because they are functions of the complete data $\{X_{1:k}, Z_{1:k}\}$. Equation 2.10 is only of theoretical interest unless the unknown parameter θ can be estimated, without any loss of information, from a function of the

complete data that has much lower dimension, the so-called sufficient statistic for θ. As an example, suppose the likelihood $P_\theta(X_t, Z_t|X_{t-1}) = f_\theta(X_t|X_{t-1})g_\theta(Z_t|X_t)$ belongs to an exponential family [30]:

$$P_\theta(X_t, Z_t|X_{t-1}) \propto P_\theta(T) = \exp[\psi(\theta) \cdot T(X_t, Z_t, X_{t-1}) - A(\theta)]$$

where $T(X_t, Z_t, X_{t-1})$ is a complete sufficient statistic for parameter θ, and ψ and A are arbitrary functions of θ. All inference about θ depends only on

$$\hat{T}(\theta_k) = k^{-1}E_{\theta_k}[\sum_{t=1}^{k} T(X_t, Z_t, X_{t-1})|Z_{1:k}],$$

which is the expected sufficient statistic of the joint distribution $P(X_{1:k}, Z_{1:k})$. The expectation $E_{\theta_k}(.|Z_{1:k})$ is taken with respect to the posterior distribution $P_{\theta_k}(X_{1:k}|Z_{1:k})$ based on the current θ_k.

An online EM algorithm can be obtained by approximating the expected sufficient statistic $\hat{T}(\theta_k)$ using the stochastic approximation (or Robbins-Monoro) procedure [133]:,

$$\hat{T}(\theta_k) \simeq \eta_k E_{\theta_{k-1}}(T(X_{k-1}, Z_k, X_k)|Z_k) + (1 - \eta_k)\hat{T}(\theta_{k-1}), \tag{2.11}$$

where the learning rate η_k is a decreasing function of k. Equation 2.11 enables us to combine new observations Z_k with the previous estimate $\hat{T}(\theta_{k-1})$ sequentially. When the learning rate is small $\eta_k \to 0$ such that θ_k changes slowly, the approximation in equation 2.11 becomes exact. In general, convergence is guaranteed when $\sum_{k=1}^{\infty} \eta_k = \infty$ and $\sum_{k=1}^{\infty} \eta_k^2 < \infty$. Note that if $\eta_k = 1/k$, $\hat{T}_k$ is simply the running average of T.

In summary, the online EM algorithm based on the sufficient statistic can be re-written as:

E-step $\hat{T}(\theta_k) = \eta_k E_{\theta_{k-1}}(T(X_{k-1}, Z_k, X_k)|Z_k) + (1 - \eta_k)\hat{T}(\theta_{k-1})$

M-step $\theta_{k+1} = \arg\max P_\theta(\hat{T}_k)$, which is the unique solution to the equation

$\nabla_\theta \psi(\theta) \cdot \hat{T}_k = \nabla_\theta A(\theta)$.

3.1 Learning transition and emission probabilities

For a discrete hidden Markov model, the unknown parameters θ consist of the transition matrix $f_{ij} = f(x^j|x^i)$ and the emission probability matrix $g_{ij} = g(z^j|x^i)$. Recall that for the spiking network in a previous section, we defined M^k and W^k as the feed-forward and recurrent weights respectively at time step k. In this section, we introduce Hebbian learning rules (based on equation 2.11) for the synaptic weights M^k and W^k such that M^k and W^k become consistent estimators of f and g respectively as $k \to \infty$.

Recall that the population of inference neurons in the model maintains a Monte-Carlo approximation of the posterior distribution $P_{\theta_k}(X_k|Z_{1:k})$ over the hidden state X_k, given observations up to time k. However, the expectation $E_{\theta_{k-1}}(T(X_{k-1}, Z_k, X_k)|Z_k)$ in equation 2.11 is taken with respect to the *smoothed* distribution

$$P_{\theta_k}(X_{k-1}, X_k|Z_{1:k}) = P_{\theta_k}(X_{k-1}|X_k, Z_{1:k})P(X_k|Z_{1:k}),$$

which is the product of the posterior distribution and the distribution of hidden state at the previous time step $k-1$ given the observations $Z_{1:k}$. Such a retrospective distribution cannot be implemented in a two-layer spiking network such as the one described above. Therefore, we employ an approximation to equation 2.11:

$$\begin{aligned}
\hat{T}(\theta_k) &\simeq \eta_k \times \sum_{X_k, X_{k-1}} T(X_{k-1}, Z_k, X_k)P(X_k|Z_{1:k}, \theta_{k-1})P(X_{k-1}|Z_{1:k-1}, \theta_{k-1}) \\
&\quad +(1-\eta_k) \times \hat{T}(\theta_{k-1}) \\
&\simeq \eta_k \times \sum_{n=1}^{N_k} \sum_{n'=1}^{N_{k-1}} T(\hat{x}_{k-1}^{n'}, Z_k, \hat{x}_k^n)/(N_k \times N_{k-1}) + (1-\eta_k) \times \hat{T}(\theta_{k-1}) \qquad (2.12)
\end{aligned}$$

where $\{\hat{x}_k^n\}$ and $\{\hat{x}_{k-1}^{n'}\}$ are Monte-Carlo samples drawn from posterior distributions $P(X_{k-1}|Z_{1:k-1}, \theta_{k-1})$ and $P(X_k|Z_{1:k}, \theta_k)$ respectively.

The sufficient statistic for g given the current estimator $M^k = \hat{g}_k$ can be written as

$$\begin{aligned}
T(X_k, Z_k|g) &= \delta(X_k = x^j, Z_k = z^i) \\
\hat{T}(M^k) &= \eta_k E_{M^{k-1}}(\delta(X_k = x^j, Z_k = z^i)|Z_k) + (1-\eta_k)\hat{T}(M^{k-1}) \qquad (2.13)
\end{aligned}$$

The expectation in the first term can be further approximated by Monte Carlo sampling of spikes:

$$E_{M^{k-1}}(\delta(X_t = x^j, Z_t = z^i)|Z_k) = \frac{N_k^j}{N_k} \times \frac{\tilde{N}_k^i}{\tilde{N}_k},$$

where N_k^j is the number of post-synaptic spikes in the j-th inference neurons, $\tilde{N}_k^i$ is the number of pre-synaptic spikes in i-th sensory neurons at time k, and $\tilde{N}_k = \sum_i \tilde{N}_k^i$.

The corresponding M-Step is given by:

$$\hat{g}_k = M^k = \frac{\hat{T}(M_{ij}^k)}{\sum_i \hat{T}(M_{ij}^k)}$$

Combining the above equation with equation 2.13, we derive a local Hebbian learning rule for M^k:

$$\begin{aligned} M_{ij}^k &= \frac{\eta_k N_k^j}{N_k} \times \frac{\tilde{N}_k^i}{\tilde{N}_k^i} + (1 - \frac{\eta_k N_k^j}{N_k}) \times M_{ij}^{k-1} \\ \frac{M_{ij}^k - M_{ij}^{k-1}}{\eta_k^M} &= -M_{ij}^{k-1} + \frac{\tilde{N}_k^i}{\tilde{N}_k}, \quad \text{when } N_k^j > 0, \end{aligned} \tag{2.14}$$

where the effective learning rate $\eta_k^M = \eta_k \frac{N_k^j}{N_k}$ is proportional to the post-synaptic activity N_k^j in the inference layer population. A higher value of N_k^j/N_k represents a higher posterior belief for the world state X^j, resulting in faster learning.

Similarly, the transition probability matrix f can be learned by estimating its sufficient statistics:

$$T(X_k, Z_t, X_{k-1}|f_{ij}) = \delta(X_k = x^j, X_{k-1} = x^i)$$

Equation 2.11 can then be implemented as

$$\begin{aligned} \hat{T}(W^k) &= \eta_k E_{W^{k-1}}(\delta(X_{k-1} = x^i, X_k = x^j)|Z_k) + (1 - \eta_k)\hat{T}(W^{k-1}) \\ &= \eta_k \times \frac{n_{k-1|k-1}^i}{N_{k-1}} \times \frac{n_{k|k}^j}{N_k} + (1 - \eta_k) \times W_{ij}^{k-1} \end{aligned} \tag{2.15}$$

The corresponding M-step also has the form:

$$W_{ij}^k = \frac{\hat{T}(W_{ij}^k)}{\sum_{j=1}^{\mathcal{X}} T(W_{ij}^k)}$$

Combining the above equation with equation 2.15, we derive a local Hebbian learning rule for M^k:

$$\begin{aligned} W_{ij}^k &= \eta_k \frac{N_{k-1}^i}{N_{k-1}} \times \frac{N_k^j}{N_k} + (1 - \eta_k \frac{N_{k-1}^i}{N_{k-1}}) \times W_{ij}^{k-1} \\ \frac{W_{ij}^k - W_{ij}^{k-1}}{\eta_k^W} &= -W_{ij}^{k-1} + \frac{N_k^j}{N_k}, \quad \text{when } N_{k-1}^i > 0, \end{aligned} \tag{2.16}$$

where the effective learning rate $\eta_k^W = \eta_k \frac{N_{k-1}^i}{N_{k-1}}$ is proportional to the pre-synaptic activity N_{k-1}^i in the inference layer population.

Numerical simulation on convergence of parameter learning

Learning both emission and transition probability matrices at the same time using the online EM algorithm with stochastic approximation is very difficult because there are many local minima in the likelihood function. To simplify the task, we divide the learning process into two phases. The first phase involves learning the emission probability g when the hidden world state is stationary, *i.e.*, $W_{ij} = f_{ij} = \delta_{ij}$. This corresponds to learning the observation model of static objects at the center of gaze before learning the dynamics f of objects. After an observation model g is learned, we relax the stationary constraint, and allow the spiking network to update the recurrent weights W to learn the arbitrary transition probability f.

Figure 2.4 illustrates the performance of learning rules (2.14) and (2.16) for a discrete HMM with $\mathcal{X} = 4$ and $\mathcal{Z} = 12$. X and Z values are spaced equally apart: $X \in \{1, \ldots, 4\}$ and $Z \in \{\frac{2}{3}, 1, \frac{4}{3}, \ldots, 4\frac{1}{3}\}$. The transition probability matrix f then involves $4 \times 4 = 16$ parameters and the emission probability matrix g involves $12 \times 4 = 48$ parameters.

In figure 2.4(a), we examine the performance of learning rule 2.14 for the feedforward weights M^k, with fixed transition matrix $f_{ij} = \delta_{ij}$. The true emission probability matrix has the form $g_{\cdot j} = P(Z_k | X_k = x^j) \sim N(x^j, \sigma_Z^2)$. Each column of g is a Gaussian with observation noise σ_Z. The solid blue curve shows the average MSE between the learned feedforward weights M^k and the true emission probability matrix g over trials with different g, with $MSE(k) = \sqrt{\sum_{ij}(M_{ij}^k - g_{ij})^2}$. The dotted lines show $\pm$ 1 standard deviation for MSE based on 10 different trials. σ_Z varied

from trial to trial and was drawn uniformly between 0.2 and 0.4, representing different levels of observation noises. The initial spike distribution was uniform $N_0^i = N_0^j, \forall i, j = 1 \ldots, \mathcal{X}$ and the initial estimate $M_{i,j}^0 = \frac{1}{\mathcal{Z}}$. The learning rate was set to $\eta_k = \frac{1}{k}$, although a small constant learning rate such as $\eta_k = 10^{-5}$ also gives rise to similar learning results.

A notable feature in figure 2.4(a) is that the average MSE exhibits a fast power-law decrease. The red solid line in figure 2.4(a) represents the power-law fit to the average MSE: $MSE(k) \propto k^{-1.1}$. Furthermore, the standard deviation of MSE approaches zero as k grows large. Figure 2.4(a) thus shows the asymptotic convergence of equation (2.14) irrespective of the σ_Z of the true emission matrix g.

We next examined the performance of learning rule 2.16 for the recurrent weights W^k, given the learned emission probability matrix g (the true transition probabilities f are unknown to the network). The initial estimator $W_{ij}^0 = \frac{1}{\mathcal{X}}$. Performance was evaluated by calculating the mean square error $MSE(k) = \sqrt{\sum_{ij}(W_{ij}^k - f_{ij})^2}$ between the learned recurrent weight W^k and the true f. Different randomly chosen transition matrices f were tested. The average MSE and standard deviation over trials with different f are displayed in blue solid and dotted lines respectively in figure 2.4(b) and figure 2.4(c).

When $\sigma_Z = 0.04$, the observation noise is $\frac{0.04}{1/3} = 12\%$ of the separation between two observed states. Hidden state identification in this case is relatively easy. The red solid line in figure 2.4(b) represents the power-law fit to the average MSE: $MSE(k) \propto k^{-0.36}$. Furthermore the standard deviation of MSE approaches zero as k grows large, indicating asymptotic convergence of equation 2.16 irrespective of the form of the true transition matrix f. Similar convergence results can still be obtained for higher σ_Z, e.g., $\sigma_Z = 0.4$ (figure 2.4(c)). In this case, hidden state identification is much more difficult as the observation noise is now 1.2 times the separation between two observed states. This difficulty is reflected in a slower asymptotic convergence rate, with a power-law fit $MSE(k) \propto k^{-0.21}$, as indicated by the red solid line in figure 2.4(c). In the extreme case when $\sigma_Z = 1$, hidden state identification becomes impossible due to high observation noise, causing the online learning rule (2.16) to fail.

Learning Example

Using the maze example in figure 1 (left) again, we show the learning results in the Poisson neural network. In this example, $\mathcal{X}$ = $\mathcal{Z}$ = 4. the true symmetric emission probability has the from $g(z|x) = \mathcal{N}(x, \sigma_z) + \mathcal{N}(x - \mathcal{Z} - 1, \sigma_z)$. and the true transition probability is defined as $f(x'|x) = \frac{x-x'-2}{6} I_{|x-x'|\leq 1}$. We initialed the feed-forward and recurrent weight matrices as the : $M_0 = g(z|x) + \epsilon_M$ and $W_0 = f(x'|x) + \epsilon_W$, assuming the agent at the beginning of the trial has only partial knowledge about these matrices. Figure 2.5(a) shows the true, initial and learned matrices for emission probabilities and transition probabilities. The mean square errors between the learned matrix and the true matrix also show power law decay as the number of time steps increases. In this simulation, the number of trials is only one and the number of time steps are $1,000,000$.

Discussion

We have described a two-layer Poisson network model that encodes the posterior probability distribution of hidden world states as a sampled distribution represented by spikes across a neural population. Neural variability in spiking arises naturally as a consequence of sampling necessary for inference. Our model embraces many biological properties that are frequently observed in CNS neurons, such as divisive normallization, and spike-time dependent Hebbian plasticity.

There have been a number of previous models of probabilistic inference in biological neural networks. The Boltzmann machine [75, 141] is perhaps the earliest example of a neural network capable of probabilistic inference. Similar to our model, Boltzmann machines employ a sampling based inference technique that allows them to learn an internal probabilistic model from the observations. Our model differs from the Boltzmann machine in the underlying generative model. Our model can represent probabilistic state transitions and can implement the state-space dynamics of arbitrary hidden Markov models. A recurrent neural network capable of statistical inference in hidden Markov models was first suggested by [22]. One limitation of Bridle's model, known as the Alpha-net, was the assumption that the network could multiply arbitrary probabilities. In contrast,

the inference performed in our model requires binary AND operations, which can be more easily implemented in a population of neurons.

The idea of representing probability distributions using populations of neurons originated in early work on basis function networks [2, 47] and distributional population coding [168, 167, 165, 162] models. In the basis function approach, probability distributions are decomposed into linear combinations of basis functions, which are proportional to the measurable tuning functions of neurons. Due to its additive nature, the probability distributions that can be represented by this approach cannot be sharper than the component distributions. In contrast, the model proposed in this article can approximate probability distributions of any shape as a sampled distribution. Distributional population coding (DPC) uses a generative model to encode a probability distribution in a population of neurons. DPC requires a sophisticated non-neural decoding mechanism to recover the distribution from the neural population response, compared to the straightforward readout of the distribution from the spiking network proposed in this article.

There have also been several neural models for Bayesian inference of hidden world state proposed in recent years. Rao [124] proposed a model in which the firing rates of a population of neurons approximate the log probabilities of the time-varying posterior distribution of hidden states, given noisy observations, for an arbitrary hidden Markov model. Beck et al. [8] extended Rao's work using nonlinear recurrent networks for exact inference, with firing rates in a population directly proportional to posterior probabilities. Rao [128] proposed a nonlinear network model for implementing belief propagation for Bayesian inference in arbitrary graphical models. Models based on predictive coding [129, 123], basis function networks [46] and line attractor networks [160] have been proposed for implementing the Kalman filter, which assumes all distributions are Gaussian and the dynamics is linear. More recently, Bobrowski et al. [18, 17] proposed a spiking network model that can compute the optimal posterior distribution in continuous time. One limitation of these models is that the model parameters (the emission probability and transition probability matrix) are assumed to be known a priori, whereas those model parameters are learned using a form of Hebbian learning in the model proposed here. Probabilistic population codes [98, 7] (PPC) provide an alternative way to estimating the probability distribution of the

hidden state in populations of neurons. The PPC model exploits neural variability to turn products in Bayesian computations into sums without the need for a log likelihood representation. However, the PPC approach assumes a static world state X. Deneve [44, 45] proposed a model for inference and learning based on the dynamics of a single neuron but assuming the number of world states is limited by 2.

The model for learning we have proposed builds on prior work on online learning [3, 107, 28, 27]. The online algorithm used in our model for estimating HMM parameters involves three levels of approximation. The first level involves performing a stochastic approximation to estimate the expected complete-data sufficient statistics over the joint distribution of all hidden states and observations. Cappe and Moulines [28] showed that under some mild conditions, such an approximation produces a consistent, asymptotically efficient estimator of the true parameters. The second approximation comes from the use of filtered rather than smoothed posterior distributions in equation 2.11. Although the convergence reported in the methods section is encouraging, a rigorous proof of convergence remains to be shown. The asymptotic convergence rate using only the filtered distribution is about one third the convergence rate obtained for the algorithms in [107] and [28], where the smoothed distribution is used. The third approximation results from Monte-Carlo sampling of the posterior distribution in equation 2.12. As discussed in the methods section, the Monte Carlo approximation converges in the limit of large numbers of particles (spikes).

Our model suggests that, contrary to the commonly held view, variability in spiking does not reflect "noise" in the nervous system but captures the animal's uncertainty about the outside world. This suggestion is similar to previous models linking firing rate variability to probabilistic representations [79, 98] but differs in the emphasis on spike-based representations and time-varying inputs. In our model, a probability distribution over a finite sample space is represented by spike counts in neural sub-populations. Treating spikes as random samples requires that neurons in a pool of identical cells fire independently. This hypothesis is supported by a recent experimental finding [55] that nearby neurons with similar orientation tuning and common inputs show little or no correlation in activity. Our model offers a functional explanation for the existence of such decorrelated neuronal activity in the cortex.

Unlike many previous models of cortical computation, our model treats synaptic transmission between neurons as a stochastic process rather than a deterministic event. This acknowledges the inherent stochastic nature of neurotransmitter release and binding. Synapses between neurons usually have only a small number of vesicles available and a limited number of post-synaptic receptors near the release sites. Recent physiological studies [113] have shown that only 3 NMDA receptors open on average per release during synaptic transmission. These observations lend support to the view espoused by the model that synapses should be treated as probabilistic computational units rather than as simple scalar parameters as assumed in traditional neural network models.

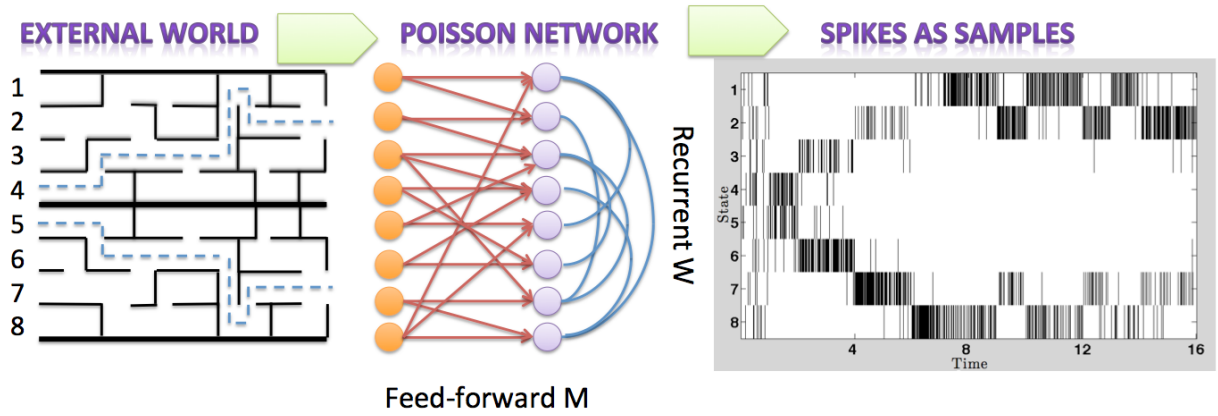
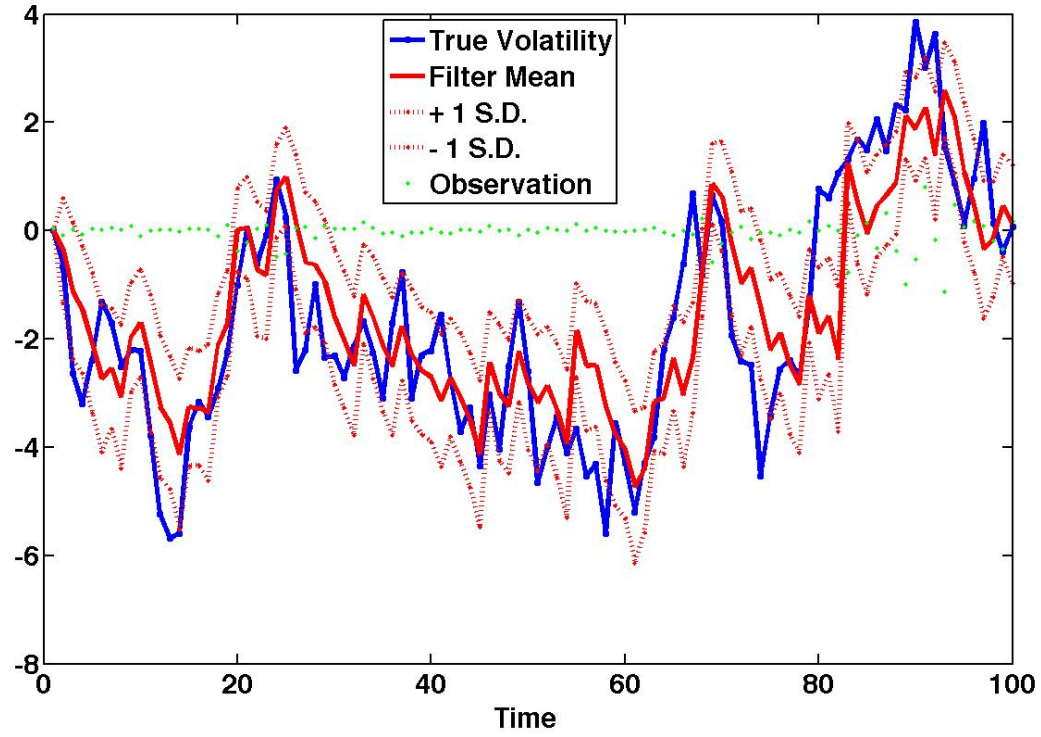
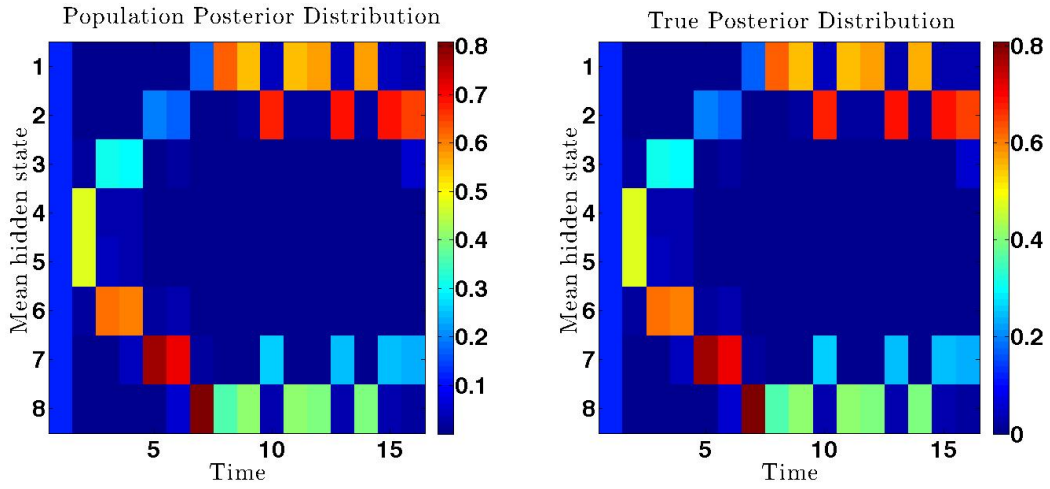


Figure 2.1. Spiking network model for sequential Monte Carlo Bayesian inference. *Left*: In a vertical symmetric maze, sensory evidence about the surround along is not enough to infer the hidden vertical coordinate $X_k \in \{1, \ldots, 8\}$. One has to maintain a bimodal belief about the hidden state. *Center*: A two layer neuronal network used to update the belief distribution given the sensory observation. Lower layer neurons on the left received sensory evidence and sent signals to higher layer neurons on the right via feed-forward connections. Recurrent connections among the higher layer neurons are used to combine prior belief from previous iteration. *Right*: Firing activities in the higher layer neurons approximate the posterior distribution of the hidden state. Each spike can be interpreted as a Monte Carlo sample of the hidden state.

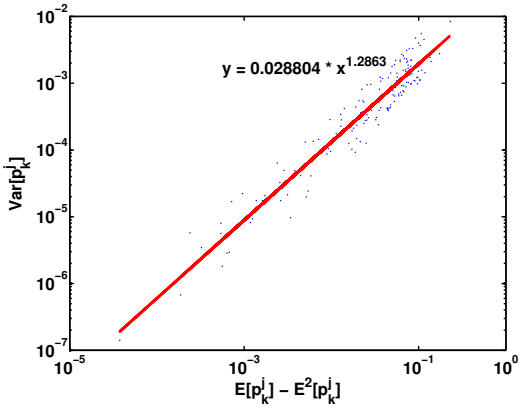


(a)

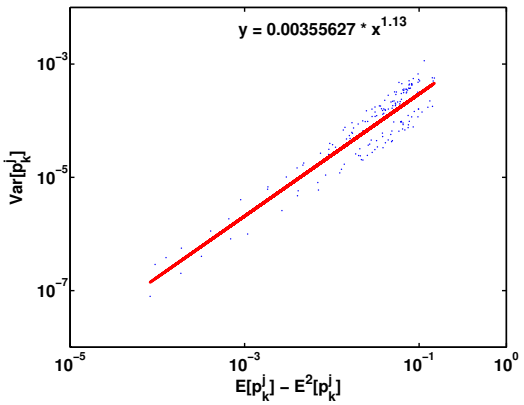


(b)

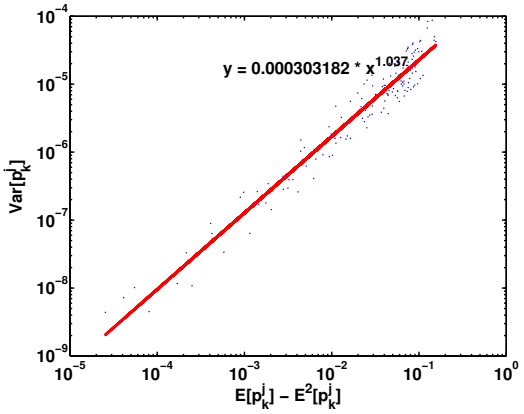
Figure 2.2. Filtering results for models with uni-modal (a) and bi-modal (b–d) posterior distribution - see text for details).


y = 0.028804 * x^1.2863
Var[p_k^j]
E[p_k^j] – E^2[p_k^j]

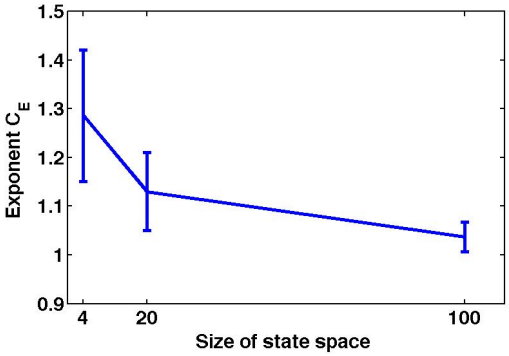
(a)


y = 0.00355627 * x^1.13
Var[p_k^j]
E[p_k^j] – E^2[p_k^j]

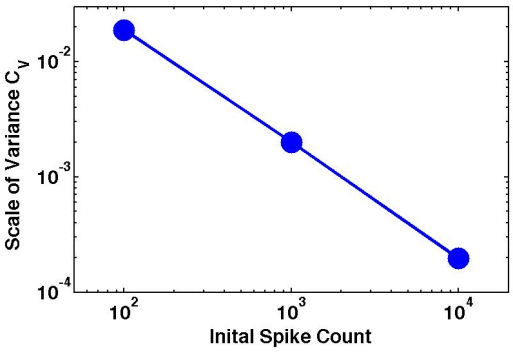
(b)


y = 0.000303182 * x^1.037
Var[p_k^j]
E[p_k^j] – E^2[p_k^j]

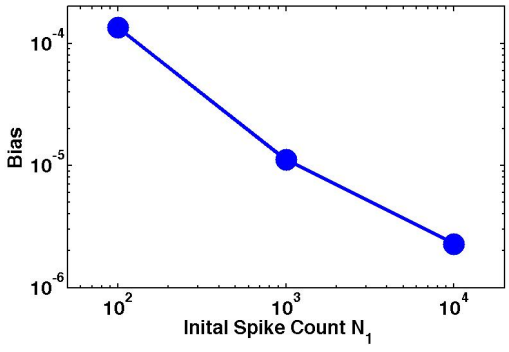
(c)


Exponent C_E
Size of state space

(d)


Scale of Variance C_V
Inital Spike Count

(e)


Bias
Inital Spike Count N_1

(f)

Figure 2.3 ***(preceding page).*** **Variance versus Mean of estimator for different initial spike counts** (a) $N_1 = 100$, (b) $N_1 = 1000$, (c) $N_1 = 10,000$. Each data point represents the variance of the estimator $\hat{P}_k^i$ (vertical axis) and "mean" $E[\hat{P}_k^i] - E^2[\hat{P}_k^i]$(horizontal axis) over 100 different trials with the same transition matrix f, for $i = 1, \ldots 20$ and $k = 2, \ldots, 10$. The solid lines are least-square power law fits $\mathrm{Var}[\hat{P}_k^i] = C_V * (E[\hat{P}_k^i] - E^2[\hat{P}_k^i])^{C_E}$ to different data sets, with coefficients (C_V, C_E) shown in the legend. (d) The mean of the exponential term C_E over 100 different transition matrices f approaches 1 as $\mathcal{X}$ increases. (e) & (f) The mean of C_V decreases as N_1 increases, as does the bias between the mean of the estimator and true posterior probability $\frac{1}{\mathcal{K}\mathcal{X}} \sum_{i,k} (E[\hat{P}_k^i] - \omega_{k|k}^i)^2$.

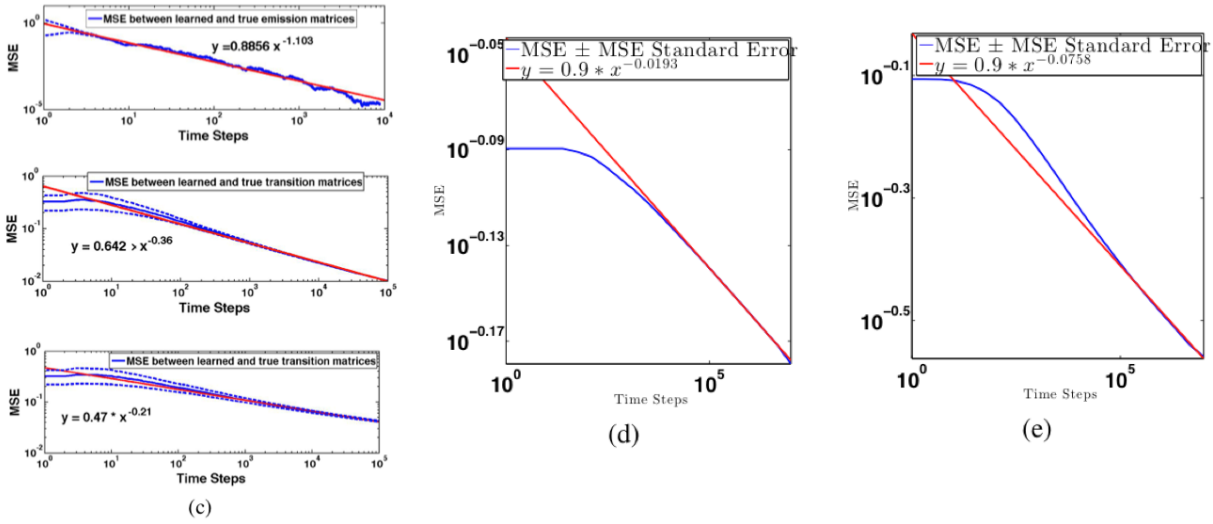
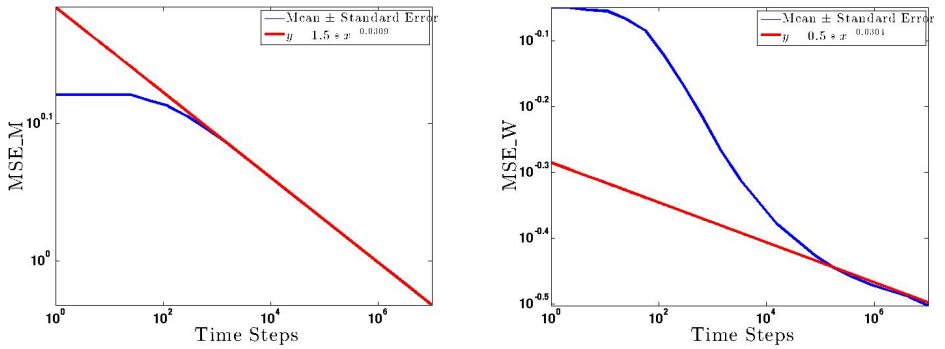
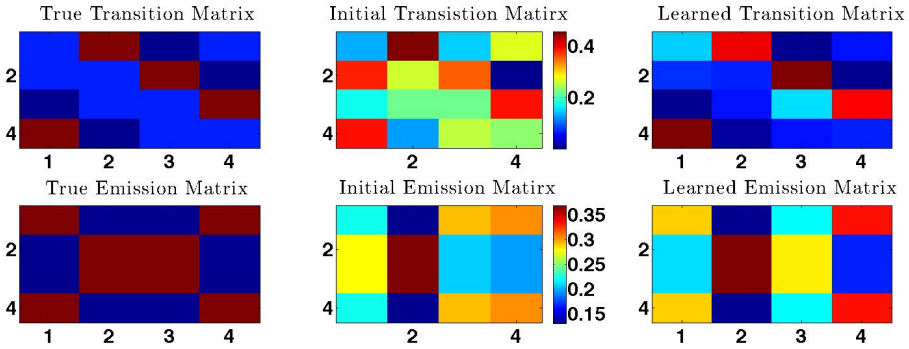


Figure 2.4. Performance of the Hebbian Learning Rules. (a) The mean square error (MSE) between the learned M^k and the true emission probability g as a function of the number of time steps k. The blue solid line shows the average MSE over trials with different g. The initial estimator M^0 was randomly chosen. The dotted lines show $\pm$ 1 standard deviation. The red straight line is the power law fit $y = ax^b$ to the average MSE. (b) MSE between learned W^k and true transition matrix f when the observation noise σ_Z is low, after the emission model g has been learned. (c) MSE between learned W^k and true transition matrix f when the observation noise σ_Z is high. As expected, learning is slower when the noise level of the observations is larger.



(a)



(b)

Figure 2.5. Learning results for model with bimodal posterior distribution.

4 Appendix: Spiking Network Model using Binary neurons

The model we have proposed assumes an underlying hidden Markov model (HMM) for processing sensory information. This assumption implies that the sensory system makes noisy observations Z of the external world at discrete time steps (corresponding to the time steps of the HMM), and updates its belief over hidden world state X each time a new observation is made. The mechanism of coincidence detection in inference neurons provides a way of bridging the gap between the discrete time steps in the HMM and continuous time in a neural network. The arrival of sensory EPSPs at time t_i mark the onset of the i-th HMM epoch. The inference neurons then compute the posterior belief by combining the current observation and prior belief before time step t_{i+1}. This implies that the coincidence detection window should be less than the length of one HMM epoch, requiring relatively precise timing and low temporal variability in the sensory observations. The brain's ability to transmit temporal information with high precision and low variability has been studied by a number of researchers [85, 155]. In particular, [155] found that the output firing rate is a highly nonlinear function of the number of synchronous synaptic events. This supports the assumption in the model that recurrent or feed-forward inputs alone are not sufficient to cause an inference neuron to spike: the coincidence of the two inputs is required to make spiking highly likely.

4.1 Network architecture

Let $\mathbf{s}_k$ denote the binary vector of activities at time k in the hidden-layer inference neurons. The following equation defines the dynamics of the network:

$$\mathbf{s}_k = \Phi(\mathbf{a}_k, \mathbf{b}_k) \tag{2.17}$$

where Φ is the neuron's response function, $\mathbf{a}_k$ is the vector representing the inference neurons' recurrent inputs, which are determined by the recurrent weight matrix W and $\mathbf{s}_{k-1}$ from the previous time step, and $\mathbf{b}_k$ is the vector representing feedforward inputs, which are determined by the feedforward weight matrix M and sensory measurement Z_k. How can Bayesian inference be

achieved using the above dynamics? We approach this problem by first showing how this neural network can represent probability distributions.

Neural representation of probability distributions

Similar to the idea of grid-based filtering, we first divide the inference neuron population into $\mathcal{X}$ sub-populations. $\mathbf{s} = \{s_l^i, i = 1, \ldots \mathcal{X}, l = 1, \ldots, \mathcal{L}\}$. $s_l^i(k) = 1$ if there is a spike in the l-th neuron of the i-th sub-population at time step k. $s_l^i(k) = 0$ otherwise. Each sub-population of $\mathcal{L}$ neurons share the same preferred world state, there being $\mathcal{X}$ such sub-populations representing each of $\mathcal{X}$ preferred states. One can, for example, view a neuron sub-population as a cortical column, within which neurons encode similar features [55]. A spike, generated by a neuron whose preferred world state is x^i at time k, represents an independent Monte Carlo sample (particle) from the posterior probability $P(X_k = x^i|Z_k)$. Neural variability can thus be interpreted as arising naturally due to sampling [79]. As depicted in the raster plot of Figure 2.1, the distribution of spikes across the entire inference layer population is a Monte-Carlo approximation to the current posterior distribution:

$$n_{k|k}^i := \sum_{l=1}^{\mathcal{L}} s_l^i(k) \propto \omega_{k|k}^i \tag{2.18}$$

$$N_k = \sum_{i=1}^{\mathcal{X}} n_{k|k}^i \tag{2.19}$$

where $n_{k|k}^i$ is the number of spiking neurons in the ith sub-population at time k, which can also be regarded as the instantaneous firing rate for sub-population i. N_k is the total spike count in the inference layer population. The set $\{n_{k|k}^i\}$ represents the unnormalized conditional probabilities of X_k, so that $P(X_k = x^i|Z_{1:k}) = \omega_{k|k}^i = n_{k|k}^i/N_k$.

With the above neural representation of probability distributions, we next show how suitable neural dynamics and synaptic weights can be chosen such that the spike distribution $n_{k|k}^i/N_k$ will propagate as illustrated in the example in figure 2.6. We tackle the problem of learning the synaptic weights in a later section.

Bayesian inference with stochastic synaptic transmission

To implement the prediction equation 2.1 in a spiking network, we require that the recurrent weights between the inference neurons encode the transition probabilities: $W_{ij} = f(x^j|x^i)/C_W$, where C_W is a scaling constant. We define the recurrent weight W_{ij} to be the synaptic release probability between i-th neuron sub-population and j-th neuron sub-population in the inference layer. Each neuron that spikes at time step k will randomly evoke, with probability W_{ij}, one recurrent excitatory post-synaptic potential (EPSP) at time step $k+1$, after some network delay. We define the number of recurrent EPSPs received by neuron l in the j-th sub-population as a_l^j. Thus, a_l^j is the sum of N_k independent (but not identically distributed) Bernoulli trials:

$$a_l^j(k+1) \;=\; \sum_{i=1}^{\mathcal{X}} \sum_{l'=1}^{\mathcal{L}} \epsilon_{l'}^i s_{l'}^i(k), \quad \forall l = 1 \ldots \mathcal{L}. \tag{2.20}$$

where $P(\epsilon_l^i = 1) = W_{ij}$ and $P(\epsilon_l^i = 0) = 1 - W_{ij}$. The sum a_l^j follows the so-called "Poisson binomial" distribution [76] and in the limit approaches the Poisson distribution:

$$P(a_l^j(k+1) \geq 1) \;\simeq\; 1 - \exp(-\sum_i W_{ij} n_{k|k}^i) \tag{2.21}$$

$$\simeq\; \sum_i W_{ij} n_{k|k}^i = \frac{N_k}{C_W} \omega_{k+1|k}^i \tag{2.22}$$

where the absolute difference between the two sides of equation 2.21 is bounded by $3\sqrt[3]{\frac{\max_i f_{ij}}{C_W}}$. Higher order terms involving $\frac{N_k}{C_W}$ are discarded in the approximation of equation 2.22. Detailed analysis of the distribution of a_l^j is provided in appendix 4.5.

Let $n_{k+1|k}^j$ be the number of neurons in j-th sub-population receiving one or more recurrent EPSPs. Then, we have

$$E[n_{k+1|k}^j | \{n_{k|k}^i\}] \;=\; \mathcal{L} \sum_{i=1}^{\mathcal{X}} W_{ij}\, n_{k|k}^i$$

$$=\; \mathcal{L} \frac{N_k}{C_W} \omega_{k+1|k}^i \tag{2.23}$$

$$\mathrm{Var}[n_{k+1|k}^j | \{n_{k|k}^i\}] \;\simeq\; \mathcal{L} \frac{N_k}{C_W} \omega_{k+1|k}^i \tag{2.24}$$

Thus, the prediction probability is represented by the expected number of neurons that receive recurrent inputs, as shown in figure 2.6.

In the model, recurrent inputs alone are not strong enough to make the inference neurons fire – these inputs leave the neurons partially activated. We can view these partially activated neurons as the "proposed" samples drawn from the prediction density $P(X_{k+1}|X_k)$. To correct the prediction distribution based on the current observation, these proposed samples are accepted with a probability proportional to the observation likelihood $P(Z_{k+1}|X_{k+1})$ when the new measurement Z_{k+1} becomes available. This implements a form of "rejection sampling" used in sequential Monte Carlo algorithms [51]. Neurally, this is implemented by feedforward inputs from sensory neurons (which receive Z_{k+1}) causing neurons to spike when coincident with recurrent inputs. Thus, the inference neurons act as coincidence detectors which fire if and only if both recurrent and sensory inputs are received:

$$s_l^j(k+1) = sgn(a_l^j(k+1) \times b_l^j(k+1)) \tag{2.25}$$

where the sign function $sgn(x) = 1$ only when $x > 0$. The feedforward input b_l^j represents the number of EPSPs caused by sensory inputs. Equation 2.25 defines the output of an abstract model neuron. In section 4.2 we show that such abstract model neurons can be implemented using leaky-integrate-and-fire (LIF) dynamics.

Note that $P(s_l^j(k+1) = 1) \propto P(X_{k+1} = x^j|Z_{1:k+1})$ if and only if $P(b_l^j(k+1) = 1) \propto g(Z_{k+1}|X_{k+1} = x^j)$. In other words, the hidden-layer inference neurons will spike with probability proportional to the updated posterior distribution if and only if the feedforward input $\{b_l^j\}$ arrives with probability proportional to the likelihood of observations. We now examine what feedforward weight matrix M between the sensory neurons and inference neurons achieves such a requirement.

The noisy measurement Z_{k+1} is not directly observed by the inference neurons, but sensed through an array of $\mathcal{Z}$ sensory neurons, whose receptive fileds are centered at $z^i \in \mathbb{Z}, i = 1, \ldots, \mathcal{Z}$. We assume for simplicity that receptive fields of sensory neurons do not overlap with each other (appendix 4.6 discusses the more general overlapping case). Again we define the feedforward weight M_{ij} to be the synaptic release probability between sensory neuron i and inference neurons in the j-th sub-population. A spiking sensory neuron i causes an EPSP in a neuron in the j-th sub-

population with probability M_{ij}. When $Z_{k+1} = z^i$ arrives, the sensory neuron centered at z^i emits a spike at time k, causing a feedforward EPSP in each of its post-synaptic neurons with probability proportional to the likelihood:

$$P(b_l^i(k+1) = 1) = g(Z_{k+1}|x^i)/C_M \tag{2.26}$$

where C_M is a scaling constant such that $M_{ij} = g(Z_{k+1} = z^i|x^j)/C_M$.

Finally, a inference neuron fires a spike at time $k+1$ if and only if it receives both recurrent and sensory inputs. The corresponding firing probability is then the product of the probabilities of the two inputs:

$$\begin{aligned} P(s_l^i(k+1) = 1) &= P(a_l^i(k+1) \geq 1)P(b_l^i(k+1) \geq 1) \\ &= \frac{N_k}{C_W C_M} P(X_{k+1}|Z_{1:k}) g(Z_{k+1}|X_{k+1}) \\ &\propto P(X_{k+1}|Z_{1:k+1}) \end{aligned} \tag{2.27}$$

Let $n_{k+1|k+1}^i$ be the number of spikes in i-th sub-population at time $k+1$,

$$n_{k+1|k+1}^i = \sum_{l=1}^{\mathcal{L}} s_l^i(k+1) \tag{2.28}$$

$$\begin{aligned} E[n_{k+1|k+1}^i|\{n_{k|k}^i\}] &= \mathcal{L}\frac{N_k}{C_W C_M} g(Z_{k+1}|x^i)\omega_{k+1|k}^i \\ &= \mathcal{L}\frac{N_k}{C_W C_M} P(Z_{k+1}|Z_{1:k})\omega_{k+1|k+1}^i \end{aligned} \tag{2.29}$$

$$\begin{aligned} \mathrm{Var}[n_{k+1|k+1}^i|\{n_{k|k}^i\}] &= \sum_{l=1}^{\mathcal{L}} [\mathrm{Var}(a_l^i)\mathrm{Var}(b_l^i) + \mathrm{Var}(a_l^i)E(b_l^i)^2 + \mathrm{Var}(b_l^i)E(a_l^i)^2] \\ &\simeq \mathcal{L}\frac{N_k}{C_W C_M} g(Z_{k+1}|x^i)\omega_{k+1|k}^i \end{aligned} \tag{2.30}$$

Equation 2.29 ensures that the expected spike distribution at time $k+1$ is a Monte Carlo approximation to the updated posterior probability $P(X_{k+1}|Z_{1:k+1})$. It also determines how many neurons are activated at time $k+1$. To keep the number of spikes at different time steps relatively constant, the scaling constant C_W and the number of neurons $\mathcal{L}$ could be of the same order of magnitude: for example, $C_W = \mathcal{L}$. Note that approximations in equations 2.22, 2.24 and 2.30

become exact when $\frac{N_k^2}{C_W^2} \to 0$. This implies a form of sparse coding: although the number of neurons in the network may be large, only a small fraction of neurons are activated.

In summary, we have proposed a spiking network model that approximates Bayesian filtering using spikes as Monte Carlo samples of probability distributions. We assume that the transition (dynamics) and emission (obsevation) models are known and encoded in the recurrent weights W and feedforward weights M, respectively. In addition, we assume that the network employs a sparse coding strategy: the total neuronal activity N_k at any time step is small compared to the number of neurons $\mathcal{L}$ in a sub-population. The model does not put any contraints on the particular form of the probability density over hidden world state. When the state space $\mathbb{X}$ is discrete, the spiking network provides the optimal Bayesian solution when the above assumptions hold.

4.2 LIF Implementation and Results

In this section, we demonstrate that the network model can be implemented using leaky integrate-and-fire neurons, which are commonly used to model CNS neurons. Model parameters are chosen to reflect those reported for biological neurons.

Figure 2.7 shows the dynamics of an example neuron. Let v_i be the membrane potential of a neuron whose preferred state is x^i.

$$\tau_m \frac{dv_i}{dt} = -v_i + R \times (I^S(t) + I^R(t)) \tag{2.31}$$

where τ_m is the membrane time constant and R is the input resistance. The neuron spikes when $v_i(t) > v_{\text{th}}$. Note that the time variable t is continuous, while the HMM time variable k is discrete. Suppose the size of the HMM time step is Δ_{hmm}. We define $Z_t = Z_k$ and $X_t = X_k$ if $(k-1)\Delta_{\text{hmm}} < t \leq k\Delta_{\text{hmm}}$. $n^i_{k|k}$ represents the spike count in the time interval $((k-1)\Delta_{\text{hmm}}, k\Delta_{\text{hmm}}]$ over neurons in the i-th sub-population. If an LIF neuron in the i-th sub-population fires at time t, $(k-1)\Delta_{\text{hmm}} < t \leq k\Delta_{\text{hmm}}$, then it evokes a recurrent EPSP in the j-th sub-population at time $t' = t + \Delta_{\text{hmm}}$, with probability W_{ij}. A neuron also receives sensory EPSPs, whose arrival probability is proportional to $M_{ij}dt$. $I^R(t)$ and $I^S(t)$ represent the accumulated recurrent and

sensory inputs respectively. Using the notation $\iota = R$ or S, we have:

$$I^{\iota}(t) = \frac{\alpha^{\iota}}{\tau_{\iota}} \sum_{\nu=1}^{\nu} \exp(-(t - t_{\nu}^{\iota})/\tau^{\iota})\Theta(t - t_{\nu}^{\iota}) \tag{2.32}$$

where τ_{ι} is the synaptic time constant, α^{ι} is the amplitude of synaptic input, and $\{t_1^{\iota}, \ldots, t_{\nu}^{\iota}\}$ are the the arrival times of pre-synaptic spikes. The Heaviside step function $\Theta(t)$ ensures causality.

The normalized EPSP evoked by one input spike (either sensory or recurrent) mimics the effect of an 'alpha' synapse:

$$\epsilon^{\iota}(\tau) \propto \frac{\exp(-\tau/\tau_m) - \exp(-\tau/\tau_{\iota})}{\tau_m - \tau_{\iota}}\Theta(\tau); \quad \tau = t - t_{\nu}^{\iota}; \tau_m > \tau_{\iota} \tag{2.33}$$

with $\max_{\tau} \epsilon(\tau) = 1$. The synaptic constants τ_{ι} are smaller than the membrane time constants τ_m [64, 144], e.g., $\tau_{\iota} = 1$ms and $\tau_m = 8$ms. Thus, one can drop the dependence of $v_i(t)$ on the arrival times of past spikes except for the most recent sensory and recurrent spikes.

Let $t_{\hat{\zeta}}^{\iota} = \max\{t_{\zeta}^{\iota} | t_{\zeta}^{\iota} < t\}$ and $t_0 = \min(t_{\hat{\zeta}}^{R}, t_{\hat{\zeta}}^{S})$. Then:

$$v_i(t) = v_i(t_0) + \alpha^R \epsilon^R(t - t_{\hat{\zeta}}^{R}) + \alpha^S \epsilon^S(t - t_{\hat{\zeta}}^{S}); \quad t_0 \leq t < \min(t_{\hat{\zeta}+1}^{R}, t_{\hat{\zeta}+1}^{S}) \tag{2.34}$$

For the model to perform Bayesian filtering correctly, the LIF neuron should fire when there is coincident recurrent and sensory input and minimize firing for a sequence of spikes of one type. Due to the sparseness of the network, the proportionality constants C_W and C_M can be chosen such that interspike intervals between two input spikes of the same type are much greater than the membrane time constant: $t_{\zeta+1}^{\iota} - t_{\zeta}^{\iota} \gg \tau_m$. This helps reduce the probability of spiking for multiple spikes of the same type. We also choose α^R and α^S such that $\max(v_i(t)) > v_{\text{th}}$ only when $|t_{\hat{\zeta}}^{R} - t_{\hat{\zeta}}^{S}| \leq \Delta_{\text{cd}}$, where Δ_{cd} is the coincidence detection window. We then obtain a LIF model neuron that fires only if it receives both sensory and recurrent inputs within Δ_{cd}. Finally, we require that $\Delta_{\text{cd}} < \Delta_{\text{hmm}}$ to ensure that the neuron's spiking probability is proportional to the product of likelihood $P(Z_k|X_k)$ and the prediction probability $P(X_k|Z_{1:k-1})$, which in turn is proportional to the posterior probability $P(X_k|Z_{1:k})$.

The simple model above can be extended to handle the case of multiple spikes of the same type (the cases where $t_{\zeta+1}^{\iota} - t_{\zeta}^{\iota}$ is small) by adding the mechanism of short-term synaptic depression

(STSD) to the model. STSD usually occurs in cortical neurons due to depletion of synaptic vesicles [169]. The amplitude of the $\hat{\zeta}$-th input in the presence of rapid STSD can be modeled by [153]: $\alpha^{\iota}_{\hat{\zeta}} = \alpha^{\iota}_{\max}[1 - \exp(-(t^{\iota}_{\hat{\zeta}} - t^{\iota}_{\hat{\zeta}-1})/\tau_m)]$. Then the maximum response to two successive EPSPs from the same synapse can be simplified as follows:

$$\begin{aligned} v_i(t) &= v_i(t_{\hat{\zeta}-1}) + \alpha^{\iota}_{\hat{\zeta}}\epsilon^{\iota}(t - t^{\iota}_{\hat{\zeta}}) \\ &\leq \alpha^{\iota}_{\max}\epsilon^{\iota}(t^{\iota}_{\hat{\zeta}} - t^{\iota}_{\hat{\zeta}-1}) + \alpha^{\iota}_{\hat{\zeta}}\epsilon^{\iota}(t - t^{\iota}_{\hat{\zeta}}) \\ &\leq \alpha^{\iota}_{\max}\exp(-(t^{\iota}_{\hat{\zeta}} - t^{\iota}_{\hat{\zeta}-1})/\tau_m) + \alpha^{\iota}_{\max}[1 - \exp(-(t^{\iota}_{\hat{\zeta}} - t^{\iota}_{\hat{\zeta}-1})/\tau_m)] = \alpha^{\iota}_{\max} \end{aligned} \tag{2.35}$$

As a result, even if a model neuron receives more than one input spikes from either sensory or recurrent synapses in a short period of time, successive spikes will not further depolarize the membrane potential due to short-term synaptic depression. Such a voltage saturation effect has also been experimentally observed at pyramidal-pyramidal cell connections in adult rat neocortex (e.g., figure 6 of [151]).

In figure 2.7, we show an example trajectory of the membrane potential $v_i(t)$. The model parameters were chosen to be consistent with those reported in typical CNS neurons [64, 144]: refractory time period $= 2.5$ms, $\tau_m = 8.33$ms, $\tau_R = \tau_S = 1$ms, $R = 138.8$ M Ω, $\alpha^R = \alpha^S = 0.6$nA and $v_{\text{th}} = 15$mV. The model neuron fires only if the sensory and recurrent inputs arrive within a time window $\Delta_{\text{cd}} = 0.6$ms. Note that short-term synaptic depression guarantees that the neuron will not fire even when the interspike interval between two recurrent spikes is 1 ms. Thus we have an LIF model neuron that is equivalent to the binary neuron described in section 4.1

In the following two sections, we illustrate how a network of such LIF neurons can perform Bayesian inference for two different tasks and compare the simulation results with biological data.

4.3 Static World State: An Example from Sensory Adaptation

We first consider the special case where the dynamics of the hidden state is static and where Bayesian filtering reduces to Kalman filtering. We relate this abstract model to neural data and show how the network introduced above for Bayesian inference can expain the data.

Let $X_k \in \mathbb{R}$ be the mean light intensity (luminance) of a static visual stimulus, $X_k = x^0, \forall\, k, 1 \leq k \leq K_0$. The measurements $Z_k \in \mathbb{R}$ are the intensities of the time-varying noisy stimulus observed by the retina, with standard deviation (contrast) σ_Z: $(Z_k - X_k) \sim N(0, \sigma_Z^2)$. The estimated mean and variance of the posterior distribution over X_k, given past inputs, can be described using a Kalman filter [137]:

$$E[X_k] = \frac{E[X_{k-1}] \times \sigma_Z^2 + Z_k \times \mathrm{Var}[X_{k-1}]}{\mathrm{Var}[X_{k-1}] + \sigma_Z^2} \tag{2.36}$$

$$\frac{1}{\mathrm{Var}[X_k]} = \frac{1}{\sigma_Z^2} + \frac{1}{\mathrm{Var}[X_{k-1}]} \tag{2.37}$$

Equation 2.36 has an intuitive explanation: the mean at time k is the weighted average of the previous mean $E[X_{k-1}]$ and the current observation Z_k, each weight corresponding to the variance of the other component. Thus, if there is more noise in the sensory input (higher σ_Z^2), more weight is given to the previous mean $E[X_{k-1}]$, and vice versa. Also, from equation 2.37, we have $\frac{\mathrm{Var}[X_k]}{\mathrm{Var}[X_{k-1}]} = \frac{\sigma_Z^2}{\sigma_Z^2 + \mathrm{Var}[X_{k-1}]} < 1$. Thus, the variance of X_k decreases with time k, and will eventually converge to zero as $k \to \infty$.

Now consider the situation where the hidden variable X_k is suddenly switched to another state after time step K_0: $X_k = x^1$ for $k > K_0$. Since X_k is hidden and the system is unaware of this change, the system continues to apply equations 2.36 and 2.37 for $k > K_0$. Thus, starting with mean $E[X_{K_0}]$ and variance $\mathrm{Var}[X_{K_0}]$, and combining equations 2.36 and 2.37, we obtain :

$$\begin{aligned}
E[X_k] &= E[X_{k-1}] \times \frac{\mathrm{Var}[X_k]}{\mathrm{Var}[X_{k-1}]} + Z_k \times \frac{\mathrm{Var}[X_k]}{\sigma_Z^2} \\
&= (E[X_{k-2}]\frac{\mathrm{Var}[X_{k-1}]}{\mathrm{Var}[X_{k-2}]} + Z_{k-1}\frac{\mathrm{Var}[X_{k-1}]}{\sigma_Z^2}) \times \frac{\mathrm{Var}[X_k]}{\mathrm{Var}[X_{k-1}]} + Z_k \times \frac{\mathrm{Var}[X_k]}{\sigma_Z^2} \\
&= E[X_{k-2}] \times \frac{\mathrm{Var}[X_k]}{\mathrm{Var}[X_{k-2}]} + (Z_{k-1} + Z_k) \times \frac{\mathrm{Var}[X_k]}{\sigma_Z^2} \\
&\cdots \\
&= E[X_{K_0}] \times \frac{\mathrm{Var}[X_k]}{\mathrm{Var}[X_{K_0}]} + \sum_{s=K_0+1}^{k} Z_s \times \frac{\mathrm{Var}[X_k]}{\sigma_Z^2}
\end{aligned} \tag{2.38}$$

$$\begin{aligned}
\frac{1}{\mathrm{Var}[X_k]} &= \frac{k - K_0}{\sigma_Z^2} + \frac{1}{\mathrm{Var}[X_{K_0}]} \\
\mathrm{Var}[X_k] &= \frac{\sigma_Z^2 \mathrm{Var}[X_{K_0}]}{\sigma_Z^2 + (k - K_0)\mathrm{Var}[X_{K_0}]}
\end{aligned} \tag{2.39}$$

We see that the mean $E[X_k]$ is a weighted average of the prior mean $E[X_{K_0}]$ and the new observations $\{Z_s\}$. If $\mathrm{Var}[X_{K_0}]$ is small, more weight is given to the prior estimate $E[X_{K_0}]$. The prior estimates $E[X_{K_0}]$ and $\mathrm{Var}[X_{K_0}]$ are determined by the time of transition K_0. For example, when the initial variance $\mathrm{Var}[X_0] = \infty$, we have $\frac{1}{\mathrm{Var}[X_0]} = 0$ and $\mathrm{Var}[X_k] = \sigma_Z^2/k$ from equation 2.37 initial state x^0, the lesser $\mathrm{Var}[X_{K_0}]$ becomes as the system accumulates more evidence for x^0. Thus, when the state is changed at time step $K_0 + 1$, it takes longer for $E[X_k]$ to converge to the new state.

Figure 2.8(a) shows three examples of temporal evolutions of $E[X_k]$ (red traces) for different values for K_0 (note the different scales on the time axis). All three trajectories display a form of exponential-like dynamics after K_0, with a half-life $\approx K_0$.

The phenomena discussed above can be interpreted as sensory adaptation, a key property exhibited by the brain. Efficient coding of the sensory world requires that the brain optimally estimate and adapt to the statistics of its sensory inputs [5]. In the example above, this corresponds to the estimation of X_k from noisy observations Z_k. A switch from x^0 to x^1 is equivalent to an abrupt change in luminance of the environment, e.g, a sudden exposure to bright daylight when coming out of a dark movie theatre. The model above suggests that the time course of "adaptation" of $E[X_k]$ to the new state x^1 is determined by the duration of the prior state x^0. This is consistent with previous observations that the dynamics of the adaptation process could be dependent on stimulus history [57, 157]. Figure 2.8(b) shows the mean synaptic current to an ON retinal ganglion cell (RGC) elicited by periodic switches between low luminance and high luminance stimuli. The time course of adaptation is dependent on the switching period K. The longer the retina is exposed to the low luminance environment, the slower the time course of adaptation to the high luminance. [157] argued that the neural response in RGC encodes the mean of the posterior distribution of the visual stimulus X_k. They hypothesized that sensory adaptation involves Bayesian inference of stimulus parameters and suggested that the visual system may employ a form of Kalman filter.

The spiking network model we have proposed can be used to model sensory adaptation phenomena such as those reported by [157]. Sensory neurons in the model measure the noisy light intensity Z_k. The inference layer LIF neurons combine this sensory likelihood information with re-

current inputs to obtain a grid-based approximation $\{\hat{P}_k^i\}$ of the posterior distribution of luminance X_k. If the model network correctly implements Bayesian filtering, one would expect the posterior mean $\sum_{i=1}^{\mathcal{X}} x^i \hat{p}_k^i$ approximates the predictions from a Kalman filter. This is indeed the case which can be seen in figure 2.8(a) (blue trace).

4.4 Dynamic World State: Adaptation in Hippocampal Place Cells

Consider an experiment where a rat moves along a linear track. Let $X_t \in \Omega = 1, \ldots N$ be the position of the rat along the track. The motion is deterministic such that the transition probability matrix f defined by $\delta(i, i+1)$ for $1 \leq i < N$, with a reset to the start position upon reaching the end of the track. The matrix f is unknown and the measurement of position X_t is noisy: $Z_t = X_t + \eta_t$, where Z_t is the observable input to the sensory system and η_t is white noise with variance σ_Z^2. The initial recurrent weights (at time 0) are set to be zero mean Gaussian with width σ_{prior}, i.e., $W_{ij}(0) = \exp(-(i-j)^2/(2\sigma_{\text{prior}}^2))$, a biased estimator of f.

Figure 2.9(a) shows the recurrent weights $W(t)$ learned using equation 2.16 after 10 laps. The synaptic weights $\mathbf{W}_j$ become asymmetric and their centers show a backward shift after learning.[1] A similar backward shift has been reported in rat hippocampal place cells [105], as shown in figure 2.9(b).

4.5 Probability Distribution of the Synaptic Inputs a_l^j

The synaptic input a_l^j is the number of EPSPs received by the l-th posterior neuron in the j-th sub-population. Since the recurrent network in the posterior population is fully connected, each spiking neuron that fired in the previous time step will attempt to send an EPSP to its neighbors with success probability W_{ij}. Therefore, $a_l^j(k+1)$ can be view as the sum of N_k independent, but

[1] The recurrent weights W in the model need not necessarily correspond to a single set of synaptic weights in the hippocampus but could instead capture the effect of a larger multi-synaptic loop such as the hippocampal-entorhinal network.

not identically distributed Bernoulli trials. Dropping all unnecessary indices, we have

$$a = \sum_{m=1}^{N} \epsilon_m \tag{2.40}$$

where each binary random variable ϵ_m has a success probability $P(\epsilon_m = 1) = P_m$. $P_m = W_{ij}$ when ϵ_m represents neurotransmitter release from cells in sub-population i to sub-population j. a has the so-called "Poisson binomial" distribution. $P(a)$ can be approximated by a Poisson distribution P_λ where $\lambda = \sum_m p_m$. In the familiar case ϵ_m are i.i.d, $P_m = p$ for all m, and a will have the exact P_λ distribution with $\lambda = Np$.

Let Y be a random variable that follows the Poisson distribution with $E(Y) = \sum_m P_m$ and let

$$D = \sup_u |P(a \geq u) - P(Y \geq U)| \tag{2.41}$$

be the maximum absolute different between the two cumulative probability distributions. Hodges et al. [76] showed that $D \leq 2\sum p_m^2$ and $D \geq 3\sqrt[3]{\alpha}$ where $\alpha = \max_m P_m$.

In our network implementation, $P_m = \frac{1}{C_W} f_{ij}$. Therefore $\sum P_m^2 \leq \frac{N_k}{C_W^2}\alpha$. Since C_W should have the same order as the network size $\mathcal{L}$, the approximation becomes exact as $\frac{N_k}{C_W^2} \to 0$, which corresponds to a sparse spiking network (large C_W and $\mathcal{L}$) with finite energy budget (finite N_k). In this case, a has the distribution

$$P(a = u) = \frac{u^\lambda}{u!} \exp(-\lambda) \tag{2.42}$$

$\lambda = \sum P_m \leq \frac{N_k}{C_W}\alpha$. Since N_k is finite, we have $\lambda^2 \to 0$. $P(a \geq 1) = 1 - exp(-\lambda) \to \lambda$. This corresponds to equation 2.22 in the text. $\lambda^2 \to 0$ also implies that $P(a > 1) \to 0$. The probability that the neuron receives more than one EPSP vanishes in the sparse network. This mechanism is similar to a winner-take-all (WTA) [99] network, where multiple pre-synaptic neurons compete to activate one post-synaptic neuron.

4.6 Sensory neurons

The noisy measurement Z_{k+1} is not directly observed by the inference neurons, but sensed through another array of $\mathcal{Z}$ sensory neurons, whose receptive fields are centered at $z^i \in \mathbb{Z}, i = 1, \ldots, \mathcal{Z}$.

Each sensory neuron i generates a Poisson spike train, with intensity proptional to $h_i(Z_{k+1})$. The probability that the i-th sensory neuron fires at time $k+1$ is proportional to $h_i(Z_{k+1})$, if the window of coincidence detection is small. Again we define the feedforward weight M_{ij} to be the neurotransmitter release probability between sensory neuron i and inference neurons in the j-th sub-population. A spiking sensory neuron i sends an EPSP to inference neurons in the j-th sub-population with probaility M_{ij}. Therefore, as in equation 2.22, the probability that neurons in the j-th sub-population receive feedforward inputs at time $k+1$ can be approximated by

$$P(b_l^j(k+1) \geq 1) \simeq \sum_{i=1}^{\mathcal{Y}} M_{ij} h_i(Z_{k+1}) = g(Z_{k+1} = z^{i'}|x^j) \tag{2.43}$$

Let $G = \{G_{ij} = g(Z_{k+1} = z^{i'}|x^j)\}$ be a $\mathcal{Y}$ by $\mathcal{X}$ matrix, and $H = \{H_{ii'} = h_i(Z_{k+1} = z^{ii'})\}$ be a $\mathcal{Y}$ by $\mathcal{Y}$ matrix. Equation 2.43 implies $G \propto H \times M$. Therefore, the $\mathcal{Y}$ by $\mathcal{X}$ feedforward weight matrix $M \propto H^{-1} \times G$.

Typical choices of the "tuning curve" function h_i for sensory neurons are radial basis functions with a peak value at the center of the receptive field z^i, e.g., Gaussian $h_i(Z_{k+1}) = h(Z_{k+1} - z^i) = \exp[-(Z_{k+1} - z^i)^2/\sigma_Z^2]$ or cosine tuning functions. However, for simplicity, we may require only one sensory neuron responds exclusively to Z_{k+1} at time $k+1$ such that $h_i(Z_{k+1}) = \chi(\frac{z^i + z^{i+1}}{2} \leq Z_{k+1} \leq \frac{z^i + z^{i+1}}{2})$, where $\chi(.)$ denotes the indicator function. In this case, receptive fields of sensory neurons do not overlap with each other, H is an identity matrix, and $M \propto G$. Moreover, the approximation in equation 2.43 becomes exact. When $Z_{k+1} \approx z^i$ arrives, only one sensory neuron centered at z^i is activated and fires one spike at time k. This pre-synaptic neuron then sends one feedforward EPSP randomly to every post-synaptic inference neuron with probability proportional to the likelihood:

$$P(b_l^i(k+1) = 1) = g((Z_{k+1}|x^i)/C_M \tag{2.44}$$

where C_M is another scaling constant such that $M_{ij} = g((Z_{k+1} = z^i|x^j)/C_M$.

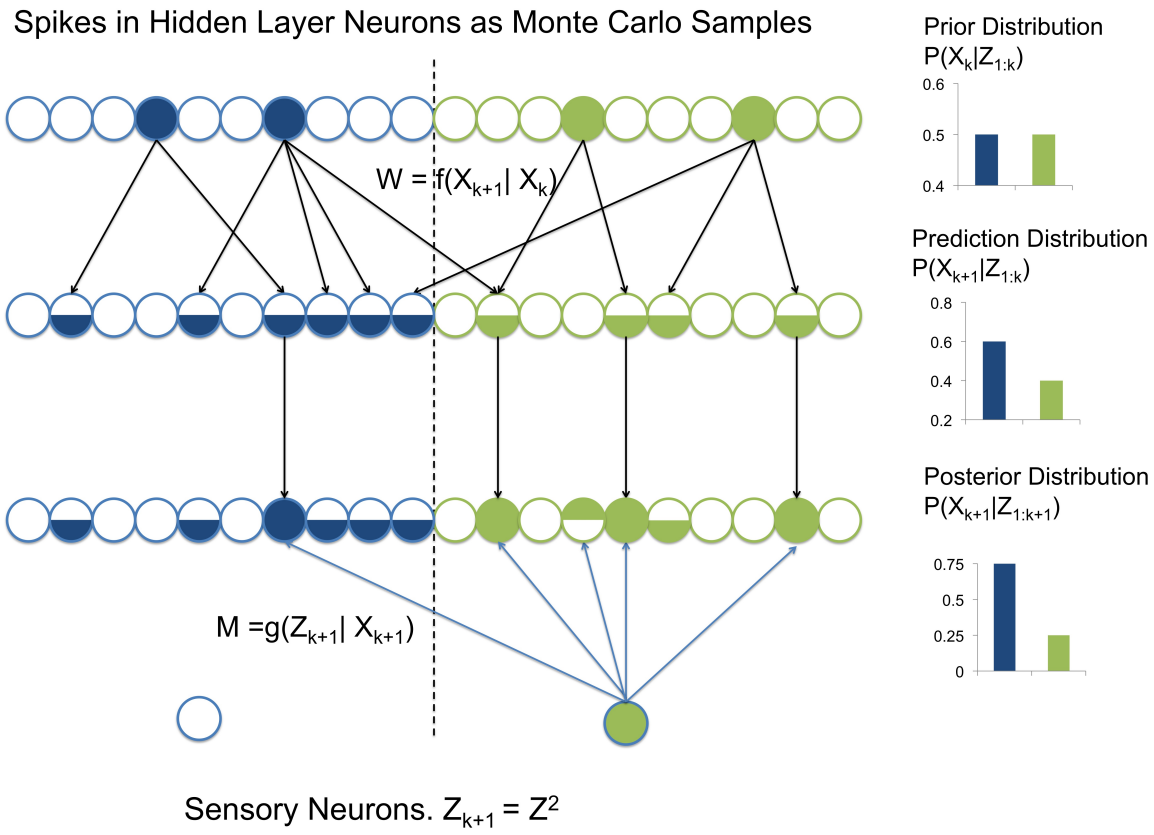
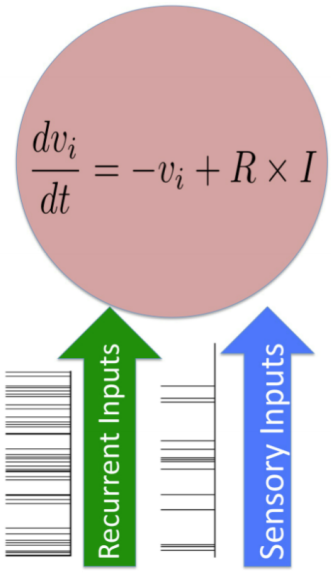
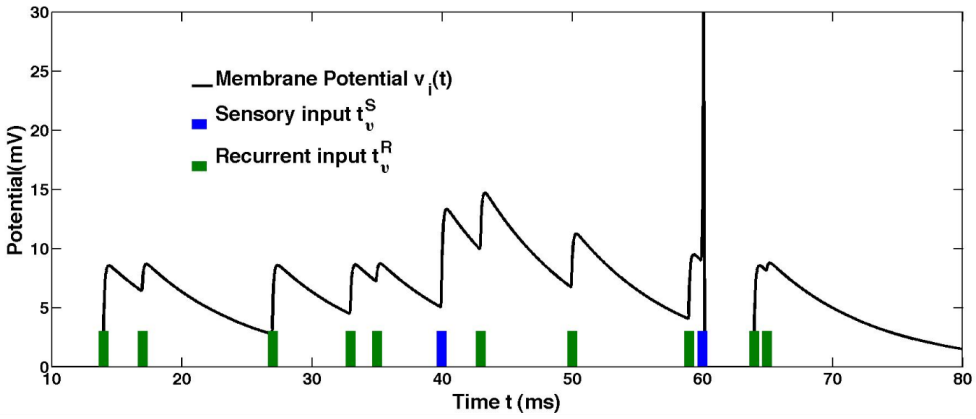

Spikes in Hidden Layer Neurons as Monte Carlo Samples
W = f(X_{k+1}| X_k)
M =g(Z_{k+1}| X_{k+1})
Sensory Neurons. Z_{k+1} = Z^2
Prior Distribution
P(X_k|Z_{1:k})
0.6
0.5
0.4
Prediction Distribution
P(X_{k+1}|Z_{1:k})
0.8
0.6
0.4
0.2
Posterior Distribution
P(X_{k+1}|Z_{1:k+1})
0.75
0.5
0.25
0

Figure 2.6 ***(preceding page).*** **Graphical Representation of spike distribution propagation.** Here, $\mathcal{X} = \mathcal{Z} = 2$ and $\mathcal{L} = 10$. At time k, spikes (shown as filled circles in the top row) in the posterior population represent the distribution $P(X_k|Z_{1:k})$. With recurrent weights $W \propto f(X_{k+1}|X_k)$, spiking neurons send EPSPs to their neighbors and make them partially activated (shown as half-filled circles in the second row). The distribution of partially activated neurons is a Monte-Carlo approximation to the prediction distribution $P(X_{k+1}|Z_{1:k})$. When a new observation Z_{k+1} arrives, sensory input neurons send feedforward EPSPs to the inference neurons using synaptic weights $M = g(Z|X)$. The inference neurons at time $k+1$ fire only if they receive both recurrent and feedforward inputs. With the firing probability proportional to the product of prediction probability $P(X_{k+1}|Z_{1:k})$ and observation likelihood $g(Z_{k+1}|X_{k+1})$, the spike distribution at time $k+1$ again represents the updated posterior $P(X_{k+1}|Z_{1:k+1})$.



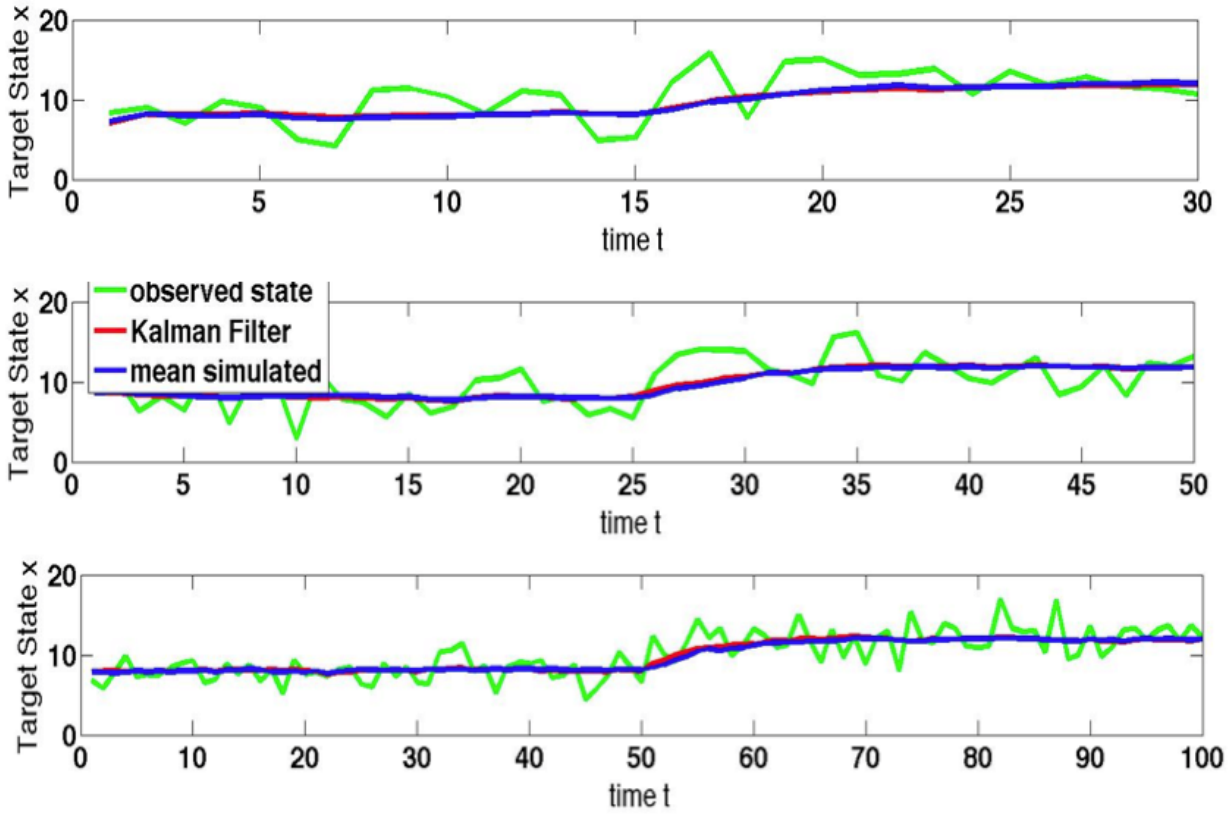
(a)



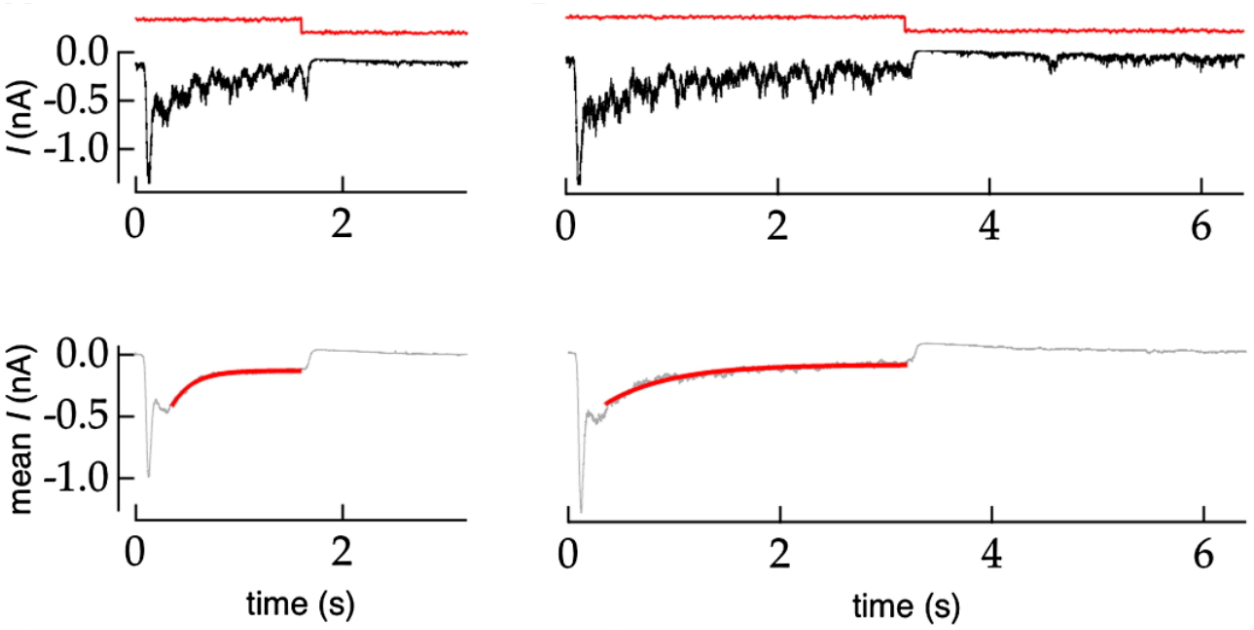
(b)

Figure 2.7 ***(preceding page).*** **Model LIF neuron.** (a) LIF neuron receiving inputs from recurrent and sensory synapses. (b) The black curve shows an example trajectory of the membrane potential. Green and blue bars represent the arrival times of recurrent and sensory spikes respectively.

[

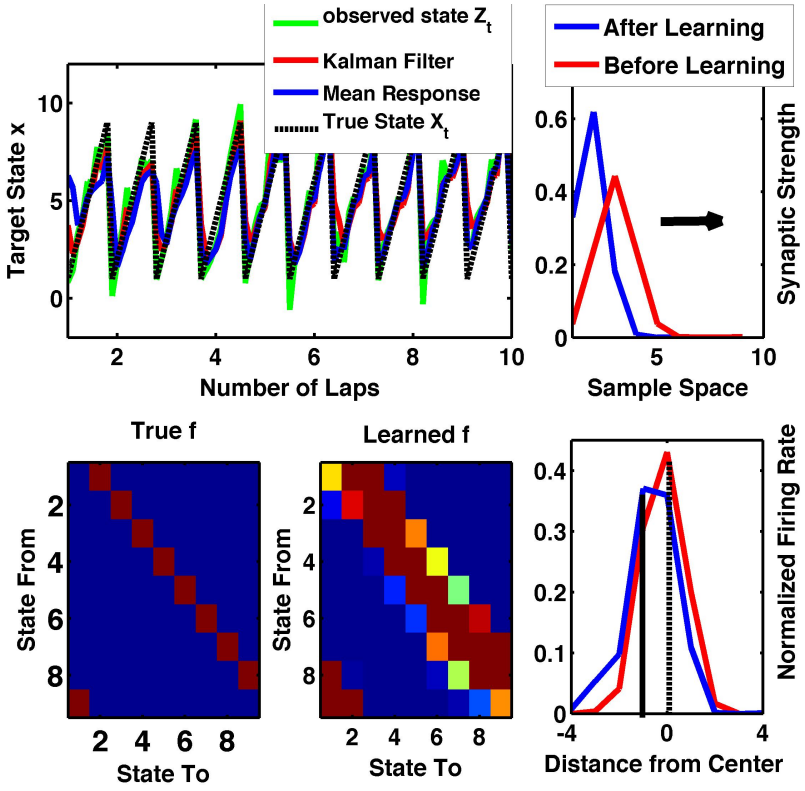

Target State x
time t
observed state
Kalman Filter
mean simulated

(a)

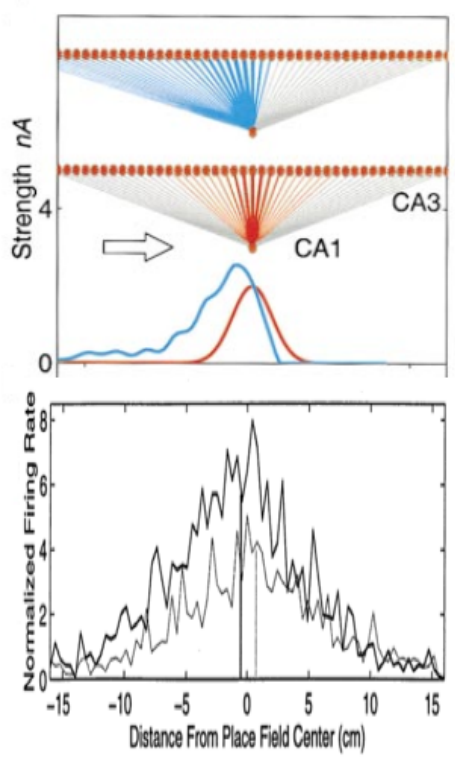

I (nA)
mean I (nA)
time (s)

(b)

Figure 2.8 ***(preceding page).*** **Sensory Adaptation and Bayesian Filtering.** (a) The hidden state (luminance) was switched from one value to another at specific time instants (time step 15, 25, and 50 respectively in the plots). The green curve represents the noisy stimuli Z_t available to the system, the red curve shows the estimation of X_t using the Kalman filter equation 2.36, and the blue curve displays the posterior mean $\sum_{i=1}^{\mathcal{X}} x^i \hat{p}_k^i$ computed from the spiking LIF network model. Note the similarity in the time course of adaptation across different time scales (different scales on time axis for the three plots). (b) Above: Time course of excitatory synaptic input to a retinal ganglion cell (black trace) in response to a single cycle of stimulus (red trace). Below: Mean synaptic current over approximately 50 trials as above. The embedded red curve is the exponential fit to the adaptation. Compare with the red and blue curves in (a). (Plots in (b) are from [157])


observed state Z_t
Kalman Filter
Mean Response
True State X_t
Target State x
Number of Laps
After Learning
Before Learning
Synaptic Strength
Sample Space
True f
Learned f
State From
State To
Normalized Firing Rate
Distance from Center

(a)


Strength nA
CA3
CA1
Normalized Firing Rate
Distance From Place Field Center (cm)

(b)

Figure 2.9 ***(preceding page).*** **Adaptation in Hippocampal Place Cells.** (a) Upper left: estimated X_k, noisy observation Z_k and the prediction from Kalman filter are shown in blue, green and red, respectively. Upper right: comparison between the learned $\mathbf{W}_3(T)$ with the initial $\mathbf{W}_3(0)$ after 10 laps. Bottom left: true transition matrix f. Bottom middle: learned recurrent weight matrix $W(T)$. Bottom right: Normalized firing rate during the first and the last lap. Model parameters: $N = 9$, $\sigma_Z = 0.1 \times N$ and $\sigma_{\text{prior}} = 0.1 \times N$ (b) Top: (Figure from [105]) Computational Model of CA3→CA1 network. The synaptic weight matrix shifts backward as the rat moves forward. Bottom: (Figure from [104]) Histograms of firing rates in place cells recorded from rats during the first and the last lap. The center of the place field shifted backwards after learning.

Chapter 3

REWARD OPTIMIZATION IN THE PRIMATE BRAIN: A PROBABILISTIC MODEL OF DECISION MAKING UNDER UNCERTAINTY

Abstract

A key problem in neuroscience is understanding how the brain makes decisions under uncertainty. Important insights have been gained using tasks such as the random dots motion discrimination task in which the subject makes decisions based on noisy stimuli. A descriptive model known as the drift diffusion model has previously been used to explain psychometric and reaction time data from such tasks but to fully explain the data, one is forced to make ad-hoc assumptions such as a time-dependent collapsing decision boundary. We show that such assumptions are unnecessary when decision making is viewed within the framework of partially observable Markov decision processes (POMDPs). We propose an alternative model for decision making based on POMDPs. We show that the motion discrimination task reduces to the problems of (1) computing beliefs (posterior distributions) over the unknown direction and motion strength from noisy observations in a Bayesian manner, and (2) selecting actions based on these beliefs to maximize the expected sum of future rewards. The resulting optimal policy (belief-to-action mapping) is shown to be equivalent to a collapsing decision threshold that governs the switch from evidence accumulation to a discrimination decision. We show that the model accounts for both accuracy and reaction time as a function of stimulus strength as well as different speed-accuracy conditions in the random dots task.

1 Introduction

Animals are constantly confronted with the problem of making decisions given noisy sensory measurements and incomplete knowledge of their environment. Making decisions under such circumstances is difficult because it requires (1) inferring hidden states in the environment that are generating the noisy sensory observations, and (2) determining if one decision (or action) is better than another based on uncertain and delayed reinforcement. Experimental and theoretical studies [87, 167, 126, 124, 98, 52] have suggested that the brain may implement an approximate form of Bayesian inference for solving the hidden state problem. However, these studies typically do not address the question of how probabilistic representations of hidden state are employed in action selection based on reinforcement. Daw, Dayan and their colleagues [35, 37] explored the suitability of decision theoretic and reinforcement learning models in understanding several well-known neurobiological experiments. Bogacz and colleagues proposed a model that combines a traditional decision making model with reinforcement learning [20] (see also [93]). Rao [125] proposed a neural model for decision making based on the framework of partially observable Markov decision processes (POMDPs) [84]; the model focused on network implementation and learning but assumed a deadline to explain the collapsing decision threshold. Drugowitsch et al. [53] sought to explain the collapsing decision threshold by combining a traditional drift diffusion model with reward rate maximization. Other recent studies have used the general framework of POMDPs to explain experimental data in decision making tasks such as those involving a stop-signal [145, 146] and different types of prior knowledge [80].

In this paper, we derive from first principles a POMDP model for the well-known random dots motion discrimination task [143]. We show that the task reduces to the problems of (1) computing beta-distributed beliefs over the unknown direction and motion strength from noisy observations, and (2) selecting actions based on these beliefs in order to maximize the expected sum of future rewards. Without making ad-hoc assumptions such as a hypothetical deadline, a collapsing decision threshold emerges naturally via expected reward maximization. We present results comparing the model's predictions to experimental data and show that the model can explain both reaction time

and accuracy as a function of stimulus strength as well as different speed-accuracy conditions.

2 Methods

2.1 POMDP Framework

We model the random dots motion discrimination task as a POMDP. The POMDP framework assumes that at any particular time step, the environment is in a particular *hidden* state, μ, that is not directly accessible to the animal. This hidden state however can be inferred by making a sequence of sensory measurements. At each time step t, the animal receives a sensory measurement (observation), o_t, from the environment, which is determined by an $emission$ probability distribution $P(o_t|\mu)$. Since the hidden state μ is unknown, the animal must maintain a *belief* (posterior probability distribution) over the set of possible states given the sensory observations seen so far: $b_t(\mu|o_{1:t})$, where $o_{1:t}$ represents the sequence of observations that the animal has accumulated so far. At each time step, an action (decision) $a_t \in \mathcal{A}$ made by the animal can affect the environment by changing the current state to another according to a $transition$ probability distribution $P(\mu'|\mu, a_t)$ where μ is the current state, and μ' is a new state. The animal then gets a reward $R(\mu, a_t)$ from the environment, depending on the current state and the action taken. During training, the animal learns a policy, $\pi(b) \in \mathcal{A}$, which indicates which action a to perform for each belief state b. We make two main assumptions in the POMDP model. First, the animal uses Bayes rule to update its belief about the hidden state after each new observation o_{t+1}: $P(\mu|o_{1:t+1}) = \frac{P(\mu|o_{1:t}) \times P(o_{t+1}|\mu)}{P(o_{t+1}|o_{1:t})}$ [1]. Second, the animal is trained to follow an *optimal policy* $\pi^*(b)$ that maximizes the animal's expected total future reward in the task. Figure 3.1 illustrates the decision making process using the POMDP framework.

[1] In the decision making tasks that we model in this paper, the hidden state μ is fixed by experimenters within a trial and thus there is no transition distribution to include in the belief update equation. In general, the hidden state in a POMDP model follows a Markov chain, making the observations $o_{1:t}$ temporally correlated.

2.2 Random Dots Task as a POMDP

We now describe how the general framework of POMDPs can be applied to the random dots motion discrimination task as shown in Figure 3.1. In each trial, experimenter chooses a fixed direction $d \in \{-1, +1\}$ corresponding to leftward and rightward motion respectively, and a stimulus strength (motion coherence) $c \in [0, 1]$, where 0 corresponds to completely random motion and 1 corresponds to 100% coherent motion (i.e., all dots moving in the same direction). Intermediate values of c represent a corresponding fraction of dots moving in the coherent direction (e.g., 0.5 represents 50% coherent motion). The animal is shown a movie of randomly moving dots, a fraction c of which are moving in the same direction d.

In a given trial, neither the direction d nor the coherence c is known to the animal. We therefore regard (c, d) as the joint hidden environment state μ in the POMDP model. Neurophysiological evidence suggests that information regarding random dot motion is received from neurons in cortical area MT [112, 139, 23, 142]. Therefore, following previous models (e.g., [156, 102, 7]), we define the observation model $P(o_t|\mu)$ in the POMDP as a function of the responses of MT neurons. Let the firing rate of MT neurons preferring rightward and leftward direction be λ_R^{MT} and λ_L^{MT} respectively. We can define:

$$\begin{aligned}\lambda_R^{\mathrm{MT}}(c, d) &= \rho_{\mathrm{pref}} \frac{d+1}{2} c + \rho_{\mathrm{null}} \frac{1-d}{2} c + \lambda_0^{\mathrm{MT}} \\ \lambda_L^{\mathrm{MT}}(c, d) &= \rho_{\mathrm{pref}} \frac{1-d}{2} c + \rho_{\mathrm{null}} \frac{1+d}{2} c + \lambda_0^{\mathrm{MT}}\end{aligned} \tag{3.1}$$

where $\lambda_0^{\mathrm{MT}} = 20$ spikes/second is the average spike rate for 0% coherent motion stimulus, and $\rho_{\mathrm{pref}} = 40$ and $\rho_{\mathrm{null}} = -20$ are the "drive" in the preferred and null directions respectively. These constants (ρ_{pref}, ρ_{null} and λ_0^{MT}) are based on fits to experimental data as reported in [24, 102]. Let τ_t be the elapsed time between time steps t and $t+1$. Then, the number of spikes emitted by MT neurons r^{MT} within τ_t follows a Poisson distribution:

$$\Pr\left[r^{\mathrm{MT}}\right] = \frac{e^{-\lambda^{\mathrm{MT}}\tau_t}(\lambda^{\mathrm{MT}}\tau_t)^{r^{\mathrm{MT}}}}{r^{\mathrm{MT}}!}. \tag{3.2}$$

We define the observation o_t at time t as the spike count from MT neurons preferring rightward motion, given the total spike count from rightward and leftward-preferring neurons, i.e., the ob-

servation is a conditional random variable $o_t = r_R^{\mathrm{MT}}|n_t$ where $n_t = r_R^{\mathrm{MT}} + r_L^{\mathrm{MT}}$. Then o_t follows a stationary Binomial distribution $\mathrm{Bino}(n, \mu)$. Note that the duration of each POMDP time step need not be fixed, and we can therefore adjust τ_t such that $n_t = n$ for some fixed n, i.e., the animal updates the posterior distribution over hidden state each time it receives n spikes from the MT population. τ_t is exponentially distributed, and the standard deviation of τ_t will approach zero as n increases. When $n = 1$, o_t becomes an indicator random variable representing whether a spike was emitted by a rightward motion preferring neuron or not.

It can be shown [30] that o_t follows a Binomial distribution $\mathrm{Bino}(n, \mu)$ with

$$\mu = \frac{\lambda_R^{\mathrm{MT}}}{\lambda_R^{\mathrm{MT}} + \lambda_L^{\mathrm{MT}}} = \frac{\rho_{\mathrm{pref}} \frac{d+1}{2} c + \rho_{\mathrm{null}} \frac{1-d}{2} c + \lambda_0^{\mathrm{MT}}}{(\rho_{\mathrm{pref}} + \rho_{\mathrm{null}})\, c + 2\lambda_0^{\mathrm{MT}}} \tag{3.3}$$

$\mu \in [0, 1]$ represents the probability that the MT neurons favoring rightward movement will spike given that there is a spike in the MT population. Since μ is a joint function of c and d, we could equivalently regard it as the hidden state of our POMDP model: $\mu > 0.5$ indicates rightward direction ($d = +1$) while $\mu < 0.5$ indicates the opposite direction ($d = -1$). The coherence $c = 0$ corresponds to $\mu = 0.5$ while $c = 1$ corresponds to the two extreme values $\mu = 0$ or 1 for direction d being left or right respectively. Note that both direction d and coherence c are unknown to the animal in the experiments, but they are held constant within a trial.

2.3 Bayesian Inference of Hidden State

Given the framework above, the task of deciding the direction of motion of the coherently moving dots is equivalent to the task of deciding whether $d = 1$ or not, and deciding when to make such a decision. The POMDP model makes decisions based on the "belief" state $b_t(\mu) = P(\mu|o_{1:t})$, which is the posterior probability distribution over $\mu = \frac{cd+1}{2}$ given a sequence of observations $o_{1:t}$:

$$\begin{aligned} b_t(\mu) &= \frac{\Pr[o_t|\mu] \Pr[\mu|o_{1:t-1}]}{\Pr[o_t|o_{1:t-1}]} \\ &= \frac{\mu^{m_R(t)} (1-\mu)^{m_L(t)} \Pr[\mu]}{Pr[o_{1:t}]} \end{aligned} \tag{3.4}$$

where $m(t) = \sum_t n_t = n * t$, $m_R(t) = \sum_{\tau=1}^{t} o_\tau$, and $m_L(t) = m(t) - m_R(t)$. To facilitate the analysis, we represent the prior probability $\Pr[\mu]$ as a beta distribution with parameters α_0 and β_0.

Note that the beta distribution is quite flexible: for example, a uniform prior can be obtained using $\alpha_0 = \beta_0 = 1$. Without loss of generality, we will fix $\alpha_0 = \beta_0 = 1$ throughout this paper. The posterior distribution can now be written as:

$$\begin{aligned} b_t(\mu) &\propto \mu^{m_R+\alpha_0-1}(1-\mu)^{m_L+\beta_0-1} \\ &= \text{Beta}[\mu|\alpha = m_R + \alpha_0, \beta = m_L + \beta_0] \end{aligned} \tag{3.5}$$

The belief state b_t at time step t thus follows a beta distribution with two parameters α and β as defined above. Consequently, the posterior probability distribution over μ depends only on the number of spikes m_R and m_L for rightward and leftward motion respectively. These in turn determine $\hat{\mu}$ and t, where

$$\hat{\mu} = \frac{m_R + \alpha_0}{m_R + m_L + \alpha_0 + \beta_0} \tag{3.6}$$

is the point estimator of μ, and $t = \frac{m_R+m_L}{n}$. The animal only needs to keep track of $\hat{\mu}$ and t in order to encode the belief state $b_t = \text{Beta}[\mu|\alpha = \hat{\mu}(nt + \alpha_0 + \beta_0), \beta = (1-\hat{\mu})(nt + \alpha_0 + \beta_0)]$. After marginalizing over coherence c, we have the posterior probability over direction d:

$$\Pr[d = 1|o_{1:t}] = \int_{\mu=0.5}^{1} \text{Beta}(\mu|\alpha, \beta)d\mu = 1 - I_{0.5}(\alpha, \beta) \tag{3.7}$$

$$\Pr[d = -1|o_{1:t}] = \int_{\mu=0}^{0.5} \text{Beta}(\mu|\alpha, \beta)d\mu = I_{0.5}(\alpha, \beta). \tag{3.8}$$

where $I_x(\alpha, \beta) = \int_{\mu=0}^{x} \text{Beta}(\mu|\alpha, \beta)d\mu$ is the regularized incomplete beta function.

2.4 Actions, Rewards, and Value Function

The animal updates its belief after receiving the current observation o_t, and chooses one of the three actions (decisions) $a \in \{A_R, A_L, A_S\}$, denoting rightward eye movement, leftward eye movement, and sampling (i.e., waiting for one more observation) respectively. The model assumes the animal receives rewards $R(\mu, a)$ as follows (rewards are modeled using real numbers). When the animal makes a correct choice, $i.e.$, a rightward eye movement A_R when $d = 1$ ($\mu > 1/2$) or a leftward eye movement A_L when $d = -1$ ($\mu < 1/2$), the animal receives a positive reward $R_P > 0$. The animal

receives a negative reward (i.e., penalty) or nothing when an incorrect action is chosen $R_N \leq 0$. We further assume that the animal is motivated by hunger or thirst to make a decision as quickly as possible. This is modeled using a unit penalty $R_S = -1$ for each observation the animal makes, representing the cost the animal needs to pay when choosing the sampling action A_S.

Recall that a belief state b_t is determined by the parameters α, β. The goal of the animal is to find an optimal "policy" π^* that maximizes the "value" function $v^\pi(b_t)$, defined as the expected sum of future rewards given the current belief state:

$$v^\pi(b_t) = \mathbb{E}[\sum_{k=1}^{\infty} R(b_{t+k}, \pi(b_{t+k}))|b_t = \text{Beta}(\mu|\alpha, \beta)] \tag{3.9}$$

where the expectation is taken with respect to all future belief states $(b_{t+1}, \ldots, b_{t+k}, \ldots)$. The reward term $R(b_t, a)$ above is the expected reward for the given belief state and action:

$$\begin{aligned} R(b_t, A_S) &= nR_S \\ R(b_t, A_R) &= \sum_d \int_{c=0}^{1} R(c, d, A_R)\text{Beta}(\mu|\alpha, \beta)\text{dc} \\ &= R_P \times [1 - I_{0.5}(\alpha, \beta)] + R_N \times I_{0.5}(\alpha, \beta) \\ &= (R_P - R_N) \times [1 - I_{0.5}(\alpha, \beta)] + R_N \\ R(b_t, A_L) &= (R_P - R_N) \times I_{0.5}(\alpha, \beta) + R_N \end{aligned} \tag{3.10}$$

The above equations can be interpreted as follows. When A_S is selected, the animal receives n more samples at a cost of nR_S. When A_R is selected, the expected reward $R(b_t, A_R)$ depends on the probability density function of the hidden parameter μ given belief state b_t. With probability $I_{0.5}(\alpha, \beta)$, the true parameter μ is less than 0.5, making A_R an incorrect decision with penalty R_N, and with probability $1 - I_{0.5}(\alpha, \beta)$, action A_R is correct, earning the reward R_P.

2.5 Finding the Optimal Policy

A policy $\pi(b_t)$ defines a mapping from a belief state to one of the available actions a. A method for learning a POMDP policy by trial and error using the method of temporal difference (TD) learning

was suggested in [125]. Here, we derive a policy from first principles and compare the result with behavioral data.

One standard way [84] to solve a POMDP is to first convert it into a Markov Decision Process (MDP) over belief state, and then apply standard dynamical programming techniques such as value iteration [149] to compute the value function in equation 4.6. For the corresponding *belief MDP*, we need to define the transition probabilities $T(b_t \mid b_{t-1}, a_{t-1})$. When $a_{t-1} = A_S$, the belief state can be updated using the previous belief state and current observation based on Bayes' rule:

$$\begin{aligned} T(b_t \mid b_{t-1}, A_S) &= \Pr\left[\alpha', \beta' | \alpha, \beta, A_S\right] \\ &= \Pr\left[o_t | \alpha, \beta\right] \delta_{\alpha' = \alpha + o_t} \delta_{\beta' = \beta + n - o_t} \end{aligned} \tag{3.11}$$

for all $o_t \in \{0, \ldots, n\}$. In the above equation, $\delta(.)$ is the Kronecker delta, and $\Pr\left[o_t | \alpha, \beta\right]$ is the expected value of the likelihood function $\Pr\left[o_t | \mu\right] = \mu$ over the posterior distribution b_t:

$$\Pr\left[o_t | \alpha, \beta\right] = \binom{n}{o_t} \frac{\alpha^{o_t} \beta^{n - o_t}}{(\alpha + \beta)^n}, \tag{3.12}$$

which is a stationary distribution independent of time t. When the selected action is A_R or A_L, the animal stops sampling and makes an eye movement. To account for such cases, we include an additional state Γ, representing a terminal state, with zero reward $R(\Gamma, a) = 0$ and absorbing behavior, $T(\Gamma \mid \Gamma, a) = 1$ for all actions a. Formally, the transition probabilities with respect to the absorbing (termination) state are defined as $\Pr\left[\Gamma \mid b_t, a \in \{A_R, A_L\}\right] = 1$ for all b_t, indicating the end of a trial.

Given the time-independent belief state transition $\Pr\left[b'_t | b_t, a\right]$, the optimal value $v^\star$ and policy $\pi^\star = \arg\max_\pi v^\pi$ can be obtained by solving Bellman's equation:

$$\begin{aligned} \pi^\star(b_t) &= \operatorname*{argmax}_a \left[R(b_t, a) + \sum_{b'_t} \Pr\left[b'_t | b_t, a\right] v^*(b'_t) \right] \\ v^\star(b_t) &= \max_a \left[R(b_t, a) + \sum_{b'_t} \Pr\left[b'_t | b_t, a\right] v^*(b'_t) \right] \end{aligned} \tag{3.13}$$

Before we proceed to results from the model, we note that the one-step belief transition probability matrix $T(b_t | b_{t-1}, A_S)$ with $n = n_0$ can be shown be mathematically equivalent to the n_0-steps

transition matrix $T^{n_0}(b_t|b_{t-1}, A_S)$ with $n = 1$. The solution to Bellman's equation 3.13 is independent of n. Therefore, unless otherwise mentioned, the results are based on the most general scenario where the animal needs to select an action whenever a new spike is received, $i.e.$, $n = 1$.

We summarize the model variables as well as their statistical relationships in table 3.13.2 .

3 Results

3.1 Optimal Value Function and Policy

Figure 3.2 (a) shows the optimal value function computed by applying value iteration [149] to the POMDP defined in the Methods and Analysis section, with parameters $R_P = 50$, $R_N = 0$, and $R_S = -0.1$. The x-axis of Figure 3.2 (a) represents the total number of observations $m = m_R + m_L$ encountered thus far, which is equal to the elapsed time t in the trial. The y-axis represents the ratio $\hat{\mu} = \frac{m_R+\alpha_0}{m+\alpha_0+\beta_0}$, which is the estimator of the hidden parameter μ. In general, the model predicts a high value when $\hat{\mu}$ is close to 1 or 0, or equivalently, when the estimated coherence is close to 1. This is because at these two extremes, selecting the appropriate action has a high probability of receiving a large positive reward R_P. On the other hand, for $\hat{\mu}$ near 0.5 (estimated c near 0), choosing A_L or A_R in these states has a high chance of resulting in an incorrect decision and a large negative reward R_N (see [125] for a similar result using a different model and under the assumption of a deadline). Thus, belief states with $m_R \sim m_L$ have a much lower value compared to belief states with $m_R \gg m_L$ or $m_R \ll m_L$.

Figure 3.2 (b) shows the corresponding optimal policy $\pi^\star$ as a joint function of $\hat{\mu}$ and t. The optimal policy $\pi^\star$ partitions the belief space into three regions: Π^R, Π^L, and Π^S, representing the set of belief states preferring actions A_R, A_L and A_S respectively. Let Π^a_m be the set of belief states preferring action a after m observations, for $a \in \{A_R, A_L, A_S\}$ and $m = m_R + m_L$. Early in a trial, when m is small, the model selects the sampling action A_S regardless of the value of $\hat{\mu}$. This is because for small m, the variance of the point estimator $\hat{\mu}(m)$ is high. For example, even when $\hat{\mu} = 1$ when $m = 2$, the probability that the true $\mu < 0.5$ is still high. The sampling action A_S is required to reduce this variance by accruing more evidence. As m becomes larger, the variance

POMDP Variables	Descriptions
μ	The hidden variable of POMDP, $\mu = \frac{c \times d + 1}{2} \in [0, 1]$. In the random dots task, μ is a constant over time
c	The coherence (motion strength) of the random dots task. $c \in [0, 1]$. c is fixed during a task.
d	The underlying direction of the random dots task. $d \in \{\pm 1\}$. d is fixed during a task.
$\lambda^{\mathrm{MT}}_{R,L}$	The average spike rate of MT neurons preferring rightward or leftward direction, respectively, as a function of both coherence c and d described in equations 3.1.
$r^{\mathrm{MT}}_{R,L}$	The number of spikes emitted by MT neurons preferring rightward or leftward direction, respectively during one POMDP step. r^{MT} follows a Poisson distribution with mean λ^{MT}
n_t	Total number of spikes emitted by MT neurons during one POMDP step. $n_t = r^{\mathrm{MT}}_R + r^{\mathrm{MT}}_L$
o_t	The noisy observation at time step t, which is a conditional random variable $o_t = r^{\mathrm{MT}}_R \vert n_t$ following a Binomial distribution $\mathrm{Bino}(n_t, \mu)$. Note that $o_1, \ldots, o_t$ are conditional dependent of each other given the hidden variable μ
b_t	The belief (posterior distribution) $b_t = P(\mu \vert o_{1:t})$. With a beta-distributed initial belief $b_0 = \mathrm{Beta}(\alpha_0, \beta_0)$, b_t is also beta distributed due to the binomial distributed emission probability $P(o_t \vert \mu)$. Without loss of generality, $\alpha_0 = \beta_0 = 1$ throughout the paper.
a_t	Action chosen by the animal at time t. $a_t \in \{A_S, A_R, A_L\}$.

Table 3.1. Summary of model variables

Model Parameters	Description
R_S	A negative reward associated with the cost of an observation.
R_P	A positive reward associated with a correct eye movement.
R_N	A negative reward associated with an incorrect eye movement.
RT_{step}	The duration of a single observation, the real elapsed time per POMDP step. Only used to translate the number of POMDP time steps to real elapsed time when comparing with experimental data.
RT_0	Non-decision residual time. Both RT_{step} and RT_0 are obtained from a linear regression to compare model predictions (in unit of POMDP steps) with animals' response time (in unit of seconds), independent of the POMDP model.

Table 3.2. Summary of model paramters

of $\hat{\mu}$ decreases, and the deviation between $\hat{\mu}$ and the true value of μ diminishes by the law of large numbers. Consequently, the animal will pick action A_R even when $\hat{\mu}$ is only slightly above 0.5. This gradual decrease in the threshold over time for choosing the overt actions A_R or A_L has been called a "collapsing bound" in the decision making literature [92, 59, 33].

The optimal policy $\pi^\star$ is entirely determined by three reward parameters $\{R_P, R_N, R_S\}$. At a given belief state, $\pi^\star$ picks one of the three available actions that leads to the largest expected future reward. Thus, the choice is determined by the relative, not the absolute, value of the expected future reward for the different actions. From equation 4.7, we have

$$R(\alpha, \beta, A_L) - R(\alpha, \beta, A_R) \propto R_N - R_P. \tag{3.14}$$

If we regard the sampling penalty R_S as specifying the unit of reward, the optimal policy $\pi^\star$ is determined by the ratio $\frac{R_N - R_P}{R_S}$ alone. Figure 3.2 (c) shows the relationship between $\frac{R_N - R_P}{R_S}$ and the optimal policy $\pi^\star$ by showing the rightward decision boundaries $\phi^R(t)$ for different values of $\frac{R_N - R_P}{R_S}$. As $\frac{R_N - R_P}{R_S}$ increases (e.g., by making the sampling cost R_s smaller), the boundary $\phi^R(t)$ gradually moves towards the upper right corner, giving the animal more time to make decisions which results in more accurate decisions. To better understand this relationship, we fit the decision

boundary to a hyperbolic function:

$$\phi^R(t) - 0.5 \propto \frac{t}{t + \tau_{1/2}} \tag{3.15}$$

We find that $\tau_{1/2}$ exhibits nearly logarithmic growth with $\frac{R_N - R_P}{R_S}$. Interestingly, a collapsing bound is obtained even with extremely small R_S because the goal is reward maximization across trials: it is better to terminate a trial and accrue reward in future trials than to continue sampling noisy (possibly 0% coherent) stimuli.

3.2 Model Predictions: Psychometric Function and Reaction Time

We compare predictions of the model based on the learned policy π^* with experimental data from the reaction time version (rather than the fixed duration version) of the motion discrimination task [134]. As illustrated in Figure 3.3, the model assumes that motion information regarding the random dots on the screen is processed by MT neurons. These neurons provide the observations o_t (and $n - o_t$) to right- and left-direction coding LIP neurons, which maintain the belief state $b_t = \{\alpha = \sum_t o_t, \beta = \sum_t (n - o_t)\}$. Actions are selected based on the optimal policy π^*. If $b_t \in \Pi_t^R$ or $b_t \in \Pi_t^L$, the animal makes a rightward or leftward decision respectively and terminates the trial. When $b_t \in \Pi_t^S$, the animal chooses the sampling action and gets a new observation o_{t+1}.

The performance on the task using the optimal policy π^* can be measured in terms of both the accuracy of direction discrimination (the so-called psychometric function), and the reaction time required to reach a decision (the chronometric function). In this section, we derive the expected accuracy and reaction time as a function of stimulus coherence c, and compare them to the psychometric and chronometric functions of a monkey performing the same task [134].

The sequence of random variables $\{\hat{\mu}_1, \hat{\mu}_2, \ldots, \hat{\mu}_t\}$ forms a (non-stationary) Markov chain with transition probabilities determined by equation 4.2. Let $\Psi(\hat{\mu}_t, t|\mu)$ be the joint probability that the animal keeps selecting A_S until time step t:

$$\Psi(\hat{\mu}_t, t|\mu = \Pr\left[\hat{\mu}_1 \in \Pi_1^S, \hat{\mu}_2 \in \Pi_2^S, \ldots, \hat{\mu}_t \in \Pi_t^S\right]. \tag{3.16}$$

At $t = 0$, the animal will select A_S regardless of $\hat{\mu}$ under π^*, making $\psi(\hat{\mu}, 0|\mu) = \Pr[\hat{\mu}_0]$. At $t \geq 1$, $\Psi(\hat{\mu}_t, t|\mu)$ can be expressed recursively as:

$$\Psi(\hat{\mu}_t, t|\mu) = \sum_{\hat{\mu}_{t-1} \in \Pi_{t-1}^S} \Pr[\hat{\mu}_t|\hat{\mu}_{t-1}] \Psi(\hat{\mu}_{t-1}, t-1|\mu) \tag{3.17}$$

Let $\Pr[t, R|\mu]$ and $\Pr[t, L|\mu]$ be the joint probability mass functions that the animal makes a right or left choice at time t, respectively. These correspond to the probability that the point estimator $\hat{\mu}(t)$ crosses the boundary of Π^R or Π^L for the first time at time t:

$$\begin{aligned}
\Pr[t, R|\mu] &= \Pr\left[\hat{\mu}_t \in \Pi_t^R, \hat{\mu}_{t-1} \in \Pi_{t-1}^S, \ldots, \hat{\mu}_1 \in \Pi_1^S | \mu\right] \\
&= \sum_{\hat{\mu}_t \in \Pi_t^R} \sum_{\hat{\mu}_{t-1} \in \Pi_{t-1}^S} \Pr[\hat{\mu}_t|\hat{\mu}_{t-1}] \Psi(\hat{\mu}_{t-1}, t|\mu) \qquad (3.18) \\
\Pr[t, L|\mu] &= \sum_{\hat{\mu}_t \in \Pi_t^L} \sum_{\hat{\mu}_{t-1} \in \Pi_{t-1}^S} \Pr[\hat{\mu}_t|\hat{\mu}_{t-1}] \Psi(\hat{\mu}_{t-1}, t|\mu) \qquad (3.19)
\end{aligned}$$

The probabilities of making rightward or leftward eye movement are the marginal probabilities summing over all possible crossing times: $\Pr[R|\mu] = \sum_{t=1}^{\infty} \Pr[t, R|\mu]$ and $\Pr[L|\mu] = \sum_{t=1}^{\infty} \Pr[t, L|\mu]$. When the underlying motion direction is rightward, $\Pr[R|\mu]$ represents the accuracy of motion discrimination and $\Pr[L|\mu]$ represents the error rate. The mean reaction times for correct and error choices are the expected crossing times over the conditional probability that the animal makes decision A_R and A_L respectively at time t:

$$RT_R(\mu) = \sum_{t=1}^{\infty} t \frac{\Pr[t, R|\mu]}{\Pr[R|\mu]} \tag{3.20}$$

$$RT_L(\mu) = \sum_{t=1}^{\infty} t \frac{\Pr[t, L|\mu]}{\Pr[L|\mu]} \tag{3.21}$$

The left panel of Figure 3.4 shows performance accuracy as a function of motion strength c for the model (solid curve) and a monkey (black dots). The model parameters are the same as those in Figure 3.2, obtained using a binary search within $R_p \in \{0, 2000\}$ with a minimum step size 10.

The right panel of Figure 3.4 shows for the same model parameters the predicted mean reacton time $RT_R(\mu)$ for correct choices as a function of coherence c (and fixed direction $d = 1$) for the

model (solid curve) and the monkey (black dots). Note that $RT_R(\mu)$ represents the expected number of POMDP time steps for making a rightward eye movement A_R. It follows from the Poisson spiking process that the duration of each POMDP time step follows a exponential distribution with its expectation proportional to $\lambda_R(\mu) + \lambda_L(\mu)$. In order to make a direct comparison to the monkey data $RT^*_R(\mu)$, which is in units of real time, a linear regression was used to to determine the duration RT_{step} of a single observation and the onset of decision time RT_0:

$$RT^*_R(\mu) = RT_{\text{step}} * (\lambda_R(\mu) + \lambda_L(\mu)) * RT_R(\mu) + RT_0. \tag{3.22}$$

Note that the reaction time in a trial is the sum of decision time plus the non-decision delays whose properties are not well understood. The offset RT_0 represents the non-decision residual time. We applied the experimental mean reaction time reported in [134] with motion coherence $c = \{0.032, 0.064, 0.128, 0.256, 0.512\}$ to compute the two coefficients RT_{step} and RT_0. The unit duration per POMDP step $RT_{\text{step}} = 9.20$ms/step, and the offset $RT_0 = 358.5$ms, which is comparable to the 300ms non-decision time on average reported in the literature [96, 102].

There is essentially one parameter in our model needed to fit the experimental accuracy data, namely, the reward ratio $\frac{R_N - R_P}{R_S}$. The other two parameters RT_{step} and RT_0 are independent of the POMDP model, and are used only to translate the POMDP time steps into real elapsed time. This reward ratio has direct physical interpretation and can be easily manipulated by the experimenters. For example, changing the amount of awards for the correct/incorrect choices, or giving subjects different speed instructions will effectively change $\frac{R_N - R_P}{R_S}$. In Figure 3.5 (a), we show performance accuracies $\Pr[R|\mu]$ and predicted mean reaction time $RT_R(\mu)$ with different values of $\frac{R_N - R_P}{R_S}$. With fixed R_N and R_P, decreasing R_S makes the observations more affordable and allows subjects to accumulate more evidence, in turn leads to a longer decision time and higher accuracy. Our model thus provides a quantitative framework for predicting the effects of reward parameters on the accuracy and speed of decision making. To test our theory, we compare the model predictions with the experimental data from a human subject, reported by Hanks et al[74], under different speed-accuracy regimes. In their experiments, human subjects were instructed to perform the random dots task under different speed-accuracy conditions. The red crosses in Figure 3.5 (b) represent the

response time and accuracy of a human subject in the direction discrimination task with instructions to perform the task more carefully at a slower speed, while the black dots represent the task under normal speed conditions. The slower speed instruction encourages human subjects to accumulate more observations before making the final decision. In the model, this amounts to reducing the negative cost associated with each sample R_s. Indeed, this tradeoff between speed and accuracy was consistent with predicted effects of changing the reward ratio. We first fit the model parameters to experimental data under normal speed conditions, based on fitting $\frac{R_N - R_P}{R_S}$, $RT_{\text{step}} = 7.7$ ms/step, and $RT_0 = 204$ms (Figure 3.5 (b), black solid curves). The red dashed lines shown in Figure 3.5 (b) are model fits to the data under slower speed instruction. There is just one degree of freedom in this fit, as all model parameters except the reward ratio were fixed to the values used to fit data in the normal speed regime.

3.3 Neural response during direction discrimination task

From Figure 3.2 (b), it is clear that for the random dots task, the animal does not need to store the whole two dimensional optimal policy but only the two one-dimensional decision boundaries ϕ^R and ϕ^L. This naturally suggests a neural mechanism for decision making similar to that in drift diffusion models: LIP neurons compute the belief state from MT responses and employ divisive normalization to maintain the point estimate $\hat{\mu}_t = \frac{m_R + \alpha_0}{m + \alpha_0 + \beta_0}$. We now explore the hypothesis that the response of LIP neurons represents the difference between $\hat{\mu}$ and the optimal decision threshold $\phi^R(t)$. In this model, a rightward eye movement is initiated only when the difference $\frac{m_R}{m_R + m_L} - \phi^R$ reaches a fixed bound (in this case, 0). Therefore, we modeled the firing rates in the lateral intraparietal area (LIP) λ^{LIP} as:

$$\lambda_R^{LIP}(t) = \lambda_0^{LIP} + \hat{B}\left(\frac{m_R + \alpha_0}{m + \alpha_0 + \beta_0} - \phi^R(t) + \frac{\beta_0}{\alpha_0 + \beta_0}\right) \tag{3.23}$$

where λ_0^{LIP} is the spontaneous firing rate for LIP neurons. Since $\phi^R(0) = 1$, a constant $\frac{\beta_0}{\alpha_0 + \beta_0}$ is added to make $\lambda_R^{LIP}(0) = \lambda_0^{LIP}$. $\hat{B}$ represents the termination bound; $\hat{B} = 49.6$ spikes s^{-1} from [33]. The firing rate λ_L^{LIP} is defined similarly.

The above model makes two testable predictions about neural responses in LIP. The first is that

the neural response to 0% coherent motion (the so called "urgency" signal [33, 34]) encodes the decision boundary $\phi^R(t)$ (or $\phi^L(t)$ for leftward-preferring LIP neurons). In Figure 3.6a, we plot the model response to 0% coherent motion, along with a fit to a hyperbolic function $u(t) \propto \frac{t}{t+\tau_{1/2}}$, the same function that Churchland et al [33] used to parametrize the experimentally observed "urgency signal." The parameter $\tau_{1/2}$ is the time taken to reach 50% of the maximum. The estimate of $\tau_{1/2}$ for the model from Figure 3.6 (a) is 123ms, which is consistent with the $\tau_{1/2} = 133.2$ms estimated from neural data [33].

The second prediction concerns the buildup rate (in units of spikes s^{-2} coh^{-1}) of the LIP firing rates. The buildup rate of LIP at each motion strength is calculated from the slope of a line fit to model LIP firing rate during the first 120ms of decision time. As shown in Figure 3.6 (b), buildup rates scaled approximately linearly as a function of motion coherence. The effect of a unit change in coherence on the buildup rate can be estimated from the slope of the fitted line to be 227.7 spike s^{-2} coh^{-1}, similar to what has been reported in the literature [33] (222.5 spike s^{-2} coh^{-1}).

4 Discussion

The random dots motion discrimination task has provided a wealth of information regarding decision making in the primate brain. Much of this data has previously been modeled using the drift diffusion model [115, 19], but to fully account for the experimental data, one has to sometimes use ad-hoc assumptions. This paper introduces an alternative model for explaining the monkey's behavior based on the framework of partially observable Markov decision processes (POMDPs).

We believe that the POMDP model provides a more versatile framework for decision making compared to the drift diffusion model, which can be viewed as a special case of sequential statistical hypothesis testing (SSHT) [91]. Sequential statistical hypothesis testing assumes that the stimuli (observations) are independent and identically distributed whereas the POMDP model allows observations be temporally correlated. The observations in the POMDP are conditionally independent given the hidden state μ, which evolves according to a Markov chain. Thus, the POMDP framework for decision making [60, 163, 145, 125, 80] can be regarded as a strictly more general model than the SSHT models. We intend to explore the applicability of our POMDP model

to time-dependent stimuli, such as temporally dynamic attention [66] and temporally blurred stimulus representations [97] in future studies.

Another advantage of a POMDP model is that the model parameters have direct physical interpretations and can be easily manipulated by the experimenter. Our analysis shows that the optimal policy is fully determined by the reward parameters $\{R_P, R_N, R_S\}$. Thus, the model psychometric and chronometric functions, which are derived from the optimal policy, are also fully determined by these model parameters. Experimenters can control these reward parameters by changing the amount of awards for the correct/incorrect choices, or by giving subjects different speed instructions. This allows our model to make testable predictions, as demonstrated by the effects of the change in the reward ratios on the speed-accuracy trade-off. It should be noted that these reward parameters can be subjective and may vary from individual to individual. For example, R_P can be directly related to the external food or juice reward provided by the experimenter while R_S may be linked to internal factors such as degree of hunger or thirst, drive, and motivation. The precise relationship between these reward parameters and the external reward/risk controlled by the experimenter remains unknown. Our model thus provides a quantitative framework for studying this relationship between internal reward mechanisms and external physical reward.

The proposed model demonstrates how the monkey's choices in the random dots task can be interpreted as being optimal under the hypothesis of reward maximization. The reward maximization hypothesis has previously been used to explain behavioral data from conditioning experiments [37] and dopaminergic responses under the framework of temporal difference (TD) learning [140]. Our model extends these results to the more general problem of decision making under uncertainty. The model predicts psychometric and chronometric functions that are quantitatively close to those observed in monkeys and humans solving the random dots task.

We showed through analytical derivations and numerical simulation that the optimal threshold for selecting overt actions is a declining function of time. Such a collapsing decision bound has previously been obtained for decision making under a deadline [59, 125]. It has also been proposed as an ad-hoc mechanism in drift diffusion models [49, 92, 33] for explaining finite response time at zero percent coherence. Our results demonstrate that a collapsing bound emerges naturally as a

consequence of reward maximization. Additionally, the POMDP model readily generalizes to the case of decision making with arbitrary numbers of states and actions, as well as time-varying state.

Instead of traditional dynamic programming techniques, the optimal policy π^* and value v^* can be learned via Monte Carlo approximation-based methods such as temporal difference (TD) learning [149]. There is much evidence suggesting that the firing rate of midbrain dopaminergic neurons might represent the reward prediction error in TD learning. Thus, the learning of value and policy in the current model could potentially be implemented in a manner similar to previous TD learning models of the basal ganglia [140, 37, 125, 20].

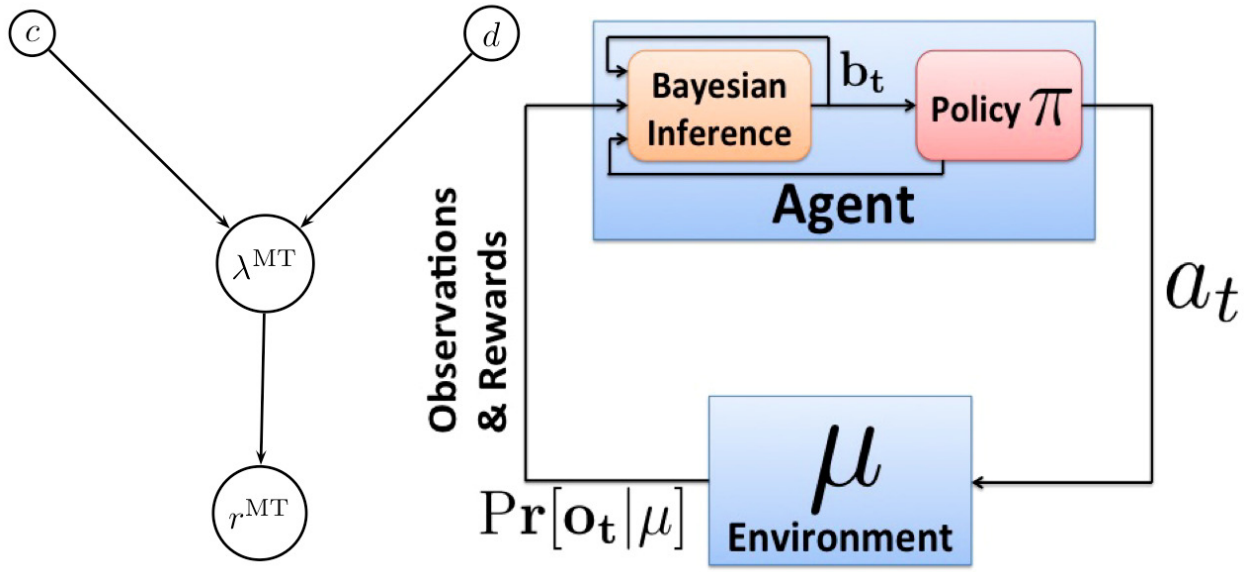
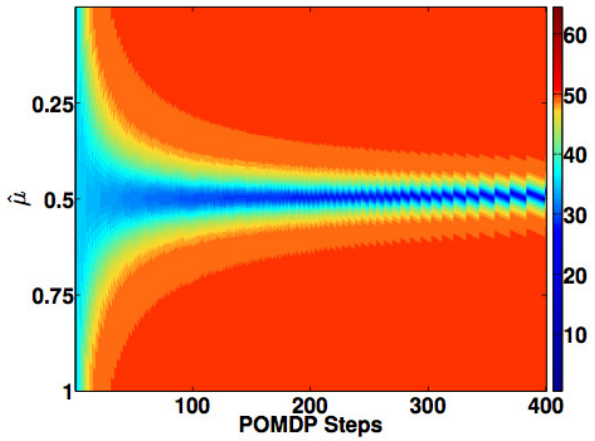
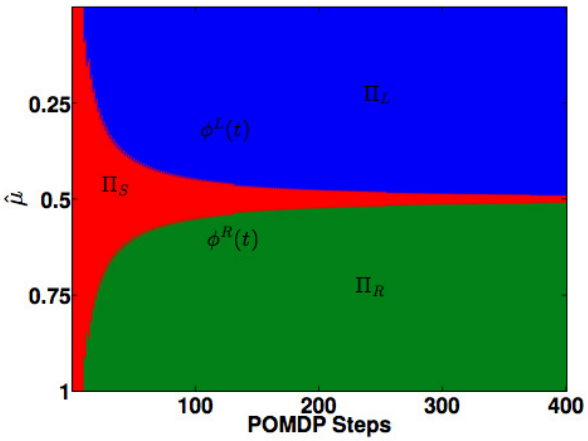


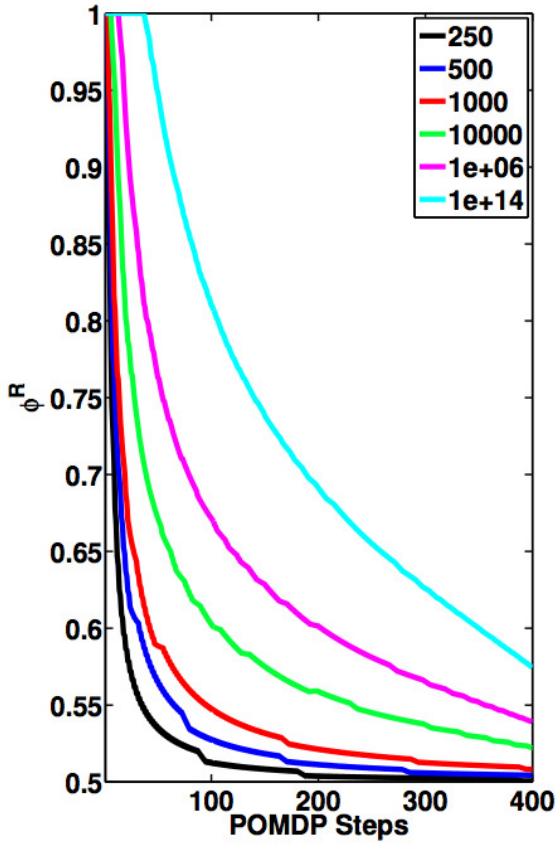
Figure 3.1. POMP Framework for Decision Making. *Left:* The graphical model representing the probabilistic relationship between random variables c, d, λ and r. In the POMDP model, the hidden state μ corresponds to coherence c and direction d jointly. The observation o_t corresponds to MT response $r^{\mathrm{MT}}(t)$. The relations between these variables are summarized in table 3.13.2. *Right:* In order to solve a POMDP problem, the animal maintains a belief b_t, which is a posterior probability distribution over hidden states $\mu =$ of the world given observations $o_{1:t}$. At a current belief state b_t, an action is selected according to the learned policy π, which maps belief states to actions.

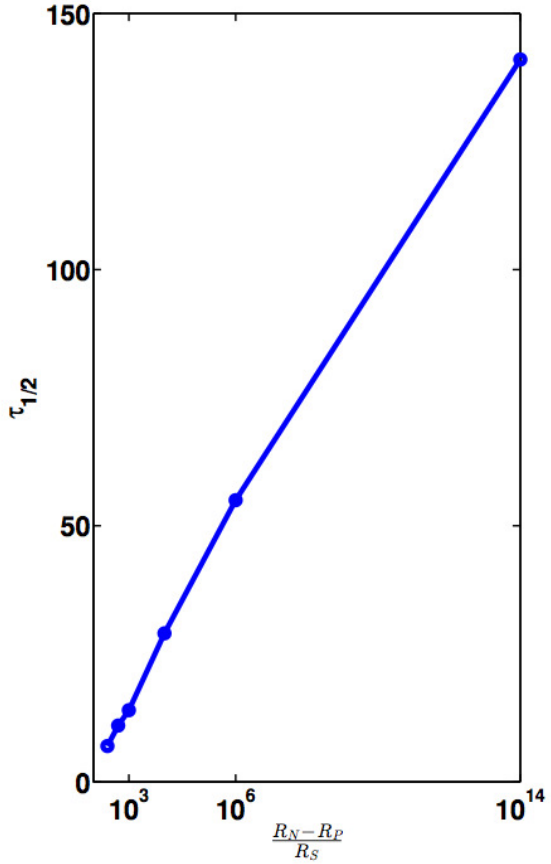

0.25
0.5
0.75
1
$\hat{\mu}$
100
200
300
400
POMDP Steps
60
50
40
30
20
10

(a)


Π_L
$\phi^L(t)$
Π_S
$\phi^R(t)$
Π_R
0.25
0.5
0.75
1
$\hat{\mu}$
100
200
300
400
POMDP Steps

(b)


250
500
1000
10000
1e+06
1e+14
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0.5
ϕ^R
100
200
300
400
POMDP Steps


150
100
50
0
$\tau_{1/2}$
10^3
10^6
10^{14}
$\frac{R_N - R_P}{R_S}$

(c)

Figure 3.2 ***(preceding page).*** **Optimal Value and Policy for the Random Dots Task**. (a) Optimal value as a joint function of $\hat{\mu} = \frac{m_R+\alpha_0}{m+\alpha_0+\beta_0}$ and the number of POMDP steps t. (b) Optimal Policy as a function of $\hat{\mu}$ and the number of POMDP steps t. The boundaries $\phi^R(t)$ and $\phi^L(t)$ divide the belief space into three areas: Π_S (red), Π_R (green), and Π_L (blue), each of which represents belief states whose optimal actions are A_S, A_R and A_L respectively. Model parameters: $R_P = 50$, $R_S = -0.1$, and $R_N = 0$. (c) ***Left:*** The rightward decision boundary $\phi^R(t)$ for different values of $\frac{R_N-R_P}{R_S}$. ***Right:*** The half time $\tau_{1/2}$ of $\phi^R(t)$ for different values of $\frac{R_N-R_P}{R_S}$, where $\phi^R(\tau_{1/2}) = \frac{\phi^R(0)-\phi^R(\infty)}{2}$.

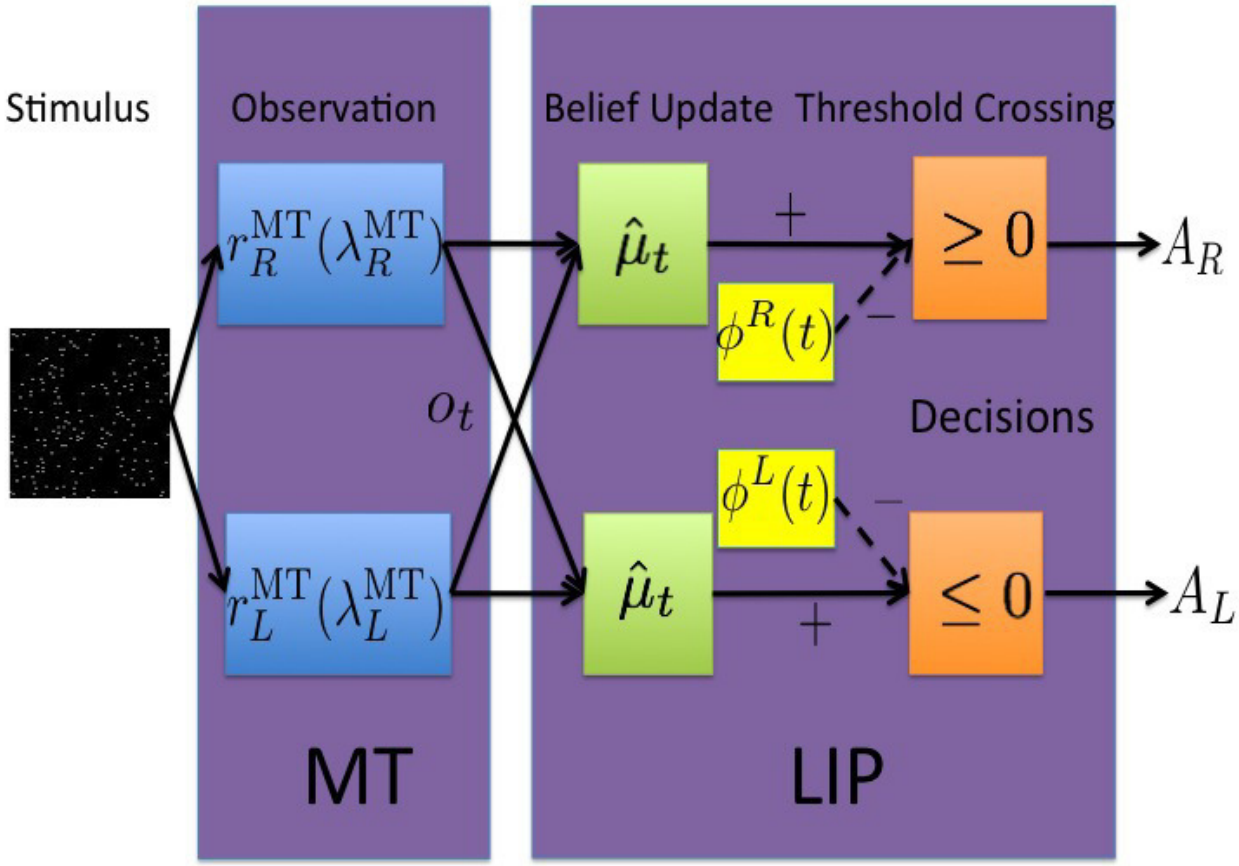


Figure 3.3. Relationship between Model and Neural Activity. The input to the model is a random dots motion sequence. Neurons in MT with tuning curves λ^{MT} emit r^{MT} spikes at time step t, which constitutes the observation o_t in the POMDP model. The animal maintains the belief state b_t by computing $\hat{\mu}_t$ (b_t can be parameterized by $\hat{\mu}_t$ and t - see text). The optimal policy is implemented by selecting rightward eye movement A_R when $\hat{\mu}_t \geq \phi^R(t)$, or equivalently, when $(\hat{\mu}_t - \phi^R(t)) \geq 0$ (and likewise for leftward eye movement A_L).

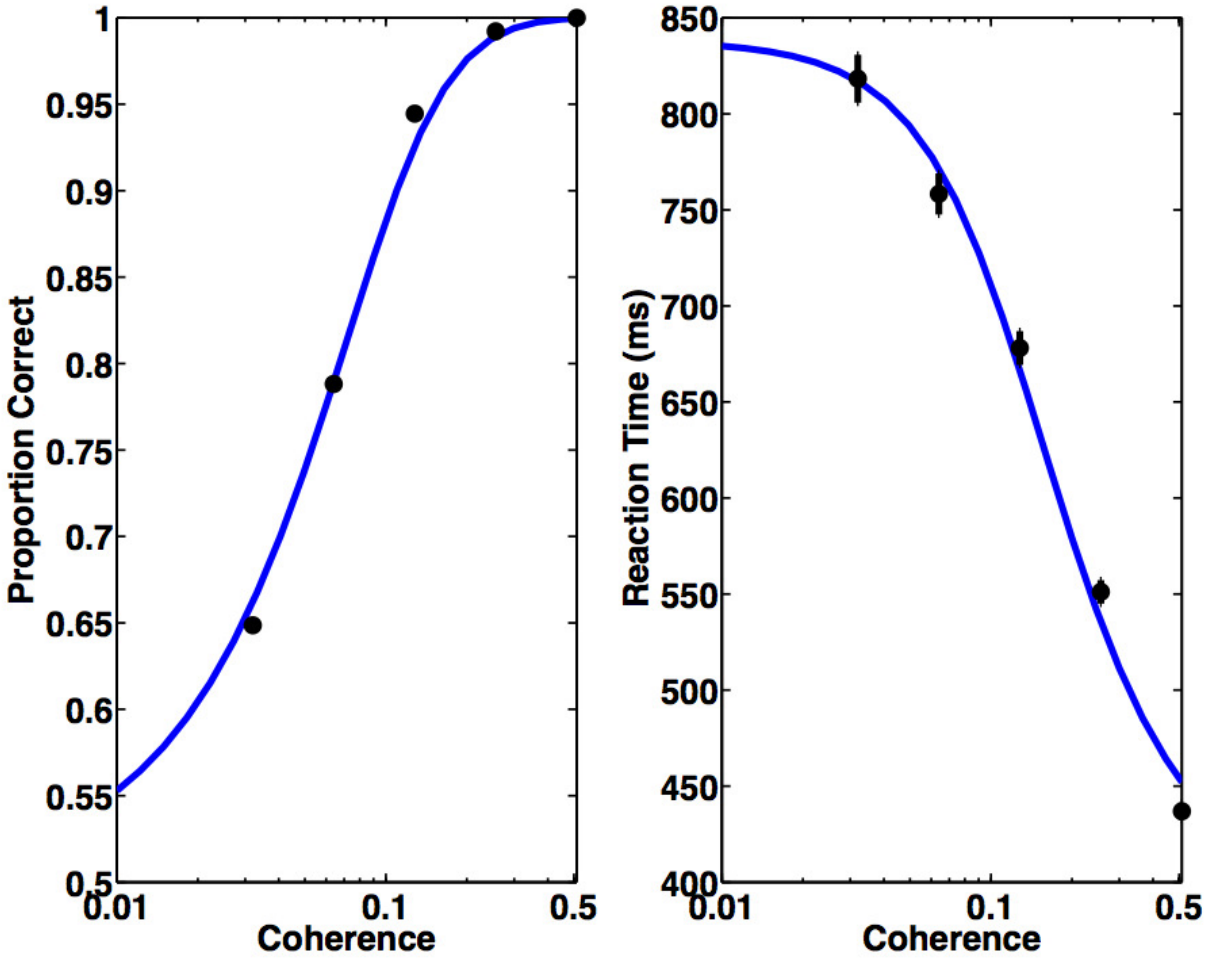
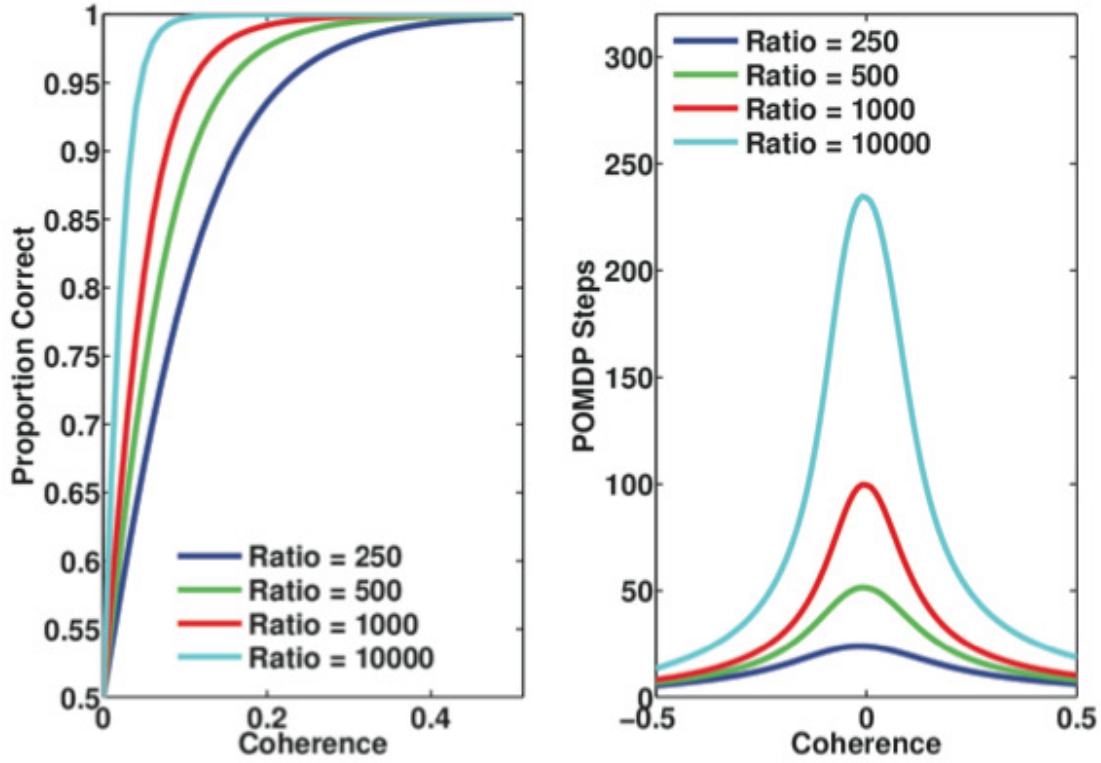
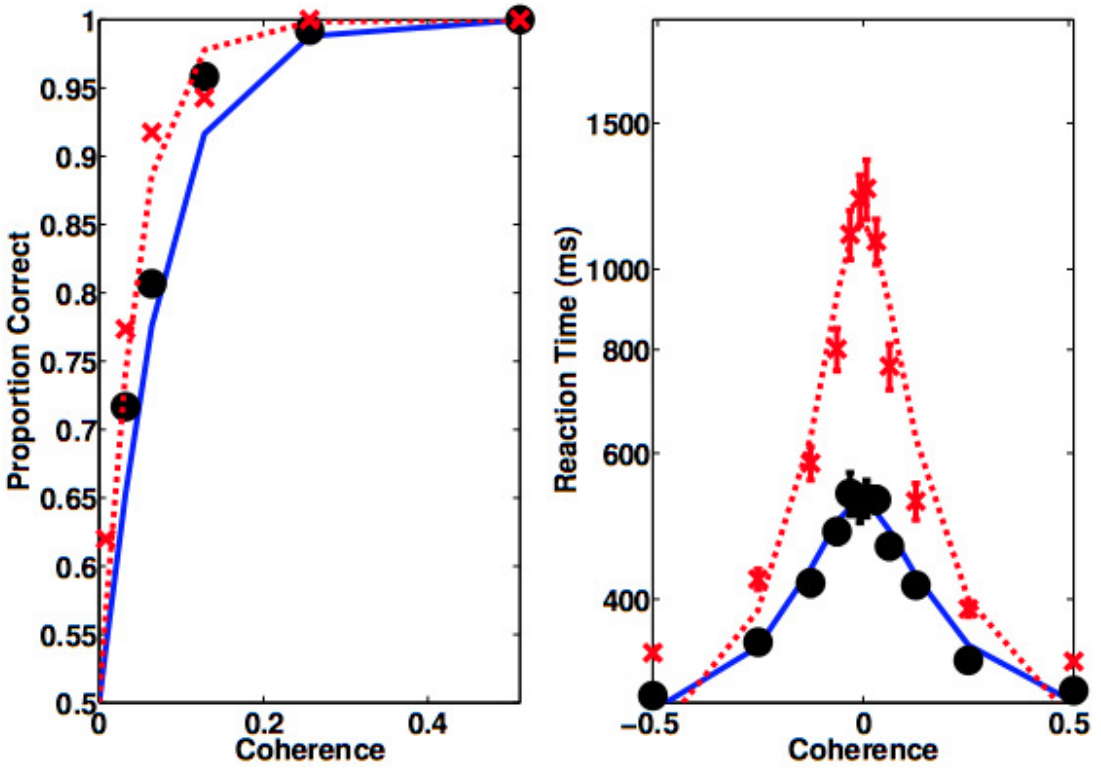


Figure 3.4. Comparison of Performance of the Model and Monkey. Black dots with error bars represent a monkey's decision accuracy and reaction time for correct trials. Blue solid curves are model predictions ($RT_R(\mu)$ and $RT_R(\mu)$ in the text) for parameter values $R_P = 50, R_S = -0.1$, and $R_N = 0$. Monkey data from [134].


Proportion Correct
Coherence
Ratio = 250
Ratio = 500
Ratio = 1000
Ratio = 10000
POMDP Steps
Coherence
Ratio = 250
Ratio = 500
Ratio = 1000
Ratio = 10000

(a)


Proportion Correct
Coherence
Reaction Time (ms)
Coherence

(b)

Figure 3.5 *(preceding page).* Effect of $\frac{R_N - R_P}{R_S}$ on speed-accuracy tradeoff. (a) Model predictions of psychometric and chronometric functions for different values of $\frac{R_N - R_P}{R_S}$. (b) Comparison of model predictions and experimental data for different speed-accuracy regimes. The black dots represent the response time and accuracy of a human subject in the direction discrimination task under normal speed conditions, while the red crosses represent data with a slower speed instruction. The model predictions are plotted as black solid curves (with $\frac{R_N - R_P}{R_S} = 450$) and red dashed lines ($\frac{R_N - R_P}{R_S} = 1250$), respectively. The per-step duration and non-decision residual time are fixed to be the same for both conditions: $RT_{\text{step}} = 7.7$ ms/step, and $RT_0 = 204$ms. Human data are from human subject LH in [74].

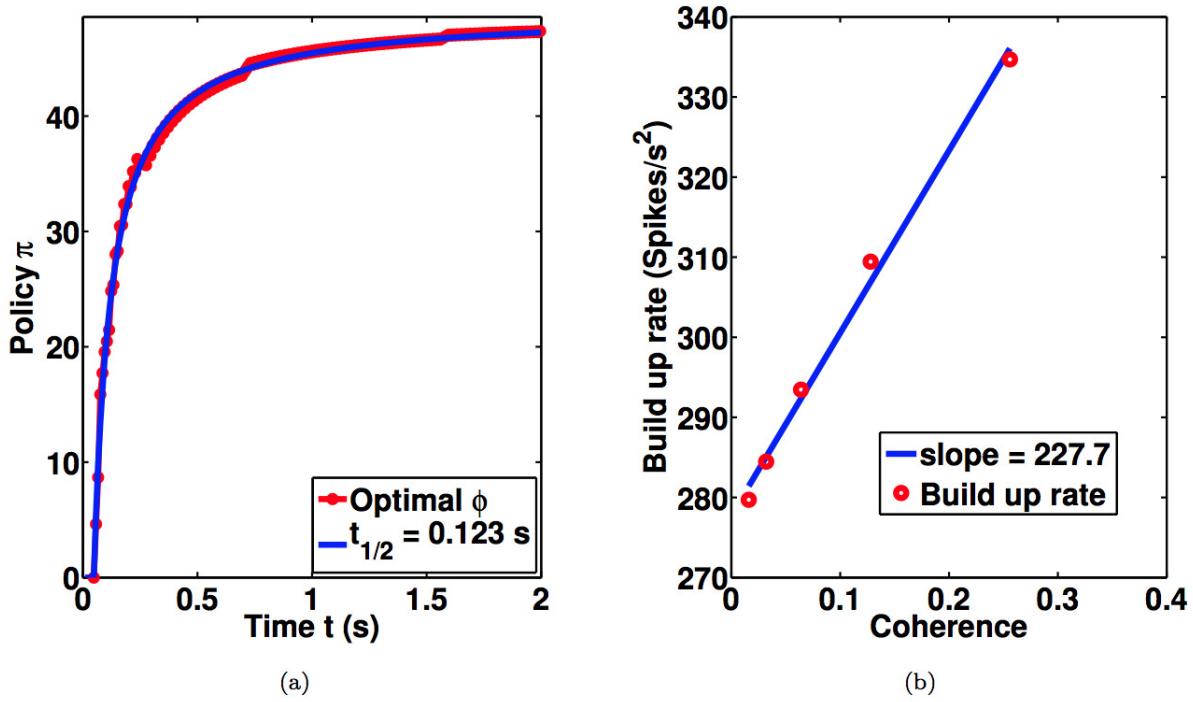


Figure 3.6. Comparison of Model and Neural Responses. (a) Model response to 0% coherence motion is shown in red. Blue curve depicts a fit using a hyperbolic function $u(t) = u_\infty \frac{t}{t+\tau_{1/2}}$ where $\tau_{1/2} = 123$ms, which is comparable to the value of 133.2ms estimated from neural data [33]. (b) The first 120ms of decision time was used to compute the buildup rate from the model response following the procedure in [33]. The red points show model buildup rates estimated for each coherence value. The effect of a unit change in the coherence on buildup rate can be estimated from the slope of the blue fitted line: this value, 227.7 spike s^{-2} coh^{-1}, is similar to the corresponding value 222.5 spike s^{-2} coh^{-1} estimated from neural data [33].

Chapter 4

OPTIMAL INTEGRATION OF PRIOR KNOWLEDGE IN SENSORY DECISION MAKING

Abstract

Decision making requires the integration of multiple sources of information: prior knowledge, noisy sensory stimuli, and rewards. Bayesian models have proved effective for understanding this process, but are yet unable to explain how the brain incorporates prior information in sequential decision making tasks where decisions have both immediate and long-term effects. Here, we present a model combining Bayesian inference with the principle of optimality. Decisions are chosen so as to maximize cumulative expected rewards over the course of sequential decision making. We show that such a model explains behavioral data from several sensory decision making tasks in humans and monkeys, while also providing a normative explanation for otherwise conflicting neurophysiological data from cortical area LIP.

1 Introduction

Our brains are extremely adept at making decisions that can achieve distant goals. Such decision making requires combining sensory information received along the way with prior knowledge acquired from past experience. For example, when finding our way to a new restaurant recommended by a friend, we effortlessly incorporate prior knowledge about roads in a neighborhood with landmarks and street signs we see as we drive. Similarly, a rat trained to find food at a distant location in a maze must combine prior knowledge about maze locations with visual, olfactory, and tactile information to select actions that lead to the delayed reward. What are the brain mechanisms guiding such action selection?

Bayesian models [87, 167, 126, 124, 98, 52] have suggested that perception and action rely

on computing a belief (posterior probability distribution) over task-relevant variables as prescribed by Bayes' rule. Whether such models apply to sequential decision making in tasks with delayed gratification remains an open question. Here we show that Bayesian inference combined with the principle of reward optimality can explain a variety of experimental results on human and monkey sequential decision making, including results that otherwise have seemed contradictory.

We begin with the same assumption as other Bayesian models [87, 126, 52], namely, that the brain handles uncertainty by maintaining a belief over task-relevant states. We additionally postulate that actions are chosen according to a "policy" which is a mapping from beliefs to actions [31, 84]. A reward or cost function assigns positive or negative values to states and actions according to the constraints of the task as well as internal drives such as hunger or thirst. Bellman's principle of optimality [10, 78, 138] prescribes that animals use policies that maximize the expected sum of future rewards. Similar models have been successfully used in modeling primates' perceptual decision making tasks [37, 163, 125, 80, 53, 81]. However, how prior knowledge influences sequential decision making has not been examined in these models.

We applied our model to a well-studied sequential decision making task (Fig. 1a). In this task, the subject (a monkey or human) observes a video stream of randomly moving dots, a fraction of which are moving in one of two possible directions (e.g., leftward or rightward). Subjects must infer this direction of motion and indicate their choice by making an eye movement to one of two visual targets, one for each direction (see Figure 1a; human subjects used button presses). The difficulty of the task is controlled by a parameter called coherence, defined as the probability that a dot is moving in the coherent direction rather than randomly displaced: the smaller the coherence, the harder the task. We postulate that subjects update their belief over the unknown direction of motion as new sensory evidence is observed (new images of dots), and pick one of three actions at each point in time: either choose one of the two possible directions, or choose to wait for more evidence. The subject gets a reward (e.g., a juice reward in the case of monkeys) when the correct choice is made and a penalty (e.g., a delay until the next trial) for an incorrect choice. Our model also assumes that each sample of evidence (at each time step) costs some small negative reward or penalty (denoting, for example, metabolic energy consumed).

The effects of prior knowledge or bias in this task can be examined by providing a partially predictive cue about motion direction before a trial, [130] by making the reward amounts for the two directions unequal, [135] or simply making one motion direction more frequent than the other. [74] In this context, two versions of the task have been considered: the fixed duration version wherein the motion stimulus is shown for a fixed duration of time before the subject can report the direction, [130, 135] and the reaction time version in which the subject can report the direction whenever they are confident enough about their choice. [74] We hypothesized that the subject learned and employed the optimal policy in these tasks and actions were selected to maximize the resulting expected cumulative future reward (see Supplementary Materials).

2 Results

Figure 1b shows the optimal policy predicted by the model for the unbiased case (both motion directions equally likely) as a function of two quantities: a "decision variable" ρ, representing the accumulated evidence for motion in a particular direction[68] and the elapsed time t since the start of the trial. By convention, $\rho > 0$ indicates evidence favoring rightward direction, $\rho < 0$ indicates leftward direction, and $\rho = 0$ indicates no preference. In our model, ρ and t together characterize the posterior probability (or belief) over the unknown direction and coherence of motion (see Supplementary Information). The decision variable ρ starts at 0 and is updated with each new observation in accordance with Bayes' rule. This results in noisy trajectories for the belief such as the one shown in Figure 1b, similar to the evolution of decision variables in classical drift diffusion models of decision making. [134, 102, 25]

The optimal policy in Figure 1b partitions the belief space into three regions (three colors), each representing the set of beliefs for which the optimal action is one of: "decide that the motion is rightward" (upper region), "sample one more time step" (middle region) or "decide that the motion is leftward" (lower region). As seen in Figure 1b, early in a trial, the model selects the "sample" action because of high uncertainty due to lack of sufficient evidence. Later in the trial, after enough observations have been received, the model chooses the "rightward" or "leftward" action, even when ρ is only slightly above or below 0, because the model's uncertainty has decreased sufficiently

and it is better to make a decision based on the accumulated evidence than incur the cost of waiting for more observations.

The gradual decrease in the decision threshold in Figure 1b, called a "collapsing bound" in the decision making literature, [92, 60, 33] is a prediction of our model arising from the principle of reward optimality. For experiments where the stimulus is shown for a fixed rather than an unbounded duration of time, the model predicts that the collapsing decision boundary will vary with the duration, collapsing to zero at the appropriate time as shown in Figure 1c. This prediction has consequences for the subject's accuracy in the task, as discussed below.

Our model predicts that the optimal policy is entirely determined by the prior probability of motion direction and the relative values of rewards and costs for correct, incorrect, and sampling actions (see Supplementary Information). Note that for the unbiased case (prior probability = 0.5 for each direction; Fig. 1b), the expected reward function is symmetric and therefore the decision boundaries are also symmetric around 0. When the prior probability is changed (e.g., prior probability for rightward direction = 0.9), the optimal policy becomes biased towards the choice with higher prior probability (Fig. 1d): the decision boundary for the "rightward" action shifts towards the center while the boundary for "leftward" shifts away from the center. As a result, the model prescribes more sampling actions for the "leftward" action compared to the "rightward" action. The same belief trajectory that picks the "leftward" action in Fig. 1b will now choose the "rightward" action at an earlier time for the biased policy in Fig. 1d.

We tested the predictions of the model for the motion discrimination task using both behavioral and neural data from experiments performed in three different laboratories. In the first experiment (Fig. 2a),[130] decision biases were introduced in the fixed-duration version of the task by assigning a high (H) or low (L) reward to the "leftward" or "rightward" choices. This results in a total of four reward conditions (HH, HL, LH, LL), where, for instance, HL represents the case where the reward for the correct choice is high if the true direction is leftward and low if rightward. The induced prior probability for a choice can be shown to be equivalent to the ratio of the reward for that choice divided by the sum of rewards for both choices (Supplementary Information). Based on parameters fit only to the HL condition, the model was able to accurately predict the accuracy of

direction discrimination (the psychometric function) for the other three reward conditions (Fig. 2b).

In the second experiment (Fig. 2c), [135] an arrow cue preceded the motion stimulus in a trial and was partially predictive of the upcoming random-dot motion direction: arrow cues pointing toward one choice for direction (labeled $preferred$; opposite direction labeled $null$) predicted that the motion was twice as likely to be in that direction. Neutral (bidirectional arrow) cues indicated that the upcoming motion was equally likely to be in either direction. Based on parameters fit only to the $preferred$ condition (arrow cue towards receptive field of recorded neuron, black), the model was able to predict the accuracies for both the $null$ (arrow cue towards opposite direction, green) and $neutral$ ("Neu", bidirectional arrow, blue) cues (Fig. 2d). The model additionally predicts that the longer the duration of the stimulus, the less the amount of noise in the accumulated evidence, and the more accurate the performance of the monkey (see Supplementary Information).

Figure 3 shows the comparison between the predictions of the model and behavioral data from humans and monkeys in the reaction time version of the motion discrimination task. [74, 134] In "neutral" trials, the two directions of motion were equally probable whereas in "biased" trials, the probability of rightward motion was 0.8. Human subjects were additionally told in biased trials that the probability of motion was 0.8. The accuracy and reaction time predicted by the model closely match human and monkey data (note the relatively high coefficients of determination). For each plot, predictions for the biased condition were made based on parameters fit to the neutral case (Supplementary Information). Note that the time-varying effect of bias seen in the experimental data[74] is inconsistent with the predictions of the standard drift diffusion model without assumptions of dynamic bias signals[108]. Our approach predicts this time-varying effect directly as a consequence of the principle of reward optimality, without any additional parameter fitting.

The optimal policy and belief trajectories in Figure 1b suggest a neural mechanism for decision making in the motion discrimination task. Neurons in cortical area LIP are known to be involved in accumulating evidence during this task based on neural responses in motion processing area MT [112, 142]. Similar to previous "race" models of decision making [33], we hypothesize that the response of LIP neurons represents the sum between the accumulated evidence for a direction of motion up to time t and the deviation of the time varying decision boundary from a fixed boundary

(Fig. 1c; Supplementary Information). Under this model, a decision, indicated by an eye movement to a target or button press, is made only when this difference reaches a fixed bound.

Figure 4a shows the predicted model LIP responses for the fixed duration random dots task under the three prior probability conditions from Figure 2d. Note the effect on model responses when prior probability of motion is increased to 0.8 ("Pref") or decreased to 0.2 ("Null") from 0.5 ("Neutral"). The model predicts a large additive offset in the initial (spontaneous) firing rate of LIP neurons, as has been observed in fixed-duration random dots tasks (Fig. 4b). [135, 130] Further, the model predicts a decreasing bias signal over time, where the bias signal is computed as a difference in decision boundaries for neutral and biased conditions (Fig. 4c).

In the reaction-time version of the task, the model predicts a much smaller difference in the initial LIP responses between the biased and neutral conditions (Fig. 4d), as seen in LIP data for this task (Fig. 4e). [74] However, the model predicts that this bias should increase as a function of elapsed time as shown in Figure 4f. Such an increasing "dynamic bias signal" is also seen in LIP neurons [74] and is a direct consequence of the optimal policies (Figs. 1b and 1d) predicted by the model for the different prior probability conditions.

Since the model hypothesizes that LIP responses encode the difference between accumulated evidence and decision boundaries, the response to 0% coherent motion should encode the decision boundary (equivalently, the "urgency signal" [33]). We therefore computed decision boundaries from LIP responses to 0% coherent motion and derived a "neural" policy for the subject. The resulting neurally-derived psychometric and chronometric functions again closely approximate the experimental accuracy and reaction time data (Figs. 4g and 4h), though, as expected, the R^2 values for the neural policy are slightly lower than those obtained for the optimal policy (Fig. 3c).

3 Discussion

These results suggest that decision making in the primate brain may be governed by the dual principles of Bayesian inference and reward optimality. Bayesian inference allows beliefs (posterior probability estimates) to be computed from prior knowledge and sensory evidence accumulated over time. These beliefs are in turn used to select actions that maximize expected cumulative

reward under uncertainty. Such a normative model differs from previous descriptive models of decision making [29, 134, 102] in that important attributes such as collapsing decision boundaries and dynamic weighting of prior information over time emerge naturally rather than as assumptions used to fit the data. Additionally, the approach can be readily generalized to allow time-varying hidden states [31, 84] and other complex dependencies between random variables (e.g., hierarchies), [103, 120, 72] opening the door to understanding more complicated decision making behaviors and their neural substrates.

Our model relies on the principle of reward optimization to explain how animals makes decisions. The same principle has been successfully applied to explain the phenomenon of collapsing decision boundaries in the random dots motion discrimination task [163, 53]. However, how an animals decision is influenced by changes in prior probability and the reward function has not been explained by previous models. Our model predicts that the optimal policy is a deterministic function of reward parameters and prior probability, which correspond to physical parameters that can be controlled in experiments. No extra free parameters are required. Moreover, the POMDP framework is general enough to model a large variety of real world decision making processes, including decision making tasks involving time dependent stimuli, in contrast to previous models that were restricted to i.i.d observations.

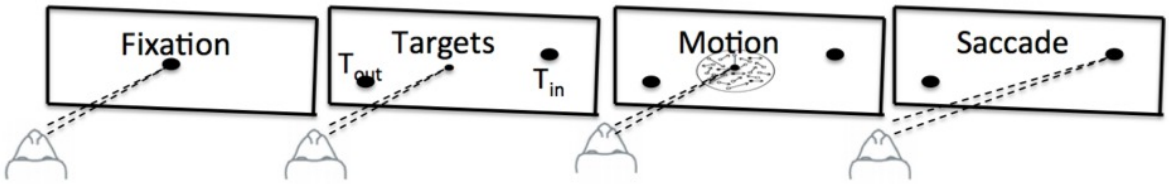
4 Methods

We model the decision making task as a partially observable Markov decision process (POMDP) [31, 84]. At any particular time t, the environment is assumed to be in a particular *hidden* state, $x \in \mathcal{X}$, that is not directly observable by the animal. The animal makes sensory measurements to observe noisy samples of this hidden state. At each time step, the animal receives a sample s_t generated from the environment as determined by an observation distribution, $\Pr[s_t|x]$. The animal maintains a posterior probability distribution (*belief*) over the set of possible true world states, given the observations $s_{1:t}$ it has received so far: $b_t(x) = \Pr[x|s_{1:t}]$. The animal's prior knowledge about the environment is given by the probability distribution $b_0(x)$.

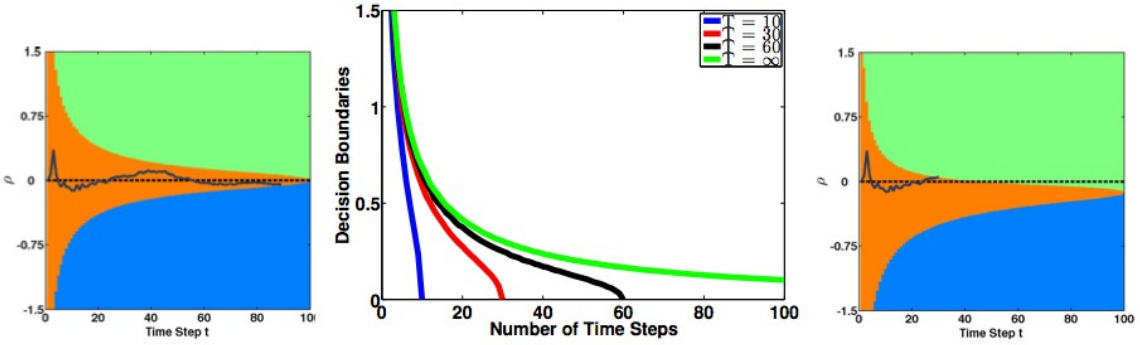
At each time step, the animal chooses an action $a \in \mathcal{A}$ and receives from the environment an

observation and a reward (or penalty) $R(x, a)$ that depends on the current state x and the action a. The model assumes that the animal uses Bayes' rule (or an approximate form of it) to update its belief about the environment after each observation. Given any belief b, the animal then uses a mapping from belief to action, known as a "policy" $\pi(b) \in \mathcal{A}$, to select an appropriate action for the current belief. We hypothesize that the animal's goal is to learn an optimal policy $\pi^*(b)$ that maximizes the animal's total expected future reward.

Details regarding the methods used to derive the optimal policy for the random dots task as well as the model of LIP responses for this task can be found in the *Supplementary Information*.



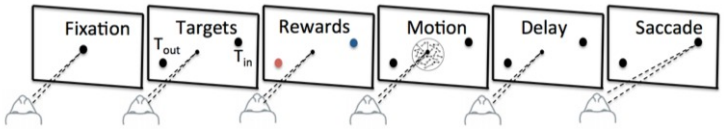
(a)



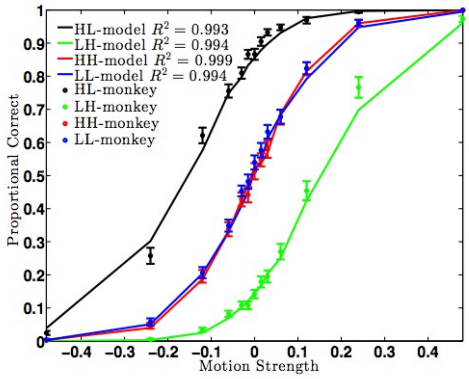
(b) (c) (d)

Figure 1

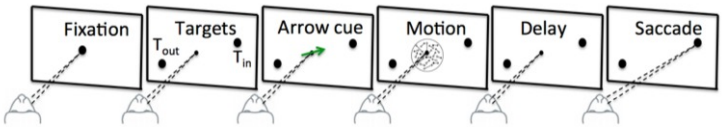
Figure 4.1 *(preceding page).* Optimal decision making in a sensory discrimination task. (*a*) Sequence of events in the reaction-time version of the random dots motion discrimination task as performed by a monkey. In the fixed-duration version, the monkey can make a decision (saccade) only after watching the motion stimulus for a fixed duration of time. (*b*) Optimal policy predicted by the model for the task in (a) when the prior probability of either motion direction is 0.5. Each point in the plot represents a particular belief state. The solid trajectory illustrates an example of how beliefs are updated at 0% motion strength. The color of each point represents the optimal action for that belief value. The policy partitions the belief space into three regions: upper ("rightward" action), middle ("sample" action) and lower ("leftward" action). Note the collapsing boundary between the decision regions. (*c*) The effects of task duration on decision boundaries in the fixed-duration version of the task. Note that in the reaction time task, the corresponding duration is infinite. (*d*) Optimal policy when the prior probability of rightward motion is 0.9. The same trajectory shown in (b) results in a different action when the prior is changed.


Fixation
Targets
T_{out}
T_{in}
Rewards
Motion
Delay
Saccade

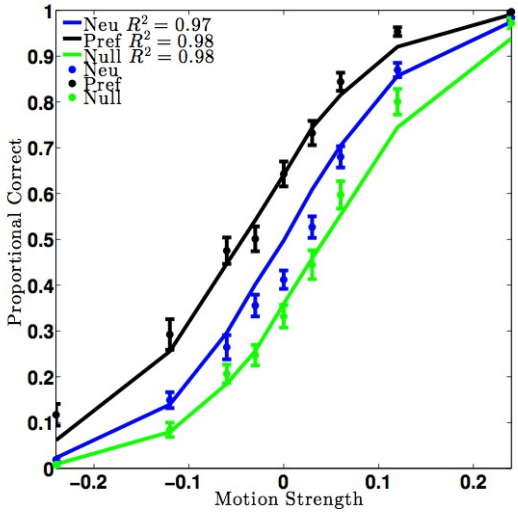
(a)


HL-model $R^2 = 0.993$
LH-model $R^2 = 0.994$
HH-model $R^2 = 0.999$
LL-model $R^2 = 0.994$
HL-monkey
LH-monkey
HH-monkey
LL-monkey
Proportional Correct
Motion Strength

(b)


Fixation
Targets
T_{out}
T_{in}
Arrow cue
Motion
Delay
Saccade

(c)


Neu $R^2 = 0.97$
Pref $R^2 = 0.98$
Null $R^2 = 0.98$
Neu
Pref
Null
Proportional Correct
Motion Strength

(d)

Figure 4.2 ***(preceding page).*** **Influence of rewards and prior knowledge on decision making.** (*a*) Sequence of events in the motion discrimination task with asymmetric rewards. The reward for a particular decision choice ("rightward" or "leftward") was chosen to be either high ("H") or low ("L"), resulting in four possible reward conditions. (*b*) Based on parameters fit to a monkey's behavioral data (points with error bars) for the HL condition (black), the model predicts the monkey's behavior for the HH (red), LL (blue), and LH (green) conditions. (*c*) Sequence of events in the motion discrimination task with prior cues. Before the onset of motion, one of three cues about the direction (a flashed unidirectional or bidirectional arrow) was given. (*d*) Based on the experimentally-set prior probability of $2/3$ and parameters fit to the preferred ("Pref") condition (black), the model predicts psychometric functions under the null (green) and neutral (blue) cue conditions. Experimental data [135, 130] are shown again as points with error bars.

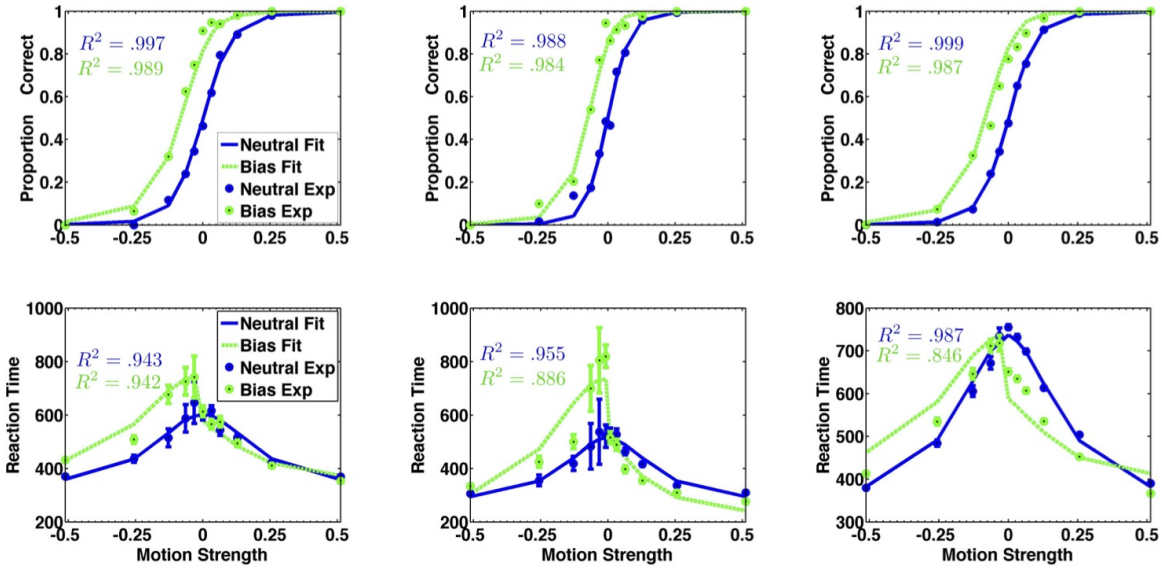
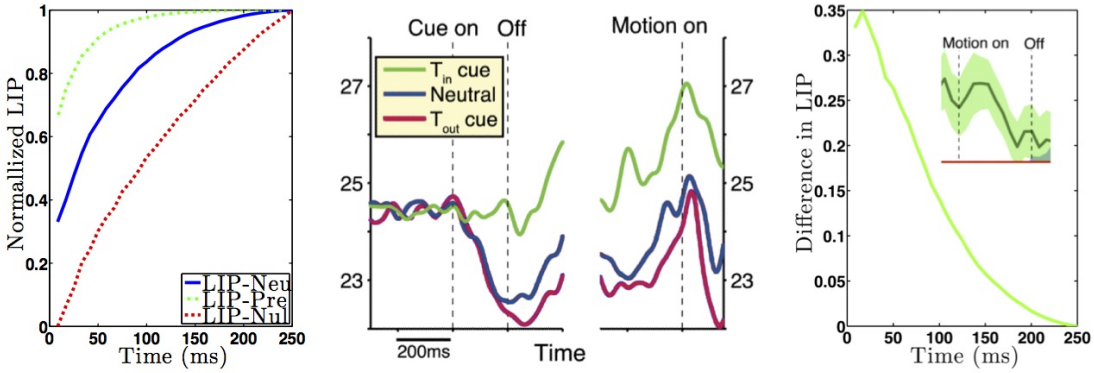
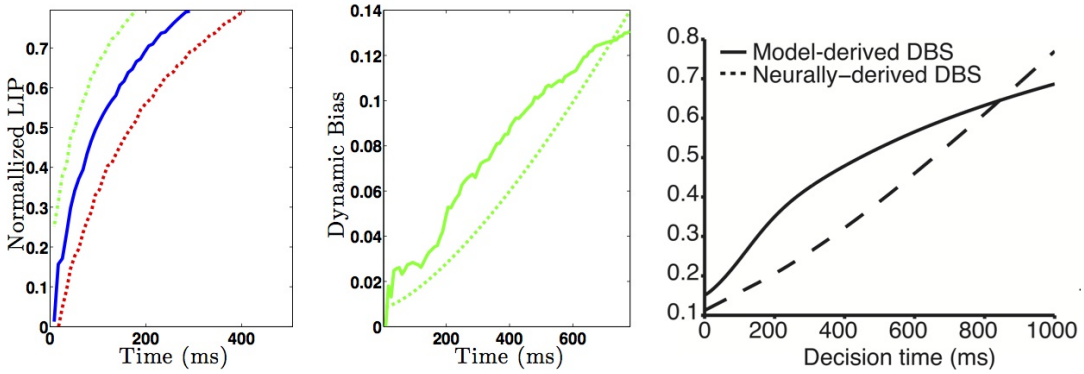


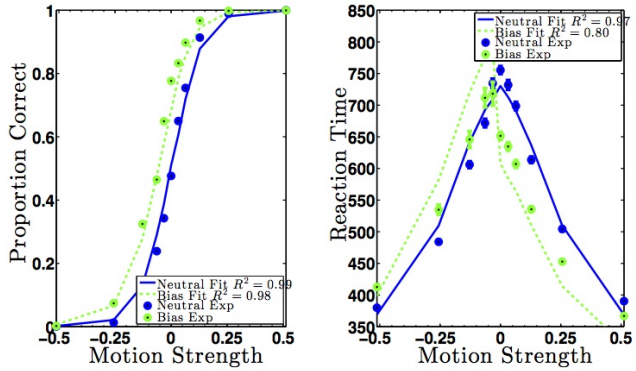
Figure 4.3. Comparison of model predictions with human and monkey data for reaction-time version of the task. The dots with error bars represent experimental data from human subject SK (a) and LH (b), and the combined results from four monkeys (c). The top panels show accuracy (the psychometric function) while bottom panels show reaction time (the chronometric function). In both cases, the model parameters were fit only to the human or monkey data for the neutral condition (both directions equiprobable; blue curves) and the model predicted the subject's performance for the biased condition (prior probability for rightward direction = 0.8; green curves).


Normallized LIP
Time (ms)
LIP-Neu
LIP-Pre
LIP-Nul
Cue on
Off
Motion on
T_in cue
Neutral
T_out cue
200ms
Time
Difference in LIP
Motion on
Off
Time (ms)

(a) (b) (c)


Normallized LIP
Time (ms)
Dynamic Bias
Time (ms)
Model-derived DBS
Neurally–derived DBS
Decision time (ms)

(d) (e) (f)


Proportion Correct
Motion Strength
Neutral Fit $R^2 = 0.99$
Bias Fit $R^2 = 0.98$
Neutral Exp
Bias Exp
Reaction Time
Motion Strength
Neutral Fit $R^2 = 0.97$
Bias Fit $R^2 = 0.80$
Neutral Exp
Bias Exp

(g) (h)

Figure 4.4 ***(preceding page).*** **Comparison of model neuronal responses with experimental data [130, 74].** (a) Model LIP responses and (b) LIP data from a monkey for the fixed-duration motion discrimination task when prior probability of motion is 0.8 ("Pre"), 0.2 ("Nul"), and 0.5 ("Neu"). (c) Bias signal predicted by the model as given by the difference in model responses for the 0.8 and 0.5 probability conditions. The signal shows a decreasing trend, also observed in experimental data (inset plot)[130]. (d) Model LIP responses and (e) Bias signal (solid line, computed as in (c)). The signal increases over time, a trend also seen in (f) which shows the dynamic bias signal (DBS) for the model and data reported by Hanks et al.[74] We used the experimentally observed neural bias signal (dashed line) to derive a policy and predict the monkey's behavior. (g) and (h) show the neurally-derived policy's predictions of the monkey's accuracy and reaction time.

5 Appendix

5.1 Probabilistic Framework for Decision Making.

In the random dots motion discrimination task, the animal is shown a movie of randomly moving dots, a fraction of which are moving in the same direction (this fraction is called the coherence). The hidden state x for the task is composed of both the coherence $c \in [0, 1]$ and the direction of motion $d \in \{-1, 1\}$ (corresponding to leftward and rightward motion respectively).[1] Both are chosen by the experimenter at the beginning of each trial and neither is known to the animal. We model the movie as a sequence of samples $s_{1:t}$. Each sensory observation s_t is assumed to be sampled from an observation distribution: $s_t \sim \Pr[s_t|kc, d]$, where $k > 0$ is a free parameter that determines the scale of s_t.

In order to discriminate the direction of motion given the observed samples, we hypothesize that the animal computes (or approximates) the posterior probability $\Pr[x|s_{1:t}]$ of the joint hidden state $x = kdc$ using Bayes' rule. At each time step, the animal chooses one of three actions, $a \in \{A_R, A_L, A_S\}$, denoting rightward eye movement (indicating the decision "rightward motion"), leftward eye movement (indicating the decision "leftward motion"), and sampling (i.e., waiting for one more sample observation) respectively. When the animal makes a correct choice (i.e., a rightward eye movement $a = A_R$ when $x > 0$ or a leftward eye movement $a = A_L$ when $x < 0$), the animal receives a positive reward $R_P > 0$ (implemented in the monkey experiments as a juice reward). When an incorrect action is chosen, the animal receives no reward or a negative reward (penalty) $R_N \leq 0$ (often implemented as a time delay until the next trial). We assume that the animal is motivated by hunger or thirst to make a decision as quickly as possible and we model this with an arbitrary penalty $R_S = -1$, representing the metabolic cost to the animal for waiting for one more observation by choosing the sampling action A_S.

---

[1] For the random dots task, the hidden state is fixed within a trial. The POMDP framework however is considerably more general because it allows transitions between states and can be used to model more complex tasks with rich hidden state dynamics [103, 120, 72].

5.2 Bayesian Inference of Hidden State.

The prior probability for a specific motion direction (e.g., rightward direction d_R) for the model above is given by: $\Pr[d_R] = \Pr[d = 1] = \Pr[x > 0] = 1 - \Pr[d_L]$. In the experiments discussed in this paper, the subject is sometimes given this prior probability; animals can also learn it from experience across many trials. However, the prior probability over direction provides only partial knowledge about the hidden state x which also includes coherence. By expressing the distribution over coherence as a normal distribution (parameterized by μ_0 and σ_0), we obtain a piecewise normal distribution for the prior over the hidden state x:

$$b_0(x) = Z_0^{-1}\mathcal{N}(x \mid \mu_0, \sigma_0) \times \begin{cases} \Pr[d_R] & x \geq 0 \\ \Pr[d_L] & x < 0, \end{cases} \tag{4.1}$$

where $Z_t = \Pr[d_R](1-\Phi(0 \mid \mu_t, \sigma_t))+\Pr[d_L]\Phi(0 \mid \mu_t, \sigma_t)$ is the normalization factor and $\Phi(x \mid b) = \int_{-\infty}^{x} \mathcal{N}(x \mid \mu, \sigma)dx$ is the cumulative distribution function (CDF) of the normal distribution b parametrized by the mean μ and standard deviation σ. Note that the case of a uniform prior over coherence can also be modeled using the equation above with $\mu_0 = 0$ and $\sigma_0 = \infty$.

Similar to previous models for the motion discrimination task (such as the drift diffusion model), we assume the observation distribution is normally-distributed: $\Pr[s_t|x] = \mathcal{N}(s_t|x, \sigma_e^2)$. By applying Bayes' rule, we obtain a piecewise normal posterior distribution or belief:

$$b_t(x) = Z_t^{-1}\mathcal{N}(x \mid \mu_t, \sigma_t) \times \begin{cases} \Pr[d_R] & x \geq 0 \\ \Pr[d_L] & x < 0 \end{cases} \tag{4.2}$$

$$\text{where} \qquad \mu_t = \left(\frac{\mu_0}{\sigma_0^2} + \frac{t\rho(t)}{\sigma_e^2}\right) / \left(\frac{1}{\sigma_0^2} + \frac{t}{\sigma_e^2}\right), \tag{4.3}$$

$$\sigma_t^2 = \left(\frac{1}{\sigma_0^2} + \frac{t}{\sigma_e^2}\right)^{-1}, \tag{4.4}$$

and $\rho(t)$ is the running average given by:

$$\rho(t) = \sum_{t'=1}^{t} s_{t'}/t = s_t/t + \sum_{t'=1}^{t-1} s_{t'}/(t-1) * (t-1)/t \qquad = \frac{t-1}{t}\rho(t-1) + \frac{1}{t}s_t \tag{4.5}$$

Note that the posterior distribution depends only on ρ and t, which are the two sufficient statistics of the observation sequence $s_{1:t}$. For the case of a piecewise uniform prior, the variance $\sigma_t^2 = \frac{\sigma_e^2}{t}$, which decreases inversely in proportion to elapsed time. For simplicity, we fix $\sigma_e = 1$ because we can rescale the POMDP time step $t' = \frac{t}{\sigma_e}$ to compensate. We also use $\sigma_0 = \infty$ and $\mu_0 = 0$.

5.3 Optimal Policy for the Task.

We now use our formulation of beliefs above to derive the optimal action for any belief b_t for the random dots task. The model postulates that the animal's goal is to find an optimal *policy* $\pi^*(b_t)$ that maximizes the expected total future reward, starting at b_t. This expected reward is encapsulated in the *value function*, defined for any policy π as:

$$v^\pi(b_t) = \mathbb{E}\left[\sum_{k=1}^{\infty} r(b_{t+k}, \pi(b_{t+k})) \mid b_t, \pi\right] \tag{4.6}$$

where $r(b, a)$ is the reward function over belief states and the expectation is taken with respect to all future belief states $(b_{t+1}, \ldots, b_{t+k}, \ldots)$ when the animal is using π to make decisions.

Given the value function, the optimal policy is given by: $\pi^*(b) = \arg\max_\pi v^\pi(b)$. Note that since the belief b is parameterized by ρ and t as discussed above, the animal only needs to keep track of these and the policy is simply a function of these two parameters (as in Figure 1b).

The reward over beliefs $r(b, a)$ can be equivalently written as the expected reward over hidden states: $r(b, a) = \int_x R(x, a)b(x)dx$, which can be simplified to:

$$r(b,a) = \begin{cases} R_S, & a = A_S \\ Z^{-1}[R_P \mathbf{Pr}\left[d_R\right](1 - \Phi(0 \mid b)) + R_N \mathbf{Pr}\left[d_L\right]\Phi(0 \mid b)], & a = A_R \\ Z^{-1}[R_N \mathbf{Pr}\left[d_R\right](1 - \Phi(0 \mid b)) + R_P \mathbf{Pr}\left[d_L\right]\Phi(0 \mid b)], & a = A_L \end{cases} \tag{4.7}$$

where the belief b is parametrized by μ_t and σ_t in (4.3) and (4.4), respectively. The above equations can be interpreted as follows. With probability $\Pr\left[d_L\right] \cdot \Phi(0 \mid b)$, the hidden state x is less than 0, making A_R an incorrect decision and resulting in a penalty R_N if chosen. Similarly, action A_R is correct with probability $\Pr\left[d_R\right] \cdot [1 - \Phi(0 \mid b)]$ and earns a reward of R_P. The inverse is true for A_L. When A_S is selected, the animal simply receives an observation at a cost of R_S.

Computing the value function defined in Equation 4.6 involves an expectation with respect to future belief. Therefore, we need to compute the transition probabilities over belief states, $T(b_{t+1}|b_t, a)$, for each action. We now derive these probabilities for each possible action. When the animal chooses to sample, $a_t = A_S$, the animal's belief distribution at the next time step is computed by marginalizing over all possible observations [84]:

$$T(b_{t+1}|b_t, A_S) = \int_s \Pr[b_{t+1}|s, b_t, A_S] \Pr[s|b_t, A_S] ds \tag{4.8}$$

$$\text{where} \qquad \Pr[b_{t+1} \mid s, b_t, A_S] = \begin{cases} 1 & \text{if } b_{t+1}(x) = \Pr[s|x] b_t(x) / \Pr[s|b_t, A_S], \forall x \\ 0 & \text{otherwise;} \end{cases} \tag{4.9}$$

$$\text{and} \qquad \Pr[s \mid b_t, A_S] = \int_x \Pr[s|x] \Pr[x|b, a] dx = \mathbb{E}_{x \sim b}[\Pr[s|x]] \tag{4.10}$$

When choosing A_S, the agent does not affect the hidden state x, so given the current state b_t and a new observation s, the transition to the updated belief b_{t+1} is uniquely determined. Thus, $\Pr[b_{t+1} \mid s, b_t, A_S]$ is a delta function following Bayes' rule. The probability $\Pr[s \mid b_t, A_S]$ can be treated as a normalization factor and is independent of hidden state x.[2] Thus, the transition probability function, $T(b_{t+1} \mid b_t, A_S)$, is solely a function of the belief b_t and is a stationary distribution over the belief space.

When the selected action is A_L or A_R, the animal stops sampling and makes an eye movement to the left or the right respectively. To account for these cases, we include a terminal state Γ with zero-reward, i.e., $R(\Gamma, a) = 0, \forall a$, and absorbing behavior, i.e., $T(\Gamma|\Gamma, a) = 1, \forall a$. Whenever the animal chooses A_L or A_R, it immediately transitions to Γ: $T(\Gamma|b, a \in \{A_L, A_R\}) = 1, \ \forall b$, indicating the end of a trial.

Given the transition probability between belief states $T(b_{t+1}|b_t, a)$ as derived above and the reward function, we can convert our POMDP model into a Markov Decision Process (MDP) over the belief state. We can then use standard dynamic programming techniques (e.g., value iteration [149] for MDPs) to compute the value function in (4.6) and the optimal policy (an example of such a derived policy is Fig. 1b).

---

[2] Explicitly, $\Pr[s|b_t, A_S] = Z_t^{-1} \mathcal{N}(s|\mu_t, \sigma_e^2 + \sigma_t^2)[\Pr[d_R] + (1 - 2\Pr[d_R]) \Phi(0| \frac{\frac{\mu_t}{\sigma_t^2} + \frac{s}{\sigma_e^2}}{\frac{1}{\sigma_t^2} + \frac{1}{\sigma_e^2}}, (\frac{1}{\sigma_t^2} + \frac{1}{\sigma_e^2})^{-1}])$.

An approximately optimal policy can also be learned by trial and error using the neurally plausible method of temporal difference (TD) learning, as suggested previously[125], but we restrict our focus for this article to comparisons between predictions from the optimal policy (derived as above) and behavioral/neural data from decision making experiments.

5.4 Dependence of Optimal Policy on Rewards and Prior.

The optimal policy π^* as derived above using the value function depends on the reward parameters $\{R_P, R_N, R_S\}$. Specifically, the optimal action for a specific belief state is determined by the relative, not the absolute, value of the expected future reward. From Equation 4.7, we have

$$r(b, A_L) - r(b, A_R) \propto R_N - R_P. \tag{4.11}$$

Moreover, if the unit of reward is specified by the sampling penalty, the optimal policy π^* is determined by the ratio $\frac{R_N - R_P}{R_S}$ (in addition to the prior probability $\Pr[d_R]$).

5.5 Model Parameters and Predictions of Behavioral Data.

The policies in Figure 1 used the model parameters: $\frac{R_N - R_P}{R_S} = 1,000$, with prior being 0.5 for Figure 1b and 0.9 for Figure 1d.

For Figure 3, the model parameters were $\frac{R_N - R_P}{R_S} = 1,000$, $k = 1$, and $\sigma = 1$, with prior probability being $2/3$ for Figure 3c and 0.8 for Figure 3d.

Model parameters for Figure 4 were: $\frac{R_N - R_P}{R_S} = 1,000$, $k = 1.45$.

5.6 Stimulus Duration and the Effect of the Prior

. As noted in the article, the model predicts that the longer the fixed duration of the stimulus, the lesser the noise in the cumulative sensory observations (due to the law of large numbers), and the more accurate the performance of the monkey. For a hidden state value x with duration T, the observations $s_{1:T}$ are independent and identically distributed. Trials with four times longer duration have four times more observations, which halves the standard deviation σ_T according to

Equation 4.4. Given this fact and with other model parameters remaining the same, we can predict the accuracy function for different prior probabilities when the duration of the task is made four times longer. We compared these predictions to the experimental data and found a good fit, as shown in Figure 4.5.

5.7 Model of LIP Responses

. In our model we assume information regarding the motion of the random dots is received by cortical area LIP from neurons in cortical area MT [112, 139, 23, 142]. From Figure 1b, it is clear that for the random dots task, the animal does not need to store the entire two-dimensional optimal policy but only the two one-dimensional beliefs at the boundaries, labeled by ψ^R and ψ^L. This naturally suggests a neural mechanism for decision making. Similar to the drift diffusion model, we begin with the following log likelihood ratio (LLR) given the current belief b:

$$LLR(b) = \log \frac{1 - \Phi(0 \mid b))}{\Phi(0 \mid b))} \tag{4.12}$$

We hypothesize that the response of LIP neurons represents the sum of the $LLR(b_t)$ and the decrease in the collapsing decision threshold $LLR(\psi^R(t))$ at time t. In this model, a rightward eye movement is initiated only when this response reaches a ***fixed*** bound (in this case, 0). Thus, the firing rates in LIP are modeled as:

$$\lambda_R^{LIP}(t) = LLR(b_t) + \lambda_0^{LIP} - LLR(\psi^R(t)) \tag{4.13}$$

where $\lambda_0^{LIP} - LLR(\psi^R(t))$ represents the amount of deviation between the collapsing decision boundary and some fixed threshold λ_0^{LIP}.

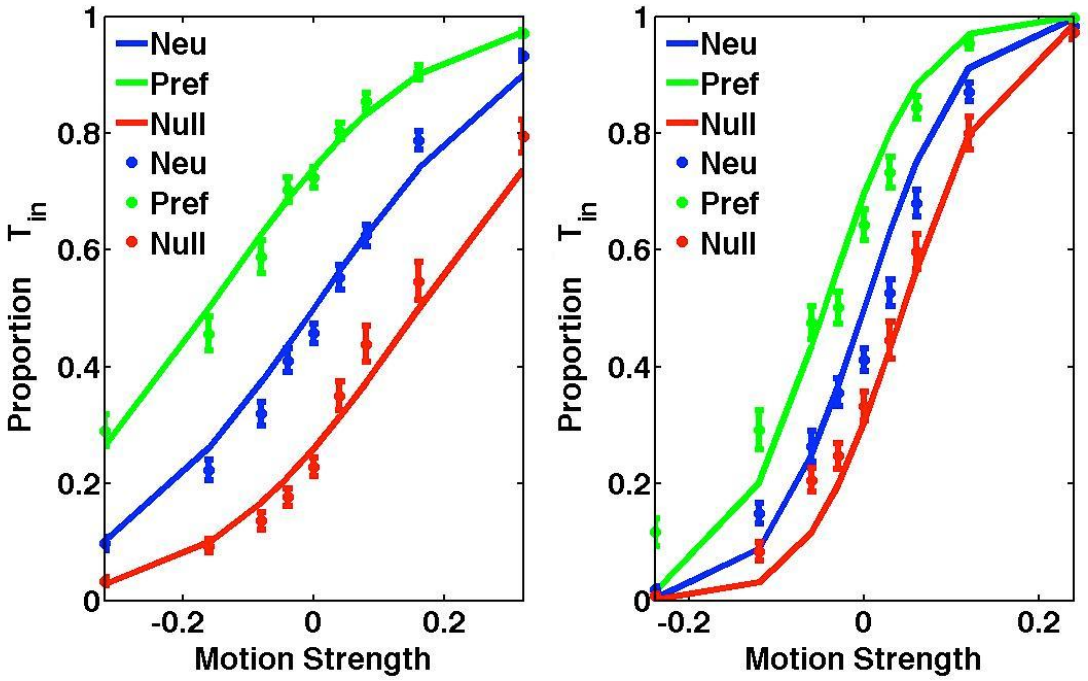


Figure 4.5. Effects of Stimulus Duration on Decision Making. Decision accuracies as a function of motion strength under different prior probabilities and different durations (250ms for the left and 1000ms for the right). Legend conventions are the same as in Fig. 3. Experimental data from Rorie et al.[135].

Chapter 5

FURTHER WORK: LEARNING EFFICIENT REPRESENTATIONS FOR REINFORCEMENT LEARNING

1 Introduction

This chapter considers sequential decision making problems where decisions can have both immediate and long-term effects. Each decision results in some immediate reward or benefit, but also affects the environment in which further decisions are to be made and thus affects the expected reward incurred in the future. The objective of the decision maker is to choose decision making policies optimally, that is, to maximize some long-term cumulative measurement of rewards. Such objective is challenging mainly because of the tradeoff between upfront and future rewards. Markov decision processes [121, 101] (MDPs) provides a mathematical formalization for this tradeoff.

1.1 Markov Decision Process

A MDP is mathematically defined in terms of a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where

- $\mathcal{S}$ is the finite set of all possible states that describes the context of the environment, also called the *state space*;
- $\mathcal{A}$ is the finite set of all actions the decision making agent can take;
- $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a transition function, a mapping specifying the probability $P^a_{s,s'}$ of going to state s' when performing action a in state s. An essential assumption made in the MDP is that the dynamics of state evolution is *Markovian*, meaning that the distribution of the next states is conditionally independent of the past, given the current state.

- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is a reward function. $R^a_{s,s'}$ describes a *finite* payoff or reward obtained when the agent goes from state s to state s' as a result of executing action a. The reward can be either positive or negative, representing an utility or a cost, respectively.

The optimality objective is to find a way or a *policy* to maximize some measure of the long turn reward received. A (stationary) policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ is a mapping from states to action, which specifies an action to be taken for each state. The choice of action is independent of the time, depends only on the state. Given a policy, we can define a *value* function $V_\pi(s)$ on the state space, which is the expected long run value an agent could expect to receive by choosing the action dedicated by the policy. A policy π_1 is said to dominate another policy π_2 if , $V_{\pi_1}(s) \leq V_{\pi_2}(s)$ for any state $s \in \mathcal{S}$, and $\exists s_1 \in \mathcal{S}$ such that $V_{\pi_1}(s_1) < V_{\pi_2}(s_1)$. A fundamental theorem [10] in MDP stated that there exists a stationary policy π^*, called the optimal policy, that dominates or has equal value to all other policies. The existence of such an optimal policy relies on the assumption that the expected long term reward, which is the objective function in the MDP, accumulates additively over time. That is to say, at each state, the optimal policy ranks the actions based on the sum of the expected rewards of the current time step and the optimal expected rewards of all subsequent steps.

To ensure the value function is well defined, one can limit the MDP to a finite number of time steps. In this case, the summation over rewards incurred in subsequent time steps terminates after a finite number of terms N, called the *horizon*, and the corresponding MDP is called a *finite horizon* MDP. The value of a policy π, starting from an initial state s_0, is

$$V^N_\pi(s) = \mathbb{E}[R(s_N) + \sum_{k=0}^{N-1} R(s_k, \pi(s_k), s_{k+1}) \mid s_0 = s] \tag{5.1}$$

where $R(s_N)$ is a terminal reward for ending up with the final state s_N, and the expectation is taken with respect to the probability distribution of the Markov Chain $\{s_0, s_1, \ldots, s_N\}$ starting at the initial state s, with transition probability matrix $P^{\pi(s_k)}_{s_k,s_{k+1}}$. The optimal value function and the

optimal policy is denoted by $V^{*N}(s)$ and $\pi^*(s)$, respectively; that is,

$$V^{*N}(s) = \max_{\pi} v_{\pi}^{N}(s) \tag{5.2}$$

$$\pi^*(s) = \operatorname*{argmax}_{\pi} v_{\pi}^{N}(s) \tag{5.3}$$

Despite the simple mathematical properties of the finite horizon MDPs, in many tasks, the reward is accumulated over an infinite (or indefinite) sequence of time steps. We refer this kind of tasks as the *infinite horizon* problems. There are three principal classes of infinite horizon problems.

(a) **Discounted problems**. Here we introduce a discount factor γ with $0 \leq \gamma < 1$. The reward incurred at the tth transition is *discounted* by a factor γ^t. Then the value function over an infinite number of time steps is given by

$$V_{\pi}(s) = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k R(s_k, \pi(s_k), s_{k+1}) \mid s_0 = s] \tag{5.4}$$

In our assumption, the one step reward $R_{ss'}^{a}$ is bounded from above by some constant, say, M. Therefore, $v_{\pi}(s) \leq \sum_{t=0}^{\infty} \gamma^t M = \frac{M}{1-\gamma}$, the infinite sum of decreasing geometric progression is finite for all policies π in all situations.

(b) **Stochastic Shortest Path Problems**. Here $\gamma = 1$ but we assume that there exists some additional termination state. Once the Markov chain reaches the termination state it remains there without any further rewards. The rewards (costs) associated with other states are negatively. In addition, the Markov chain is assumed to be such that termination is inevitable within finite number of steps, at least under an optimal policy. Thus, the problem is in effect a finite horizon one, but the length of horizon may be random. It can be shown that any discounted problems can be converted to a stochastic shortest path problem.

(c) **Average reward problems**. Without the discount factor, the sum over an infinite sequence of rewards may be infinite, however, it turns out that in many problems the average reward per

time step, given by

$$\tilde{V}_{\pi}^{N} \lim_{N\to\infty} \frac{1}{N} V_{\pi}^{N}(s) \tag{5.5}$$

where $V_{\pi}^{N}(s)$ is the N-horizon value function of policy π starting at state s, is well defined as a limit and is finite.

The optimal value function $V^*(s)$ can be shown to satisfy the well known *Bellman equation*

$$V^*(s) = \max_{a\in\mathcal{A}} \mathbb{E}[R(s,a,s') + \gamma V^*(s')]. \tag{5.6}$$

1.2 Representations of MDPs

Exact solutions to MDP, such as value iteration [16], policy iteration [78], and linear programming [40], involve a *lookup table representation* of the value function, in the sense that the whole vector $V(s)$ is kept in memory for each state s. The complexity of these algorithms are at least polynomial [116] in the size of the state space $|\mathcal{S}|$ as well as the size of action space $|\mathcal{A}|$. However, the order of the polynomials is large enough that those exact algorithms are not efficient in practice. The computation requirements of large scale MDP are still overwhelming. In such problems a sub-optimal approximation solution using *compact representation* of MDPs needed to be used. compact representations for approximately solving MDPs. Widely used compact representations include

- Construct a low dimensional vector space representation of the value function by building a set of linear basis functions [15].

- Kernel (instance) based methods [114] that represent the value function as a convex combination of observed values in the simulation samples.

- Factored MDPs [21] construct a representation of the state space using a vector of state variables, and represent the transition models between state variables using a dynamic Bayesian network.

- Hierarchical representations [39, 48] of MDPs exploit the task structure, where the actions are temporally extended.

- Symbolic representation of MDPs express the state space as binary decision diagrams(BDD) and algebraic decision diagrams(ADD) [77].

However, finding a good compact representations for a given reinforcement learning (RL) task requires carefully hand-coding by a human designer, which can be quite difficult and time consuming. We further review recent developments in automatic discovery of efficient representations in MDPs. We elaborate the problems of automatically constructing structured kernel for kernel based RL, a popular approach to learning non-parametric approximations for value function. We provide algorithms for exploring a space of kernel structures which are built compositionally from base kernels using a context-free grammar, and greedy algorithms for searching over the structure space.

2 Solutions for a Lookup Table Representation

In this section, we review basic solutions to MDP with a lookup table representation of value function.

There are two fundamental classes of exact solution methods to MDPs. The first approach is based on iterative algorithms that use dynamic programming, whereas the second approach formulates an MDP as a linear program. These exact solutions require a perfect knowledge of the explicit models of the reward structure and transition probabilities of the system, which many not be available. Simulation methods based on Monte Carlo simulations, instead requires only sample transitions (s_t, a_t, r_t, s_{t+1}) of the system.

The iterative algorithms typically employs the Bellman equation 5.6 to recursively relating the value of the current state to values of adjacent states. The form of Bellman equation motivates the introduction of two essential operators, also known as Bellman backup or dynamic programming backup operators in literature, that provide a convenient shorthand notation in expressions.

For any vector $V = (V(1), \ldots, V(|S|))$, we consider the vector TV obtained by applying one iteration of right hand side of Bellman equation:

$$(TV)(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p_{ss'}^{a} (R(s, a, s') + \gamma V(s')) \tag{5.7}$$

and similarly, for any vector V and any stationary policy π, we consider the vector $T_\pi V$ with components

$$(T_\pi V)(s) = \sum_{s' \in \mathcal{S}} p_{ss'}^{\pi(s)} (R(s, \pi(s), s') + \gamma V(s')) \tag{5.8}$$

Given a stationary policy π, we define the $|\mathcal{S}| \times |\mathcal{S}|$ matrix P_π whose (i, j) entry is $p_{i,j}^{\pi(i)}$. Then we can re-write $T_\pi V$ in matrix form as

$$T_\pi V = R_\pi + \gamma P_\pi V \tag{5.9}$$

where

$$R_\pi(s) = \sum_{s' \in \mathcal{S}} p_{ss'}^{\pi(s)} R(s, \pi(s), s') \tag{5.10}$$

We denote T^k and T_π^k as the operator obtained by applying the mapping T and T_π with themselves k times, respectively. It can be shown [15] that the following properties hold for T_π and T.

(a) The optimal value vector V^* is the only solution to the equation $V = T\ V$.

(b) We have $\lim_{k \to \infty} T^k V = V^*$. for every vector V

(c) A stationary policy is optimal if and only if $T_\pi V^* = TV^*$.

(d) For every vector V, we have $\lim_{k \to \infty} T_\pi^k V = V_\pi$. And V_π is the only solution of the equation $V = T_\pi\ V$

(e) The operator T is a contraction mapping with respect to a weighted maximum norm. That is, there exists a vector ρ of size $|\mathcal{S}|$ and a positive scalar $\beta < 1$ such that

$$\|TV - TV'\|_\rho \leq \beta \|V - V'\|_\rho \tag{5.11}$$

for all vectors V and V', and the weighted maximum norm is $\|V\|_\rho = \max_{s \in \mathcal{S}} \frac{|V(s)|}{\rho(s)}$

2.1 Value Iteration

A principal method, called value iteration, for calculating the optimal value V^* is to generate a sequence $T^k V$ starting from some vector V as $\lim_{k \to \infty} T^k V = V^*$. The value functions so computed are guaranteed to converge in the limit to the optimal value function. In the stochastic shortest path and average reward problems some additional assumptions for convergence are needed.

- *Finite (N) horizon problem*: the algorithm always converge in N steps.
- *Infinite horizon problems with discount rewards*: the algorithm always converges to the unique optimal solution.
- *Stochastic shortest path problem*: the algorithm converges if there is a policy with positive probability of termination after at most finite time steps, regardless the initial state.
- *Average Reward problems*: the algorithm converges if every state can be reached from every other state in finite time step with positive probability for some policy.

A commonly used stopping rule is to set $\epsilon = \epsilon' \frac{1-\gamma}{2\gamma}$, which ensures the resulting value function is within $\frac{\epsilon'}{2}$ of the optimal value function, and the resulting policy is ϵ'-optimal [159].

The running time for each iteration in algorithm1 is $O(|\mathcal{A}|\,|\mathcal{S}|^2)$. The number of iterations until convergence it shown [95] to be polynomial in the size of the state space $|\mathcal{S}|$ as well as the size of action space $|\mathcal{A}|$, which in turn makes value iteration polynomial in time. However, the order of the polynomials is nontrivial, thus in practice value iteration is usually inefficient.

Algorithm 1 Value Iteration

1: Initial V_0 arbitrarily for each state and $t = 0$
2: **repeat**
3: Compute $V_t = TV_{t-1}$
4: Compute Residual $e_t = \|V_t - V_{t-1}\|_{max}$
5: $t = t + 1$
6: **until** $e_t < \epsilon$
7: **return** Greedy policy $\pi(s) = \underset{a \in \mathcal{A}}{\text{argmax}} \sum_{s' \in \mathcal{S}} P^a_{ss'}[R(s, a, s') + \gamma V_t(s')]$

2.2 Policy Iteration

Another widely used iterative algorithm is known as policy iteration [78]. At each iteration, the decision maker first carries out a *policy evaluation* phase, in which the value function associated with the current policy is computed, and a *policy improvement* phase, in which a greedy attempt is made to improve the current policy.

The basic policy iteration algorithm is described in algorithm 2, where policy evaluation step

Algorithm 2 Policy Iteration

1: Let π_0 be some random initial policy and $t = 0$
2: **repeat**
3: Policy Evaluation: compute V_{π_t} in equation 5.12.
4: Policy Improvement: $\pi_{t+1}(s) = \underset{a \in \mathcal{A}}{\text{argmax}} \sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma V_{\pi_t}(s'))$, for all $s \in \mathcal{S}$
5: $t = t + 1$
6: **until** $\pi_{t+1}(s) = \pi_t(s)$, for all $s \in \mathcal{S}$

involves solving a system of $\mathcal{S}$ equations with $\mathcal{S}$ unknowns. Let ρ be the invariant distribution of a Markov chain P_π, and let $\mathcal{N}$ be the set of non-terminal states and $\mathcal{T} = \mathcal{S} - \mathcal{N}$ be the set of zero reward termination states in stochastic shortest path problems.

$$
\begin{aligned}
V_\pi(\mathcal{N}) &= (I - P_\pi(\mathcal{N},\mathcal{N}))^{-1}(R_\pi(\mathcal{N}) + P_\pi(\mathcal{N},\mathcal{T})R_\pi(\mathcal{T})) && \text{Stochastic Shortest Path} \\
V_\pi &= (I - \gamma P_\pi)^{-1} R_\pi && \text{Discounted Reward} \\
\tilde{V}_\pi &= (1 - P_\pi)^{-1}(R_\pi - \rho) && \text{Average Reward}
\end{aligned} \tag{5.12}
$$

For each iteration, policy evaluation phase can be performed in $O(|\mathcal{S}|^3)$ arithmetic operations and policy improvement in $O(|\mathcal{A}|\,|\mathcal{S}|^2)$ operations. When the number of states is large, it's usually preferable to carry out the policy evaluation phase by using iterative methods such as value iteration. It can be shown that the policy iteration algorithm generates an improving sequence of policies and terminates with an optimal policy. There is no theoretical guarantees for the number of iterations required, yet policy iteration has been listed as one of the preferred solution method for MDP.

2.3 Linear Programming

A third approach to solve MDPs exactly is based on linear programming [40]. The primal linear program involves

$$
\begin{aligned}
\text{Variables:} \quad & V(s), \quad \forall s \in \mathcal{S} \\
\text{Minimize:} \quad & \textstyle\sum \rho(s) V s \\
\text{Subject to:} \quad & V(s) \geq \textstyle\sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma V_{\pi_t}(s')) \quad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}
\end{aligned} \tag{5.13}
$$

where ρ is known as the state relevance weight vector whose elements are all positive. There are $|\mathcal{A}|\,|\mathcal{S}|$ constraints and $|\mathcal{S}|$ variables, one constrainst for each state s and action a. Thus, MDPs can be solve in polynomial time. A drawback of this algorithm is that it is typically slower than those iterative dynamic programming methods.

2.4 Temporal Difference Learning

In this subsection, we discuss an implementation of the Monte Carlo algorithm that incrementally updates the value function $V(s)$ after each transition. We first express the value function as

$$\begin{aligned} V_\pi(s_t) &= \mathbb{E}[\sum_{m=0}^{\infty} \gamma^m g(s_{t+m}, s_{t+m+1})] \\ &= \mathbb{E}[g(s_t, s_{t+1}) + \gamma V_\pi(s_{t+1})] \end{aligned} \tag{5.14}$$

The Robbins-Monro stochastic approximation method for solving the above expectation equation takes the form

$$\begin{aligned} \hat{V}(s_t) &= (1-\alpha_t)\hat{V}(s_t) + \alpha_t(g(s_t, s_{t+1}) + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t)) \\ &= (1-\alpha_t)\hat{V}(s) + \alpha_t d_t \end{aligned} \tag{5.15}$$

where $\alpha_t \in (0,1)$ is the learning rate and $d_t = g(s_t, s_{t+1}) + \gamma\hat{V}(s_{t+1}) - \hat{V}(s_t)$ is called the temporal difference (TD) [150], representing the difference between an estimate $g(s_t, \pi(s_t), s_{t+1}) + \gamma\hat{V}(s_{t+1})$ of the value function based on the one-step ahead simulated outcome of the current time step, and the current estimate $\hat{V}(s_t)$. Alternatively, we might fix a non-negative integer L and take into accounts the $L+1$-step ahead simulated outcome,

$$V_\pi(s_t) = \mathbb{E}[\sum_{m=0}^{L} \gamma^m g(s_{t+m}, s_{t+m+1}) + V_\pi(s_{t+L+1})] \tag{5.16}$$

We cannot assume one L better than another in the absence of any special knowledge. For the sake of generality, we may combine a weighted average of L-step Bellman equation 5.16 over all possible L. We introduce a constant $\lambda < 1$, multiply Eq.5.16 by $(1-\lambda)\lambda^L$, and sum over all non-negative L. We then have,

$$\begin{aligned} V_\pi(s_t) &= (1-\lambda)\mathbb{E}[\sum_{L=0}^{\infty} \lambda^L (\sum_{m=0}^{L} \gamma^m g(s_{t+m}, s_{t+m+1}) + V_\pi(s_{t+L+1}))] \\ &= \mathbb{E}[(1-\lambda)\sum_{m=0}^{\infty} g(s_{t+m}, s_{t+m+1}) \sum_{L=m}^{\infty} \lambda^m + \sum_{L=0}^{\infty} (\lambda^L - \lambda^{L+1}) V_\pi(s_{t+L+1})] \\ &= \mathbb{E}[\sum_{m=0}^{\infty} \lambda^m \gamma^m d_{m+t}] + V_\pi(s_t) \end{aligned} \tag{5.17}$$

The resulting Robbins-Monro stochastic approximation method is then

$$\hat{V}(s_t) = (1 - \alpha_t)\hat{V}(s_t) + \alpha_t \sum_{m=t}^{\infty} (\lambda\gamma)^{m-t} d_m \tag{5.18}$$

The above equation provides a family of algorithms, one for each λ, and is known as TD(λ). The choice of λ reflects a trade-off between bias and variance in the Monte Carlo based approximation. The general conclusion from [147] shows that intermediate values of λ seem to work best in practise. Sutton [150] has shown that under TD(0), the temporal difference algorithm converges to the true value function V_π. Dayan [38] extended this result to the case of general λ.

A temporal difference based method for learning action values called Q-learning was introduced by Waktins [158]. Q-learning updates directly estimates of the Q-factors associated with an optimal policy, thereby avoiding the multiple policy evaluation phases of policy iteration. The following learning rule for learning the action value function $Q(s, a)$ is used:

$$Q_{t+1}(s, a) = (1 - \alpha_t)Q_t(s, a) + \alpha_t(g(s, a, s') + \gamma \max_{a' \in \mathcal{A}} Q_t(s', a')) \tag{5.19}$$

where s' and $g(s, a, s')$ are generated from the pair (s, a) by simulation, according to the transition probability matrix $P^a_{ss'}$. Q-learning is sometimes referred to as an *off-policy* learning algorithm since it estimates the optimal action value function $Q(s, a)$ while simulation the MDP using any policy. During simulation, a sequence of states is generated with the greedy actions provided by the current available Q-factors. It's possible that certain profitable actions are never explored. In practice, variants of Q-learning algorithms with parameters control the degree of exploration are introduced to ensure sufficient exploration during simulations.

3 Compact Representation of Markov Decision Processes

The solutions described in previous section require a lookup table representations of the value function $V(s)$ with size $|\mathcal{S}|$. In environments with large discrete state space is large or even with continuous state spaces, the time complexity of the MDP solution algorithms makes them inefficient in practise. In this section, we review a variety of compact representations for approximately solving

MDPs, including low dimensional vector space representations by constructing linear basis functions [15], instance based representations of value function using kernels in Hilbert space [114], factored representation [73], hierarchical representations [39, 48], and symbolic representations such as binary decision diagrams(BDD) and algebraic decision diagrams(ADD) [77]. All these approaches depend crucially on a choice of low dimensional compact representation of a MDP, and assume these are carefully provided by the human designer. The focus of this section is on approximation, rather than automatic representation discovery.

3.1 Linear Value Function Approximation

In this subsection, we consider the policy evaluation phase for a single stationary policy π. Thus we suppress in our notation for the value functions the dependence on π. We approximate the value function $V(s)$ with a linear architecture:

$$\hat{V}(s, w) = \phi(s)'w, \quad \forall s \in \mathcal{S} \tag{5.20}$$

where w is a weight vector and $\phi(i)$ is an $|\mathcal{D}|$-dimensional feature vector associated with state s. That is, we represent the value function in a compact form $V \approx \hat{V} = \Phi w$, where Φ is the $|\mathcal{S}| \times |\mathcal{D}|$ matrix that has as rows the feature vectors $\phi(s)$, $s \in \mathcal{S}$. Thus, we want to approximate the value function V with the subspace $\mathcal{D}$ spanned by $|\mathcal{D}|$ basis function, each of which is in the columns of Φ. The rank of matrix Φ is $|\mathcal{D}|$. Let Π be the projection operator on to the linear subspace, with respect to some norm $\|\cdot\|_\rho$:

$$\|V\|_\rho = \sqrt{\sum_{s \in \mathcal{S}} \rho_s V^2(s)}, \tag{5.21}$$

where ρ is a vector of positive components. ΠV is the unique vector in the subspace that minimizes $\|V - \Phi\, w\|_\rho$.

$$\Pi V = \Phi\, w_\Phi \tag{5.22}$$

$$w_V = \underset{w \in \mathbb{E}^{\mathbb{D}}}{\operatorname{argmin}} \|V - \Phi\, w\|_\rho^2 \tag{5.23}$$

By setting the gradient of Eq. 5.23 to 0, we have

$$\Pi = \Phi(\Phi' D_\rho \Phi)^{-1} D_\rho \tag{5.24}$$

where D_ρ is the $|\mathcal{S}| \times |\mathcal{S}|$ diagonal matrix whose entries are $\rho(s)$. Now consider the Bellman backup operator T_π updating projected value functions,

$$\begin{aligned} \Phi\ w &= \Pi T_\pi(\Phi\ w) \\ \Phi\ w &= \Pi[R_\pi + \gamma P_\pi \Phi\ w] \end{aligned} \tag{5.25}$$

This equation is known as the projected Bellman's equation. And the solution $\phi\ w_\Phi$ of this equation is the approximation to value function V_π in the subspace spanned by Φ. w_Φ satisfied

$$\begin{aligned} [\Phi' D_\rho (I - \gamma P_\pi)\Phi]\ w_\phi &= \Phi' D_\rho R_\pi \\ A w_\phi &= b \end{aligned} \tag{5.26}$$

and can be solved by matrix inversion $w = A^{-1}b$ or other iterative algorithms. It can be shown that both mapping T_π and ΠT_π are contraction [109] with respect to the weighted Euclidean norm $\|\cdot\|_\rho$, where ρ is the steady state probability vector of the Markov chain with transition probabilities P_π. Analog to value iteration, the so-called projected value iteration algorithm iteratively apply the contraction operator ΠT_π, starting with some arbitrary vector w_0

$$\Phi\ w_{t+1} = \Pi T_\pi(\Phi\ w_t) \tag{5.27}$$

However, the projected value iteration algorithm is not practical when $|\mathcal{S}|$ is large since $T_\pi(\Phi\ w_t)$ is of size $|\mathcal{S}|$, and the steady state probabilities ρ are assumbed to be known.

Alternative way to solve equation 5.26 from simulation trajectories sampled from the Markov chain associated with policy π. After collecting t samples we have

$$\hat{A}_t = \frac{1}{t+1} \sum_{k=0}^{t} \phi(s_k)(\phi(s_k) - \gamma \phi(s_{k+1}))' \tag{5.28}$$

$$\hat{b}_t = \frac{1}{t+1} \sum_{k=0}^{t} \phi(s_k) R(s_k, s_{k+1}) \tag{5.29}$$

Given $\hat{A}_t$ and $\hat{b}_t$, one can construct a simulation bases solution

$$w_t = \hat{A}_t^{-1}\hat{b}_t \tag{5.30}$$

This is known as the least square temporal difference (LSTD) method.

Similar to TD(λ) method, we can introduce a constant $\lambda < 1$ and define

$$\hat{A}_t^{\lambda} = \frac{1}{t+1}\sum_{k=0}^{t}\phi(s_k)\sum_{m=k}^{t}\gamma^{m-k}\lambda^{m-k}(\phi(s_m) - \gamma\phi(s_{m+1}))' \tag{5.31}$$

$$\hat{b}_t^{\lambda} = \frac{1}{t+1}\sum_{k=0}^{t}\phi(s_k)\sum_{m=k}^{t}\gamma^{m-k}\lambda^{m-k}R(s_m, s_{m+1}) \tag{5.32}$$

the corresponding matrix inversion solution $w_t = (\hat{A}_t^{\lambda})^{-1}\hat{b}_t^{\lambda}$ is called the LSTD(λ) method.

3.2 Factored Markov Decision Processes

When some structure knowledge about the state space is known, one can construct a *factored MDP* representation of the state space using a vector of state variables, and represent the transition models between state variables using a dynamic Bayesian network. In this way, the value function can be approximated by a linear combination of basis functions, where each basis function involves only a small subset of the state variables. In particular, Guestrin et al [73] proposed an algorithm that generalize exact linear programming using basis functions Φ.

$$\begin{aligned}
\text{Variables:} \quad & w_1, \ldots, w_{|\mathcal{D}|} \\
\text{Minimize:} \quad & \textstyle\sum_s \rho(s) \sum_i w_i \phi_i s \\
\text{Subject to:} \quad & \textstyle\sum_i w_i \phi_i(s) \geq \sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma \sum_i w_i \phi_i(s')) \quad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}
\end{aligned} \tag{5.33}$$

where ρ is known as the state relevance weight vector whose elements are all positive. The number of variables in linear program has now been reduced from $|\mathcal{S}|$ to $|\mathcal{D}|$, the number of basis function in sub-space $\mathcal{D}$. Without a factored representation of the state space, the number of constraints remains $|\mathcal{S}| \times |\mathcal{A}|$. For factored MDPs, the number of constraints can be reduced exponentially by exploiting conditional independence properties in the conditional probability table of the dynamic Bayesian network.

3.3 Kernel Based Reinforcement Learning

In the kernel based reinforcement learning (KBRL) algorithms [114, 83], value functions are approximated by a set of sample outcomes $\{s_t, a_t, r_t, s_{t+1}\}_{t=1}^{N_T}$. Specifically, KBRL approximates the outcome of an action a from a given state s as the convex combination of sampled outcomes of that action, weighted by a function of the distance between s and sampled states. Then the Bellman backup operator is represented by an operator T_K on the samples:

$$\hat{V}(s) = T_K V(s) = \max_{a \in \mathcal{A}} \hat{Q}(s, a) \tag{5.34}$$

$$\hat{Q}(s, a) = \sum_{t \in \{t: a_t = a\}} K_a(s_t, s)[r_t + \gamma V(s_{t+1})] \tag{5.35}$$

where the summation is over a subset of indices t where $a_t = a$, and the kernel $K_a(s_t, s)$ is normalized in the sense that for each state s and action a, $\sum_{t \in \{t: a_t = a\}} K_a(s_t, s) = 1$.

Kernel-based reinforcement learning has several promising properties. First, the operator T_K has a unique fixed point. One can obtain an algorithm analog to value iteration to solve the MDP by iteratively applying T_K. Second, the fix point of this operator converges in probability to the true value function for the Gaussian Kernel:

$$K_a(s_t, s) = \exp[-\frac{d^2(s_t, s)}{2\sigma^2}] \tag{5.36}$$

when the number of samples $N_T \to \infty$ and the bandwidth $\sigma \to 0$. The distance metric $d(s_t, s)$ denotes the distance function. However, the time complexity of KBRL is N_T^2), which make it impractical when the sample size is large. To make it practical, Kveton [89] employs an unsupervised learning method to cluster the simulation samples onto k representative ones, and is able to compute the optimal policy in $O(n)$ time assuming $k \ll n$ a constant regardless n. Another advantage of the kernel based methods is the straightforward incorporation of the structure knowledge of the state space by using the structure kernel [90], where the kernel $K_a(s_t, s)$ can be decomposed into a product of base kernels.

The kernel based algorithm defined above requires knowledge about the metric function of the state space. Alternatively, the Gaussian Process Temporal Difference (GPTD) [56] learning offers

a Bayesian solution. Consider an episode in which a terminal state is reached at time step $T+1$, with $r_{T+1} = V(X_{T+1}) = 0$. We have a generated model for the value function at state s_t:

$$V(s_t) = r_t + \gamma r_{t+1} + \ldots + \gamma^{T-t} r_T - \epsilon_t \tag{5.37}$$

with $\epsilon_t \sim \mathbb{N}(0, \sigma_t^2))$. In a matrix form, we have

$$Z_T r_{1:T} = V_{1:T} + \epsilon_{1:T} \tag{5.38}$$

$$r_{1:T} = H_{T+1} V_{1:T} + \epsilon'_{1:T} \tag{5.39}$$

where

$$Z_T = \begin{bmatrix} 1 & \gamma & \gamma^2 & \ldots & \gamma^T \\ 0 & 1 & \gamma & \ldots & \gamma^{T-1} \\ \ldots & & & \ldots & \\ 0 & 0 & 0 & \ldots & 1 \end{bmatrix} \quad H_T = Z_{T-1}^{-1} = \begin{bmatrix} 1 & -\gamma & 0 & \ldots & 0 \\ 0 & 1 & -\gamma & \ldots & 0 \\ \ldots & & & \ldots & \\ 0 & 0 & \ldots & 1 & -\gamma \end{bmatrix} \tag{5.40}$$

Assuming a state-wise noise model with $\epsilon_t \sim \mathbb{N}(0, \sigma^2)$, we have $\epsilon'_{1:T} \sim \mathbb{N}(0, \sigma^2 H_T H_T^T)$.

Since both the value prior and the noise are Gaussian, so is the posterior distribution of the value conditioned on an observed sequence of rewards $r_{1:T} = \{r_t\}_{t=1:T}$. The joint distribution between a test point $V(s^*)$ and the observed sequence is:

$$\begin{pmatrix} Z_T\, r_{1:T} \\ V(s^*) \end{pmatrix} = \mathbb{N}\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{bmatrix} K_T & K_T(s^*) \\ K_T(s^*)^T & K(s^*, s^*) \end{bmatrix} \right] \tag{5.41}$$

where K_T denotes the $T \times T$ matrix of the covariances evaluated at all pairs of observed states, and $K_T(s^*)$ denotes the $T \times 1$ vector of the covariances evaluated at pairs of observed state s_t and the test state s^*. The posterior mean and variance of the value at s^* are given, respectively, by

$$\hat{V}(s^*) = K_T(s^*)^T (K_T + \sigma I)^{-1}\, r_{1:T} \tag{5.42}$$

$$\mathrm{VAR}(\hat{\mathrm{V}}(\mathrm{s}^*)) = K(s^*, s^*) - K_T(s^*)^T (K_T + \sigma I)^{-1} K_T(s^*) \tag{5.43}$$

3.4 Hierarchical Methods

Another approach to solving MDPs with large state spaces is to treat them as a hierarchical of task structures. In many cases, hierarchical solutions don't aim at providing an optimal value function to a MDP problem, but focus on gaining efficiency in execution time and learning time. Hierarchical learners are commonly structured as *delegation* behaviors. Feudal Q-learning [39] involves a hierarchy of learning problems, with higher level agents being masters and lower level agents being slaves. The highest level agent receives rewards r_t and states s_t from the external environment. It learns a mapping from states s_t to some pre-defined intermediate commands and feeds the lower level slaves commands and corresponding rewards for taking actions that satisfy the command. The lower level agents learns a mapping from commands and states to external actions a_t. However, the set of intermediate commands and their associated reinforcement functions should be established in advance of the learning. Similarly, by assuming one can identify useful subgoals and define subtasks that achieve these subgoals, the MAXQ algorithms [48] that decompose the target MDP into a hierarchy of smaller MDPs were proposed. Using the MAXQ decomposition, the value function of the target MDP can be expressed as an additive combination of the value functions of the smaller MDPs. To amend restriction of human designed hierarchy, Mehta et al [106] further introduced an algorithm that can automatic discover the task hierarchy, given that the dynamic Bayesian networks associated with the action and reward models are provided, as well as successful sample trajectories following the optimal policy.

3.5 Symbolic Algorithms for Solving MDPs

We briefly discussed symbolic algorithms in this subsection. The key idea of symbolic algorithms is to compactly represent the MDP models (value function, transition probabilities, reward functions, etc) using decision diagrams, instead of using the table lookup representation. Similar to *aggregation* methods, these decision diagram representations cluster the states that share similar values. Instead of applying Bellman operator to each state, it is sufficient to update the subset of states with similar values as a whole at once, by just a single Bellman backup. This representation

allows one to describe a value function as a function of the variables describing the domain and speeds up the value iteration based algorithms. However, these symbolic algorithms assume states in the MDP be factored. That is, the state space $\mathcal{S}$ is factored into a set of d boolean state variables $s = \{s_1, \ldots, s_d\}$. Although any finite valued non boolean variable can be split into a number of boolean variables, it often makes the new state space using decision diagram representation larger than the original one using the lookup table representation.

4 Representation Learning in Markov Decision Processes

In this section, we discuss methods for constructing compact representation of MDPs.

4.1 Feature Generation through Automatic Basis Construction

The policy evaluation phase can be viewed as solving systems of linear equation of the form $Aw = b$. The Krylov space method has long been among the most successful methods currently available for efficiently solving systems of linear equations. The k-order Krylov subspace is the linear subspace spanned by the image of b under the first $k-1$ powers of A, that is,

$$Krylov_k(A, b) = \text{span}\{b, Ab, A^2b, \ldots, A^{k-1}b\} \tag{5.44}$$

For an MDP, typically we set $b = R_\pi$. The Krylov basis can be significantly accelerated by a computational trick called the Schultz expansion,

$$(1 - A)^{-1}b = (I + A + A^2 + \ldots)b = \prod_{k=0}^{\infty}(I + A^{2^k})b \tag{5.45}$$

For example, we can compute the policy evaluation phase as follows:

$$V_\pi = (1 - \gamma P_\pi)^{-1}R_\pi = \prod_{k=0}^{\infty}(I + (\gamma P_\pi)^{2^k})R_\pi \tag{5.46}$$

Another way to construct basis automatically is based on the residual error in the current feature set [118]. Formally, if Φ_k is the current set of basis functions, the Bellman error basis functions (BEBFs) add $\phi_{k+1} = R + \gamma P\Phi_k w_{\Phi_k} - \Phi_k w_{\Phi_k}$ as the next basis function.

It's been shown [117] that a basis Φ is not only useful in approximating value functions, but also induces a *low-dimensional* MDP. The induced approximate reward function R_π^Φ and approximate transition function P_π^Φ are defined as

$$R_\pi^\Phi = (\Phi' D_\rho \Phi)^{-1} \Phi' D_\rho R_\pi \tag{5.47}$$

$$P_\pi^\Phi = (\Phi' D_\rho \Phi)^{-1} \Phi' D_\rho P_\pi \Phi \tag{5.48}$$

where R_π^Φ is the projection of the reward function R_π onto the column space of Φ, with respect to $\|\cdot\|_\rho$. Similarly, P_π^Φ is the least square solution to the system $\Phi P_\Pi^\Phi \approx P_\Pi \Phi$. The exact solution to this approximate MDP is the same as that given by the exact solution to the original MDP projected onto the basis Φ.

Given basis constructed by Krylov space or BEBF methods with k basis functions, Mahadevan [100] propose the representation policy iteration algorithm, as described in Algorithm 3

Algorithm 3 Model-based representation policy iteration

1: Let π_0 be arbitrary policy and $t = 0$
2: **repeat**
3: Construct basis matrix Φ
4: From the MDP compute $R_{\pi_t}^\Phi$ and $P_{\pi_t}^\Phi$
5: Find the solution to $(1 - \gamma P_{\pi_t}^\Phi) w_\Phi = R_{\pi_t}^\Phi$
6: Project solution back to the original state space $V_{\pi_t}^\Phi = \Phi w_\Phi$.
7: Find the greedy policy π_{t+1} as in the policy improvement phase

$$\pi_{t+1}(s) = \operatorname*{argmax}_{a \in \mathcal{A}} \sum_{s'} P_{ss'}^a (R_{ss'}^a + \gamma V_{\pi_t}^\Phi(s')) \tag{5.49}$$

8: $t = t + 1$
9: **until** $\pi_t = \pi_{t+1}$
10: **return** π_{t+1}

4.2 Feature Generation through Adaptive State Aggregation

Another basis construction algorithm [13] called the *adaptive state aggregation* partitions the original state space $\mathcal{S}$ into a set of m subsets $\mathcal{S}_1, \ldots, \mathcal{S}_m$, where $\cup_{i=1}^{m} \mathcal{S}_i = \mathcal{S}$ and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$, for $i \neq j$. We can view state aggregation as a special form of basis matrix Φ, where each column represents an indicator function for each cluster. At each iteration, the algorithm first carries out the regular value iteration to compute V^{k+1}, then corrects, rather than projects, V^{k+1} using the basis matrix

$$V^{k+1} = V^k + \Phi\ w_\Phi \tag{5.50}$$

where w_Φ is the solution to the compact policy evaluation problem

$$w_\Phi = (I - \gamma P_\Pi^\Phi)^{-1} R_\Pi^\Phi \tag{5.51}$$

$$P_\Pi^\Phi = (\Phi' D_\rho \Phi)^{-1} \Phi' P_\pi \Phi \tag{5.52}$$

$$R_\Pi^\Phi = (\Phi' D_\rho \Phi)^{-1} \Phi' (T(V^k) - V^k) \tag{5.53}$$

To create the basis Φ automatically, Keller [86] proposed to use neighborhood component analysis (NCA), a supervised learning algorithm with the state s as the input attributes, and the Bellman error or the temporal difference error as the supervised signal. In this way, NCA places basis function in the lower-dimensional space. The new lower dimensional features are then added as new features for the linear function approximator.

4.3 Structure Learning in Factored MDPs

Factored MDPs [21, 73] compactly represent the transition and reward functions of a MDP using dynamic Bayesian networks (DBNs). Efficient algorithms based linear program were developed even when the state space is large. However, they require a complete knowledge of the transition and reward functions of the problem in advance. Structure learning algorithms [42], as sketched in Algorithm 4 has been proposed to learn these functions by simulation trials, where decision tree induction algorithms are used to learn a factor representation of the reward and transition functions. Given the sample transitions $\{s_t, a_t, r_t, s_{t+1}\}$ observed in a MDP system, decision

Algorithm 4 Structure Learning Algorithm for factored MDP

1: Initialization
2: **for** each time step t **do**
3: Given $s, \pi_{t-1}(s)$, observe s' and r
4: Update the factored representation of reward Fact(R_t) and transition Fact(P_t) functions.
5: Learn a policy π_t using structure value iteration or algorithms for factored MDP.
6: **end for**

tree induction algorithms learn the compact reward model with $\{s_t\}$ being example attributes and $\{r_t\}$ being example labels, and learn a conditional probabilities table representation of the transition model with $\{s_t\}$ being example attributes and $\{s_{t+1}\}$ being example labels. A χ^2 test is used to detect the independence between two random variables. After a factored representation of the model is learned incrementally, the improved policy can be obtained by an incremental version of structured value iteration [21]. At the next iteration, the agent will follow the ϵ-greedy variant of the updated policy and generate new simulation samples. The algorithm will again update its factored representation for the model.

4.4 Structure Discovery through Compositional Kernel Search

Unlike the parametric linear function approximation using basis Φ, Kernel-based reinforcement learning (KBRL) [114, 131] is a popular approach to learning a non-parametric representation of the value function, where the similarities between two states are captured by a kernel $K_a(s, s')$. In problems where the state space is factored and s can be expressed as a set of state variables, among which there exists some conditional independencies, structured kernels [90] should be used to capture the independent relationships. When the conditional independencies between the state variables are unknown in advance, kernel learning techniques need to be employed. By defining a space of kernel structures which are built compositionally from a context free grammar, a future direction for research can be designing a greedy search algorithm based on the previous works [71, 54] to search over the grammar and automatically choose the decomposition structure from raw

data by evaluation only a small fraction of all structures.

5 Related Work and Future Challenges

The representation learning methods described in this report can be applied to build representations from sampled examples over a large variety of problems in AI. They are also close related to recent work on manifold learning [136, 9] and spectral learning [110], which have largely been applied to nonlinear dimensionality reduction and semi-supervised learning problems on graphs. However, learning the compact MDP representation introduces new challenges not represented in supervised learning and dimensionality reduction, as the set of training examples is not available as a batch, but must be collected through active exploration of the state space. Another challenge for representation learning in reinforcement learning is how well a compact representation transfers from one problem to another.

Chapter 6

CONCLUSION

In this dissertation, we answer two major challenges we face to understand how mammals make decisions: 1) How do mammals compute probability and make inference under uncertainty. 2) How do mammals choose actions to achieve long term goals, given the noisy, stochatic nature of the external world.

We address the first challenge by describing a two-layer Poisson network model that encodes the posterior probability distribution of hidden world states as a sampled distribution represented by spikes across a neural population. We implement approximate inference and learning for a Bayesian Filter for a hidden Markov model in a recurrent neural network. We derive a Hebbian learning rule based on online EM and present experimental tests. As a major novelty, the authorwe propose that the stochasticity of synaptic transmission is directly involved in the implementation of stochasticity necessary for Monte Carlo sampling. Our model embraces many biological properties that are frequently observed in CNS neurons, such as divisive normallization, and spike-time dependent Hebbian plasticity.

The model for learning we have proposed builds on prior work on online learning [3, 107, 28, 27]. The online algorithm used in our model for estimating HMM parameters involves three levels of approximation. The first level involves performing a stochastic approximation to estimate the expected complete-data sufficient statistics over the joint distribution of all hidden states and observations. Cappe and Moulines [28] showed that under some mild conditions, such an approximation produces a consistent, asymptotically efficient estimator of the true parameters. The second approximation comes from the use of filtered rather than smoothed posterior distributions in equation 2.11. Although the convergence reported in the methods section is encouraging, a rigorous proof of convergence remains to be shown. The asymptotic convergence rate using only the fil-

tered distribution is about one third the convergence rate obtained for the algorithms in [107] and [28], where the smoothed distribution is used. The third approximation results from Monte-Carlo sampling of the posterior distribution in equation 2.12.

We address the second challenge by proposing an optimum decision theoretic model of evidence accumulation under uncertainty. Decision making requires the integration of multiple sources of information: prior knowledge, noisy sensory stimuli, and rewards. Bayesian models have proved effective for understanding this process, but are yet unable to explain how the brain incorporates prior information in sequential decision making tasks where decisions have both immediate and long-term effects. Here, we present a model combining Bayesian inference with the principle of optimality. Decisions are chosen so as to maximize cumulative expected rewards over the course of sequential decision making. The model is based upon partially observable Markov decision processes and provides a rational (Bayes optimal) account of temporal integration of uncertain sensory evidence. Crucially, it explains psychophysical data without invoking extra free parameters in contrast to existing models (using static priors or a time varying prior). In particular, we show that the urgency hypothesized in previous models emerges naturally as a collapsing decision boundary. We illustrate the explanatory power of their model using empirical results from monkeys and humans (psychometric and chronometric). We also provide a normative explanation for otherwise conflicting neurophysiological data from cortical area LIP.

We believe that the POMDP model provides a more versatile framework for decision making compared to the drift diffusion model, which can be viewed as a special case of sequential statistical hypothesis testing (SSHT) [91]. Sequential statistical hypothesis testing assumes that the stimuli (observations) are independent and identically distributed whereas the POMDP model allows observations be temporally correlated. The observations in the POMDP are conditionally independent given the hidden state μ, which evolves according to a Markov chain. Thus, the POMDP framework for decision making [60, 163, 145, 125, 80] can be regarded as a strictly more general model than the SSHT models.

Another advantage of a POMDP model is that the model parameters have direct physical interpretations and can be easily manipulated by the experimenter. Our analysis shows that the optimal

policy is fully determined by the reward parameters $\{R_P, R_N, R_S\}$. Thus, the model psychometric and chronometric functions, which are derived from the optimal policy, are also fully determined by these model parameters. Experimenters can control these reward parameters by changing the amount of awards for the correct/incorrect choices, or by giving subjects different speed instructions. This allows our model to make testable predictions, as demonstrated by the effects of the change in the reward ratios on the speed-accuracy trade-off. It should be noted that these reward parameters can be subjective and may vary from individual to individual. For example, R_P can be directly related to the external food or juice reward provided by the experimenter while R_S may be linked to internal factors such as degree of hunger or thirst, drive, and motivation. The precise relationship between these reward parameters and the external reward/risk controlled by the experimenter remains unknown. Our model thus provides a quantitative framework for studying this relationship between internal reward mechanisms and external physical reward.

The proposed model demonstrates how the monkey's choices in the random dots task can be interpreted as being optimal under the hypothesis of reward maximization. The reward maximization hypothesis has previously been used to explain behavioral data from conditioning experiments [37] and dopaminergic responses under the framework of temporal difference (TD) learning [140]. Our model extends these results to the more general problem of decision making under uncertainty. The model predicts psychometric and chronometric functions that are quantitatively close to those observed in monkeys and humans solving the random dots task.

We showed through analytical derivations and numerical simulation that the optimal threshold for selecting overt actions is a declining function of time. Such a collapsing decision bound has previously been obtained for decision making under a deadline [59, 125]. It has also been proposed as an ad-hoc mechanism in drift diffusion models [49, 92, 33] for explaining finite response time at zero percent coherence. Our results demonstrate that a collapsing bound emerges naturally as a consequence of reward maximization. Additionally, the POMDP model readily generalizes to the case of decision making with arbitrary numbers of states and actions, as well as time-varying state.

Our results suggest that decision making in the primate brain may be governed by the dual principles of Bayesian inference and reward optimality. Bayesian inference allows beliefs (posterior

probability estimates) to be computed from prior knowledge and sensory evidence accumulated over time. These beliefs are in turn used to select actions that maximize expected cumulative reward under uncertainty. Such a normative model differs from previous descriptive models of decision making [29, 134, 102] in that important attributes such as collapsing decision boundaries and dynamic weighting of prior information over time emerge naturally rather than as assumptions used to fit the data. Additionally, the approach can be readily generalized to allow time-varying hidden states [31, 84] and other complex dependencies between random variables (e.g., hierarchies), [103, 120, 72] opening the door to understanding more complicated decision making behaviors and their neural substrates.

Our model relies on the principle of reward optimization to explain how animals makes decisions. The same principle has been successfully applied to explain the phenomenon of collapsing decision boundaries in the random dots motion discrimination task [163, 53]. However, how an animals decision is influenced by changes in prior probability and the reward function has not been explained by previous models. Our model predicts that the optimal policy is a deterministic function of reward parameters and prior probability, which correspond to physical parameters that can be controlled in experiments. No extra free parameters are required. Moreover, the POMDP framework is general enough to model a large variety of real world decision making processes, including decision making tasks involving time dependent stimuli, in contrast to previous models that were restricted to i.i.d observations.

BIBLIOGRAPHY

[1] B. Delyon an dM. Lavielle and E. Moulines. Convergence of a stochastic approximation version of the em algorithm. *Ann. Statist*, 27(1):94–128, 1999.

[2] C.H. Anderson and D.C. Van Essen. Neurobiological computational systems. In J. M. Zurada, R. J. Marks II, and Robinson C. J., editors, *Computational Intelligence: Imitating Life*. New York: IEEE Press, 1990.

[3] C. Andrieu, A. Doucet, and V. Tadic. Online parameter estimation in general state-space models. *Proceedings of the 44th Conference on Decision and Control*, pages 332–337, 2005.

[4] B. Barbour, N. Brunel, V. Hakim, and J. Nadal. What can we learn from synaptic weight distributions? *Trends in Neurosciences*, 30(12):622 – 629, 2007.

[5] H.B Barlow. A theory about the functional role and synaptic mechanism of visual aftereffects. In C. Blakemore, editor, *Vision: Coding and Efficiency*, pages 363–375. Cambridge University Press, 1990.

[6] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Statist.*, 41(1):164–171, 1970.

[7] J.M. Beck, W.J. Ma, R. Kiani, T.D. Hanks, A.K. Churchland, J.D. Roitman, M.N. Shadlen, P.E. Latham, and A. Pouget. Bayesian decision making with probabilistic population codes. *Neuron*, 60(6):1142–1145, 2008.

[8] J.M. Beck and A. Pouget. Exact inferences in a neural implementation of a hidden Markov model. *Neural Computation*, 19(5):1344–1361, 2007.

[9] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1-3):209–239, 2004.

[10] Richard Ernest Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[11] P. Berkes, G. Orban, M. Lengye, and J. Fisher. Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science*, 331(6013), 2011.

[12] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[13] D. P. Bertsekas and D. A. Castañon. Adaptive Aggregation Methods for Infinite Horizon Dynamic Programming. *IEEE Trans. on Automatic Control*, 34(6):589–598, 1989.

[14] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[15] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

[16] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1st edition, 1996.

[17] O. Bobrowski, R. Meir, and Y. Eldar. Bayesian filtering in spiking neural networks: noise adaptation and multisensory integration. *Neural Computation*, 21(5):1277–1320, 2009.

[18] O. Bobrowski, R. Meir, S. Shoham, and Y. Eldar. A neural network implementation optimal state estimation based on dynamic spike train decoding. *Neural information procession systems 21*, 20:145–152, 2008.

[19] R. Bogacz, E. Brown, J. Moehlis, P. Hu, P. Holmes, and J. D. Cohen. The physics of optimal decision making: A formal analysis of models of performance in two-alternative forced choice tasks. *Psychological Review*, 113:700–765, 2006.

[20] R. Bogacz and T. Larsen. Integration of reinforcement learning and optimal decision making theories of the basal ganglia. *Neural Computation*, 23:817–851, 2011.

[21] Craig Boutilier, Richard Dearden, and Moises Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121:2000, 1999.

[22] J.S. Bridle. Alpha-nets: A recurrent "neural" network architecture with a hidden Markov model interpretation. *Speech Communication*, 9(1), 1990.

[23] K. H. Britten, M. N. Shadlen, W. T. Newsome, and J. A. Movshon. The analysis of visual motion: a comparison of neuronal and psychophysical performance. *J. Neurosci*, 12:4745–4765, 1992.

[24] K. H. Britten, M. N. Shadlen, W. T. Newsome, and J. A. Movshon. Responses of neurons in macaque MT to stochastic motion signals. *Vis Neurosci*, 1993.

[25] Bingni W. Brunton, Matthew M. Botvinick, and Carlos D. Brody. Rats and Humans Can Optimally Accumulate Evidence for Decision-Making. *Science*, 340(6128):95–98, 2013.

[26] L. Buesing, J. Bill, B. Nessler, and W. Maass. Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput Biol*, 7(11), 2011.

[27] O. Cappe. Online EM algorithm for hidden Markov models, 2009.

[28] O. Cappe and E. Moulines. Online EM algorithm for latent data models, 2009.

[29] R. H. S. Carpenter and M. L. L. Williams. Neural computation of log likelihood in the control of saccadic eye movements. *Nature*, 377:59–62, 1995.

[30] G. Casella and R. Berger. *Statistical Inference; 2nd edition*. Duxbury Press, 2001.

[31] Anthony R. Cassandra, Leslie Pack Kaelbling, and James A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 963–972, 1996.

[32] Frances S. Chance and L. F. Abbott. Divisive inhibition in recurrent networks. *Network*, 11:119–129, 2000.

[33] A. K. Churchland, R. Kiani, and M. N. Shadlen. Decision-making with multiple alternatives. *Nat. Neurosci.*, 11(6), 2008.

[34] P. Cisek, G.A. Puskas, and S. El-Murr. Decisions in changing conditions: The urgency-gating model. *Journal of Neuroscience*, 29(37):11560–11571, 2009.

[35] N. D. Daw, A. C. Courville, and D. S.Touretzky. Representation and timing in theories of the dopamine system. *Neural Computation*, 18(7):1637–1677, 2006.

[36] N.D. Daw and A.C. Courville. The pigeon as particle lter. *Advances in Neural Information Processing Systems*, 19, 2007.

[37] P. Dayan and N. D. Daw. Decision theory, reinforcement learning, and the brain. *Cognitive, Affective and Behavioral Neuroscience*, 8:429–453, 2008.

[38] Peter Dayan. The convergence of td(lambda) for general lambda. *Machine Learning*, 8:341–362, 1992.

[39] Peter Dayan and Geoffrey E. Hinton. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems 5*, pages 271–278. Morgan Kaufmann, 1993.

[40] D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Oper. Res.*, 51(6):850–865, November 2003.

[41] A.F. Dean. The variability of discharge of simple cells in the cat striate cortex. *Experimental Brain Research*, 44:437–440, 1981.

[42] Thomas Degris, Olivier Sigaud, and Pierre-Henri Wuillemin. Learning the structure of factored markov decision processes in reinforcement learning problems. In William W. Cohen and Andrew Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning*, volume 148 of *ACM International Conference Proceeding Series*, pages 257–264. ACM, 2006.

[43] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. ROy. Statist. Soc. Ser. B*, 39(1):1–38, 1977.

[44] S. Deneve. Bayesian spiking neurons i: Inference. *Neural Computation*, 20:91–117, 2008.

[45] S. Deneve. Bayesian spiking neurons ii: Learning. *Neural Computation*, 20:118–145, 2008.

[46] S. Deneve, J.R.Duhamel, and A. Pouget. Optimal sensorimotor integration in recurrent cortical networks: a neural implementation of kalman filters. *J. Neurosic*, 27(21):5744–5756, 2007.

[47] S. Deneve and A. Pouget. Bayesian estimation by interconnected neural networks. *Society of Neuroscience Abstracts*, 27(237.11), 2001.

[48] Thomas G. Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

[49] J. Ditterich. Stochastic models and decisions about motion direction: Behavior and physiology. *Neural Networks*, 19:981–1012, 2006.

[50] M. C. Dorris and D. P. Munoz. Saccadic probability influences motor preparation signals and time to saccadic initiation. *J. Neurosci*, 18:7015–7026, 1998.

[51] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer-Verlag, 2001.

[52] K. Doya, S. Ishii, A. Pouget, and R. P. N. Rao. *Bayesian Brain: Probabilistic Approaches to Neural Coding*. Cambridge, MA: MIT Press, 2007.

[53] J. Drugowitsch, R. Moreno-Bote, A. K. Churchland, M. N. Shadlen, and A. Pouget. The cost of accumulating evidence in perceptual decision making. *J. Neurosci*, 32(11):3612–3628, 2012.

[54] David K. Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. *CoRR*, 2013.

[55] A. S. Ecker, P. Berens, G.A. Kelirls, M. Bethge, N. K. Logothetis, and A. S. Tolias. Decorrelated neuronal firing in cortical microcircuits. *Science*, 327(5965):584–587, 2010.

[56] Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 201–208, New York, NY, USA, 2005. ACM.

[57] A.L. Fairhall, G.D. Lewen, W. Bialek, and R. de Ruyter van Steveninck. Efficiency and ambiguity in an adaptive neural code. *Nature*, 412(23):787–792, 2001.

[58] M. M. Fard and Joelle Pineau. PAC-Bayesian model selection for reinforcement learning. *In Advances in Neural Information Processing Systems*, 23, 2011.

[59] P. Frazier and A.J. Yu. Sequential hypothesis testing under stochastic deadlines. *In Advances in Neural Information Processing Systems*, 20:465–472, 2008.

[60] P. L. Frazier and A. J. Yu. Sequential hypothesis testing under stochastic deadlines. *In Advances in Neural Information procession Systems*, 20, 2007.

[61] T. F. Freund, K. A. Martin, P. Somogyi, and D. Whitteridge. Nervation of cat visual areas 17 and 18 by physiologically identified x- and y- type thalamic afferents. ii. identification of postsynaptic targets by gaba immunocytochemistry and golgi impregnation. *J. Comp. Neurol*, 242(2):275–291, 1985.

[62] F. Gabbiani, J. Midtgaard, and T. Knoepfl. Synaptic integration in a model of cerebellar granule cells. *J. Neurophysiol*, 72:999–1099, 1994.

[63] C. R. Gallistel. and A. P. King. *Memory and the computational brain: why cognitive science will transform neuroscience*. Blackwell/Maryland lectures in language and cognition. Wiley-Blackwell, 2009.

[64] W. Gerstner and W.M. Kistler. *Spiking Neuron Models. Single Neurons, Populations, Plasticity*. Cambridge University Press, 2.

[65] W. Gerstner and W.M. Kistler. *Theoretical neuroscience: computation and mathematical modeling of neural systems*. The MIT Press, 2001.

[66] G. M. Ghose and J. H. R. Maunsell. Attentional modulation in visual cortex depends on task timing. *Nature*, pages 616–620, 2002.

[67] J. I. Gold, C. T. Law, P. Connolly, and S. Bennur. The relative influences of priors and sensory evidence on an oculomotor decision variable during perceptual learning. *J. Neurophysiol*, 100(5):2653–2668, 2008.

[68] J.I. Gold and M.N. Shadlen. The influence of behavioral context on the representation of a perceptual decision in developing oculomotor commands. *J. Neurosci.*, 23(2):632–651, 2003.

[69] Robbe L T Goris, J Anthony Movshon, and Eero P Simoncelli. Partitioning neuronal variability. *Nature Neuroscience*, 17:858–865, 2014.

[70] Tom Griffiths. Neural implementations of importance sampling. *NIPS preprint*, 2008.

[71] Roger B. Grosse, Ruslan Salakhutdinov, William T. Freeman, and Joshua B. Tenenbaum. Exploiting compositionality to explore a large space of model structures. Technical report, MIT, 2012.

[72] Carlos Guestrin, Daphne Koller, and Ronald Parr. Solving factored pomdps with linear value functions. In *In IJCAI-01 workshop on Planning under Uncertainty and Incomplete Information*, 2001.

[73] Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Efficient solution algorithms for factored mdps. *Journal of Artificial Intelligence Research (JAIR)*, 19:399–468, 2003.

[74] T. D. Hanks, M. E. Mazurek, R. Kiani, E. Hopp, and M. N. Shadlen. Elapsed decision time affects the weighting of prior probability in a perceptual decision task. *Journal of Neuroscience*, 31(17):6339–6352, 2011.

[75] G.E. Hinton and T.J. Sejnowski. Optimal perceptual inference. *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition,*, 1983.

[76] Jr. Hodges, J. L. and Lucien Le Cam. The Poisson approximation to the Poisson binomial distribution. *The Annals of Mathematical Statistics*, 31(3):737–740, 1960.

[77] Jesse Hoey, Robert St-aubin, Alan Hu, and Craig Boutilier. Spudd: Stochastic planning using decision diagrams. In *In Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 279–288. Morgan Kaufmann, 1999.

[78] R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA, 1960.

[79] P.O. Hoyer, A. Hyrinen, and A.H. Arinen. Interpreting neural response variability as Monte Carlo sampling of the posterior. *Advances in Neural Information Processing Systems 15*, 2002.

[80] Y. Huang, A. L. Friesen, T. D. Hanks, M. N. Shadlen, and R. P. N. Rao. How prior probability influences decision making: A unifying probabilistic model. *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[81] Y. Huang and R. P. N. Rao. Reward optimization in primate brain: A pomdp model of decision making under uncertainty. *PLoS One*, 8(1), 2013.

[82] Yanping Huang and Rajesh P Rao. Neurons as monte carlo samplers: Bayesian inference and learning in spiking networks. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1943–1951. Curran Associates, Inc., 2014.

[83] Nicholas Jong and Peter Stone. Kernel-based models for reinforcement learning in continuous state spaces. In *ICML workshop on Kernel Machines and Reinforcement Learning*, June 2006.

[84] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[85] P. Kara, P. Reinagel, and R.C. Reid. Low response variability in simultaneously recorded retinal, thalamic, and cortical neurons. *Neuron*, 27(3):635–646, 2000.

[86] Philipp W. Keller, Shie Mannor, and Doina Precup. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 449–456, New York, NY, USA, 2006. ACM.

[87] D. Knill and W. Richards. *Perception as Bayesian inference*. Cambridage University Press, 1996.

[88] K. Kording and D. Wolpert. Bayesian integration in sensorimotor learning. *Nature*, 427:244–247, 2004.

[89] Branislav Kveton and Georgios Theocharous. Kernel-based reinforcement learning on representative states. In *Association for the Advancement of Artificial Intelligence*, 2012.

[90] Branislav Kveton and Georgios Theocharous. Structured kernel-based reinforcement learning. In *Association for the Advancement of Artificial Intelligence*, 2013.

[91] T. L. Lai. Nearly optimal sequential tests of composite hypotheses. *The Annals of Statistics*, 16(2), 1988.

[92] P. E. Latham, Y. Roudi, M. Ahmadi, and A. Pouget. Deciding when to decide. *Soc.Neurosci.Abstracts*, 740(10), 2007.

[93] C. T. Law and J. I. Gold. Reinforcement learning can account for associative and perceptual learning on a visual-decision task. *Nat. Neurosci*, 12(5):655–663, 2009.

[94] T. S. Lee and D. Mumford. Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America*, 20:1434–1448, 2003.

[95] Michael L. Littman, Thomas L. Dean, and Leslie Pack Kaelbling. On the complexity of solving markov decision problems. In *IN PROC. OF THE ELEVENTH INTERNATIONAL CONFERENCE ON UNCERTAINTY IN ARTIFICIAL INTELLIGENCE*, pages 394–402, 1995.

[96] R. D. Luce. *Response times: their role in inferring elementary mental organization*. Oxford University Press, 1986.

[97] C. J. H. Ludwig. Temporal integration of sensory evidence for saccade target selection. *Vision Research*, 49:2764–2773, 2009.

[98] W.J. Ma, J.M. Beck, P.E. Latham, and A. Pouget. Bayesian inference with probabilistic population codes. *Nature Neuroscience*, 9(11):1432–1438, 2006.

[99] W. Maass. On the computational power of winner-take-all. *Neural Computation*, 12(11), 2000.

[100] Sridhar Mahadevan. Representation policy iteration. *CoRR*, abs/1207.1408, 2012.

[101] Mausam and Andrey Kolobov. *Planning with Markov Decision Processes: An AI Perspective*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2012.

[102] M. E. Mazurek, J. D. Roitman, J. Ditterich, and M. N. Shadlen. A role for neural integrators in perceptual decision-making. *Cerebral Cortex*, 13:1257–1269, 2003.

[103] David A. McAllester and Satinder Singh. Approximate planning for factored pomdps using belief state simplification. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 409–416, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[104] M.R. Mehta, C.A. Barnes, and B.L. McNaughton. Experience-dependent, asymmetric expansion of hippocampal place field. *PNAS US*, 94:8918–8921, 1997.

[105] M.R. Mehta, M.C. Quirk, and M.A. Wilson. Experience-dependent asymmetric shape of hippocampal receptive fields. *Neuron*, 25(3):707–715, 2000.

[106] Neville Mehta, Soumya Ray, Prasad Tadepalli, and Thomas G. Dietterich. Automatic discovery and transfer of task hierarchies in reinforcement learning. *AI Magazine*, 32(1):35–50, 2011.

[107] G. Mongillo and S. Deneve. Online learning with hidden Markov models. *Neural Computation*, 20:1706–1716, 2008.

[108] Martijn J Mulder, Eric-Jan Wagenmakers, Roger Ratcliff, Wouter Boekel, and Birte U Forstmann. Bias in the brain: a diffusion model analysis of prior probability and potential payoff. *J Neurosci*, 32(7):2335–43, 2012.

[109] Rmi Munos. Error bounds for approximate policy iteration. In *International Conference on Machine Learning*, 2003.

[110] Hariharan Narayanan, Mikhail Belkin, and Partha Niyogi. On the relation between low density separation, spectral clustering and graph cuts. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[111] B. Nessler, M. Pfeiffer, and W. Maass. Stdp enables spiking neurons to detect hidden causes of their inputs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1357–1365. MIT Press, 2009.

[112] W. T. Newsome, K. H. Britten, and J. A. Movshon. Neuronal correlates of a perceptual decision. *Nature*, 341:52–54, 1989.

[113] E.A. Nimchinsky, R. Yasuda, T.G. Oertner, and K. Svoboda. The number of glutamate receptors opened by synaptic stimulation in single hippocampal spines. *J Neurosci*, 24:2054–2064, 2004.

[114] Dirk Ormoneit and Saunak Sen. Kernel-based reinforcement learning. In *Machine Learning*, pages 161–178, 1999.

[115] J. Palmer, A. C. Huk, and M. N. Shadlen. The effects of stimulus strength on the speed and accuracy of a perceptual decision. *Journal of Vision*, 5:376–404, 2005.

[116] Christos Papadimitriou and John N. Tsitsiklis. The complexity of markov decision processes. *Math. Oper. Res.*, 12(3):441–450, August 1987.

[117] Ronald Parr, Lihong Li, Gavin Taylor, Christopher Painter-Wakefield, and Michael L. Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 752–759, New York, NY, USA, 2008. ACM.

[118] Ronald Parr, Christopher Painter-Wakefield, Lihong Li, and Michael L. Littman. Analyzing feature generation for value-function approximation. In *ICML*, pages 737–744, 2007.

[119] M G Paulin. Evolution of the cerebellum as a neuronal machine for bayesian state estimation. *J. Neural Eng.*, 2:S219–S234, 2005.

[120] Joelle Pineau, Nicholas Roy, and Sebastian Thrun. A hierarchical approach to pomdp planning and execution. In *Workshop on Hierarchy and Memory in Reinforcement Learning (ICML*, 2001.

[121] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[122] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[123] R. P. N. Rao. An optimal estimation approach to visual perception and learning. *Vision Research*, 39(11), 1999.

[124] R. P. N. Rao. Bayesian computation in recurrent neural circuits. *Neural Computation*, 16(1):1–38, 2004.

[125] R. P. N. Rao. Decision making under uncertainty: A neural model based on POMDPs. *Frontiers in Computational Neuroscience*, 4(146), 2010.

[126] R. P. N. Rao, B. A. Olshausen, and M. S. Lewicki. *Probabilistic Models of the Brain: Perception and Neural Function*. The MIT Press, 2002.

[127] R. P. N. Rao, B. A. Olshausen, and M. S. Lewicki. *Probabilistic Models of the Brain: Perception and Neural Function*. Cambridge, MA: MIT Press, 2002.

[128] R.P.N. Rao. Bayesian inference and attentional modulation in the visual cortex. *Neuroreport*, 16(16):1843–1848, 2005.

[129] R.P.N. Rao and D.H. Ballard. Dynamic model of visual recognition predicts neural response properties in the visual cortex. *Neural Computation*, 9(4), 1997.

[130] V. Rao, G. C. DeAngelis, and L. H. Snyder. Neural correlates of prior expectations of motion in the lateral intraparietal and middle temporal areas. *The Journal of Neuroscience*, 32(29):10063–10074, 2012.

[131] Carl Edward Rasmussen and Malte Kuss. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems 16*, pages 751–759. MIT Press, 2004.

[132] R. Ratcliff and G. McKoon. The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Computation*, 20:127–140, 2008.

[133] H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statist*, 22(3):400–407, 1951.

[134] J. D. Roitman and M. N. Shadlen. Response of neurons in the lateral intraparietal area during a combined visual discrimination reaction time task. *Jounral of Neuroscience*, 22, 2002.

[135] A. E. Rorie, J. Gao, J. L. McClelland, and W. T. Newsome. Integration of sensory and reward information during perceptual decision making in lateral intraparietal cortex (LIP) of the macaque monkey. *PlosOne*, 5, 2010.

[136] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

[137] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[138] Stuart J. Russell, Peter Norvig, John F. Candy, Jitendra M. Malik, and Douglas D. Edwards. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[139] C. D. Salzman, K. H. Britten, and W. T. Newsome. Cortical microstimulation influences perceptual judgements of motion direction. *Nature*, 346:174–177, 1990.

[140] W. Schultz, P. Dayan, and P. R. Montague. A neural substrate of prediction and reward. *Science*, 275:1593–1599, 1997.

[141] T. J. Sejnowski. Higher-order Boltzmann machines. *AIP Conference Proceedings*, 151(1), 1986.

[142] M. N. Shadlen and W. T. Newsome. Motion perception: seeing and deciding. *Proc. Natl. Acad. Sci.*, 93:628–633, 1996.

[143] M. N. Shadlen and W. T. Newsome. Neural basis of a perceptual decision in the parietal cortex (area LIP) of the rhesus monkey. *Journal of Neurophysiology*, 86(4), 2001.

[144] M.N. Shadlen and W.T. Newsome. Noise, neural codes and cortical organization. *Current Opinion in Neurobiology*, 4(4):569–579, 1994.

[145] P. Shenoy, R. P. N. Rao, and A. J. Yu. A rational decision-making framework for inhibitory control. *Advances in Neural Information Processing Systems (NIPS)*, 23, 2010.

[146] P. Shenoy and A. J. Yu. Rational impatience in perceptual decision-making: a bayesian account of discrepancy between two-alternative forced choice and go/nogo behavior. *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[147] Satinder P. Singh and Peter Dayan. Analytical mean squared error curves in temporal difference learning. In *NIPS*, pages 1054–1060. MIT Press, 1996.

[148] R. J. Snowden, S. Treue, R. G. Erikson, and R. A. Andersen. The response of area mt and v1 neurons to transparent motion. *Journal of Neuroscience*, 11:2768–2785, 1991.

[149] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[150] Richard S. Sutton. Learning to predict by the methods of temporal differences. In *MACHINE LEARNING*, pages 9–44. Kluwer Academic Publishers, 1988.

[151] A. M. Thomson. Activity-dependent properties of synaptic transmission at two classes of connections made by rat neocortical pyramidal axons in vitro. *J. Physiol*, 502:131–147, 1997.

[152] D. J. Tolhurst, J. A. Movshon, and A. F. Dean. The statistical reliability of signals in single neurons in cat and monkey visual cortex. *Vision Research*, 23:775–785, 1983.

[153] M. Tsodyks and H. Markram. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proceedings of the National Academy of Sciences of the United States of America*, 94(2):719–723, 1997.

[154] W. M. Usrey, J. M. Alonso, and R. C. Reid. Synaptic interactions between thalamic inputs to simple cells in cat visual cortex. *J. Neurol*, 20(14):5461–5467, 2000.

[155] H. P. Wang, D. Spencer, J. M. Fellous, and T. J. Sejnowski. Synchrony of thalamocortical inputs maximizes cortical reliability. *Science*, 328(5974):106–109, 2010.

[156] X. J. Wang. Probabilistic decision making by slow reverberation in cortical circuits. *Neuron*, 36:955–968, 2002.

[157] B. Wark, A. L. Farihall, and F. Rieke. Timescales of inference in visual adaptation. *Neuron*, 61:750–761, 2009.

[158] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[159] Ronald Williams and Leemon C. Baird. Tight performance bounds on greedy policies based on imperfect value functions. Technical report, Northesaster University, College of Computer Science, 1993.

[160] R.C. Wilson and L.H. Finkel. A neural implmentation of the kalman filter. *Advances in Neural Information Processing Systems*, 22:2062–2070, 2009.

[161] C. F. J. Wu. On the convergence properties of the em algorithm. *Ann. Statist*, 11(1):95–103, 1983.

[162] S. Wu, D. Chen, M. Niranjan, and S.I. Amari. Sequential Bayesian decoding within a population of neurons. *Neural Computation*, 15, 2003.

[163] A.J. Yu and J.D. Cohen. Sequential effects: Superstition or rational behavior. *In Advances in Neural Information Processing Systems*, 21:1873–1880, 2008.

[164] O. Zeitouni and A. Dembo. Exact filters for the estimation of the number of transitions of finite-state continuous-time Markov process. *IEEE Trans. Inform. Theory*, 43(4), 1988.

[165] R. S. Zemel and P. Dayan. Distributional population codes and multiple motion models. *Advances in neural information procession system*, 11, 1999.

[166] R. S. Zemel, Q. J. M. Huys, R. Natarajan, and P. Dayan. Probabilistic computation in spiking populations. *Advances in Neural Information Processing*, 17:1609–1616, 2005.

[167] R.S. Zemel, P. Dayan, and A. Pouget. Probabilistic interpretation of population codes. *Neural Computation*, 10(2), 1998.

[168] K. Zhang, I. Ginzburg, B. L. McNaughton, and T. J. Sejnowski. Interpreting neuronal population activity by reconstruction: A unified framework with application to hippocampal place cells. *Journal of Neuroscience*, 16(22), 1998.

[169] R. S. Zuker and W.G. Regehr. Short-term synaptic plasticity. *Annual Review of Physiology*, 64:355–405, 2002.