LETTER Communicated by Uri Eden

Encoding and Decoding Spikes for Dynamic Stimuli

Rama Natarajan
rama@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 3G4

Quentin J. M. Huys
qhuys@gatsby.ucl.ac.uk
Peter Dayan
dayan@gatsby.ucl.ac.uk
Gatsby Computational Neuroscience Unit, University College London, London WC1N 3AR, U.K.

Richard S. Zemel
zemel@cs.toronto.edu
Department of Computer Science, University of Toronto, Ontario, Canada M5S 3G4

Naturally occurring sensory stimuli are dynamic. In this letter, we consider how spiking neural populations might transmit information about continuous dynamic stimulus variables. The combination of simple encoders and temporal stimulus correlations leads to a code in which information is not readily available to downstream neurons. Here, we explore a complex encoder that is paired with a simple decoder that allows representation and manipulation of the dynamic information in neural systems. The encoder we present takes the form of a biologically plausible recurrent spiking neural network where the output population recodes its inputs to produce spikes that are independently decodeable. We show that this network can be learned in a supervised manner by a simple local learning rule.

1 Introduction

Naturally occurring sensory stimuli are seldom static; they comprise a large number of sensory elements, the statistics of which at any given instant are often a function of what happened previously in time. Light levels change very quickly over a natural range; odor intensity often varies rapidly and unpredictably in nature; auditory stimuli such as vocal communication signals of various animals are made up of many sound elements, the amplitude and frequency of which change rapidly with time. Experimental studies

 © 2008 Massachusetts Institute of Technology

suggest that neural responses to such stimuli reflect the ongoing changes in stimulus dynamics, even those that occur on a millisecond timescale, resulting in stimulus-induced temporal correlations in the neural activity. For example, studies in the insect antennal lobe show that odor intensity is represented by a spatiotemporal pattern of activity that varies predictably with the fine-scale temporal dynamics of the odor (Vickers, Christensen, Baker, & Hildebrand, 2001).

Proper decoding or downstream interpretation of information in neural response to temporally correlated stimuli requires knowledge of the stimulus-induced correlations in neural activity. This is evident from studying the computational consequences of encoding and decoding stimulus trajectories in a Bayesian framework. Even for a simple encoding of a dynamic stimulus variable, decoding as a Bayesian inverse of the encoding model can be computationally difficult. In Huys, Zemel, Natarajan, and Dayan (2007), we considered noisy memoryless input spikes as being generated by an independent, inhomogeneous Poisson process. We then derived the instantaneous posterior distribution over stimulus position as a Bayesian inverse of the encoding model. Even in this constrained setup, our results showed that decoding correct estimates of the probability distribution can be a complex computational problem, requiring access to past spikes.

For stimuli with Markovian dynamics, the posterior distribution can be written in a recursive form that depends on only current spikes and a single population vector summarizing information in past spikes. If correlations extend further back in time (i.e., for stimuli with "smooth" autocorrelations), the complete history of population spikes is needed to properly interpret a new spike. This requirement is ecologically relevant, as natural trajectories tend to vary smoothly over time. In a network of neurons, the downstream neural population that obtains these spikes as inputs faces a complicated inference problem, as computational and behavioral constraints do not permit retaining the full spiking history. However, it may be that an adequate approximation to the spike statistics exists and that a simple representation of an approximate posterior can be maintained online.

In this letter, we investigate the capacity of a simple method of retaining the information online using a nonlinear recurrent neural network to form such an approximation. This network is an instantiation of a hypothesis about population coding: we suggest that one of the tasks at every level of processing is to form a new encoding of dynamic stimulus information to facilitate access to the relevant information by downstream neurons. The main idea in our approach is that a population of neurons might recode information in its inputs (i.e., generate a new set of spikes) such that the resulting representation will obviate the need for the interpretation of a spike to depend on past spikes. The objective is to recode the input representation in a form that is relevant and computationally sensible for

downstream processing. We further propose a particular representation where each spike can be decoded independent of others in the spike train, in a way that does not require maintaining the entire spike history; instead, only a summary of past spikes is maintained in the current belief state.

The rest of the letter is organized as follows. In section 2, we present a simple recurrent neural network for encoding stimulus trajectories and formulate a decoding scheme for recovering the distribution over a dynamic stimulus variable from neural responses. Section 3 describes how the network parameters can be learned in a supervised manner, using a local learning rule. Section 4 presents the instantiation of the network we employ in our simulations, particularly the encoding model for generating input spikes and its corresponding optimal decoder. Section 5 presents simulations that examine the efficacy of the representational scheme and analyzes the results.

2 Recoding in a Recurrent Network

Let us consider a dynamic stimulus variable, position s, that evolves over time, forming a trajectory defined by $\overrightarrow{s_T} = \{s_1, \ldots, s_t, \ldots, s_T\}$, where s_t is the position at time t. Note that we employ a discrete time representation. A population of neurons $i = \{1, 2, \ldots, N\}$ that responds selectively to the position generates a population spike sequence $\overrightarrow{\boldsymbol{\xi}_T} = \{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t, \ldots, \boldsymbol{\xi}_T\}$ where $\boldsymbol{\xi}_t$ is a binary spike vector at time t, such that ξ_t^i is 1 when neuron i spikes at time t.

We assume that the spikes $\overrightarrow{\boldsymbol{\xi}_T}$ are generated by an encoding model $P(\overrightarrow{\boldsymbol{\xi}_T}|\overrightarrow{s_T})$ that specifies the probability of a particular population spike sequence being evoked by the trajectory $\overrightarrow{s_T}$. Since neural spiking is stochastic and not capable of representing the stimulus value exactly, we consider the population response to implicitly define a distribution over likely stimulus trajectories. Then the posterior distribution $p(s_T|\overrightarrow{\boldsymbol{\xi}_T})$ over stimulus positions can be estimated as a Bayesian inverse of the respective encoding model. Thus, we adopt the filtering goal of estimating the posterior distribution over trajectories implied by the input spikes rather than the prediction one of making an estimate about the future course of the trajectory.

2.1 Population Dynamics. Let $j = \{1, 2, \ldots, M\}$ be a population of recurrently connected output neurons in a network, receiving inputs $\overrightarrow{\boldsymbol{\xi}_T}$ from the population $i = \{1, 2, \ldots, N\}$. We ascribe simple dynamics to the recurrent population, similar to the general spiking neuron model of Kistler, Gerstner, and van Hemmen (1997). The output population spikes are specified by $\overrightarrow{\boldsymbol{\rho}_T}$, where $\rho_t^j = 1$ if neuron j spikes at time t. Like the inputs, the output spikes also convey information about $\overrightarrow{s_T}$.

The strength of the synaptic connection from an input neuron i to an output neuron j is characterized by the weight W_{ij} and lateral connections

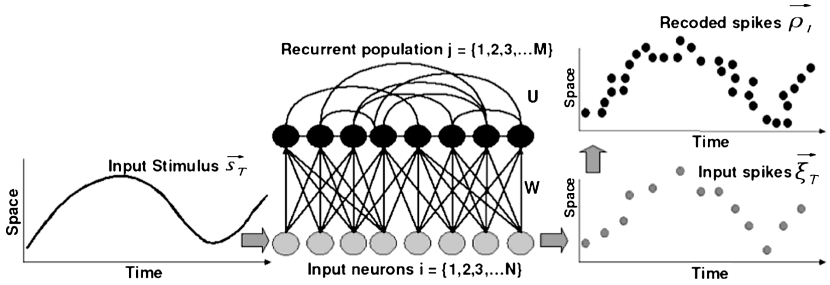

Figure 1: Schematic description of our network. (Left) Dynamic stimulus variable position s evolves over time in a one-dimensional space forming a *trajectory*. (Center) Sensory neurons $i = \{1, \ldots, N\}$ generate population spikes in response to the trajectory. Downstream population $j = \{1, \ldots, M\}$ receives the input via feedforward synapses $\mathbf{W}$ and recurrently processes the input using learned lateral synaptic weights $\mathbf{U}$ to recode the input into a representation that is independently decodable by downstream neurons. (Right) Input and recoded population spikes. Note the convention for illustrating the population spike train in this and all subsequent figures: the neurons are arranged along the y-axis based on their preferred stimulus values, and a shaded circle indicates that the neuron at that position spiked at the particular time.

between neurons j and k in the output population by U_{jk}. This scheme is illustrated in Figure 1. In a discrete time representation, the internal state (analogous to the membrane potential) of a neuron j at time T can be characterized by a continuous variable h_T^j:

$$h_T^j = \sum_{\tau=0}^{T-1} \sum_{i=1}^{N} \xi_{T-\tau}^i W_{ij} \eta(\tau) + \sum_{\tau=0}^{T-2} \sum_{k=1}^{M} \rho_{T-\tau-1}^k U_{kj} \eta(\tau). \tag{2.1}$$

The term $\eta(\tau)$ specifies the effect of a past spike on the current membrane potential; this typically has a form where temporally distant spikes have diminishing effects compared to more recent ones.

For each neuron j in the population, its response is governed by a stochastic binary spiking rule:

$$P\left(\rho_T^j = 1\right) = \sigma\left(h_T^j\right). \tag{2.2}$$

The spikes are generated independently among the population. This implies that the probability of any population binary (spike/no spike) vector $\boldsymbol{\rho}_T$ at time T is

$$P(\boldsymbol{\rho}_T | \vec{\boldsymbol{\xi}}_T) = \prod_j \sigma\left(h_T^j\right)^{\rho_T^j} \left(1 - \sigma\left(h_T^j\right)\right)^{\left(1-\rho_T^j\right)}, \tag{2.3}$$

where the sigmoid function is defined by $\sigma(h_T^j) = \frac{1}{1+\exp(-h_T^j)}$.

Note that the dynamics can be simplified considerably if the influence of past spikes is defined as

$$\eta(\tau) = \exp(-\beta\tau), \tag{2.4}$$

where β is the temporal decay constant. Using an exponentially decaying function to characterize the postsynaptic potential of a neuron allows us to express the dynamics of the output population in a recursive formulation:

$$h_T^j = \sum_i \xi_T^i W_{ij} + \sum_k \rho_{T-1}^k U_{kj} + \eta(1) h_{T-1}^j. \tag{2.5}$$

Then the probability of a population spike vector is a function solely of the membrane potential and population outputs at the previous time step and the current input spikes:

$$P(\boldsymbol{\rho}_T | \overrightarrow{\boldsymbol{\xi}_T}) = P(\boldsymbol{\rho}_T | \boldsymbol{h}_{T-1}; \boldsymbol{\rho}_{T-1}; \boldsymbol{\xi}_T). \tag{2.6}$$

2.2 An Independent Probabilistic Decoder. As stated in section 1, our objective is to encode dynamic stimuli into a representation that is computationally sensible for downstream decoding. By treating the spikes as if they were independent, formulating the decoder becomes highly simplified. However, a key question arises as to whether population spikes are naturally and faithfully decodable in this manner. Theoretical work on the effect of noise correlations on information decoding (Nirenberg, Carcieri, Jacobs, & Latham, 2001; Pola, Thiele, Hoffmann, & Panzeri, 2003) shows that it may be possible to decode independently despite correlations in the input. Hence, our primary hypothesis with respect to decoding is that each spike can be considered independently of the others in the spike sequence—that spikes from different neurons can be combined without taking correlations into account.

Even if downstream neurons, and indeed the organism as a whole, may never explicitly decode the information in the neural activity, making relevant information simply and readily interpretable can only ease the computational task faced by downstream neurons. Thus, decoding constitutes a canonical test for the computational consequences of the encoding scheme; if all the encoded information can be easily accessed, then subsequent manipulation of the information must be statistically easier to carry out. This is particularly important for dynamic stimuli since subsequent processing needs to be carried out at a pace relevant to the timescales of behaviorally significant decision making.

2.2.1 Formulating the Decoder. The decoding model specifies the probabilities of various stimulus trajectories based on the spike sequence. Rather than considering the complicated space of possible trajectories given an entire spike train, we focus on an instantaneous version of the problem. In mathematical terms, our aim is to decode at time T a distribution over the

stimulus position at that time, s_T, given all the spike observations up to that particular time, $\overrightarrow{\boldsymbol{\rho}_T} = \{\boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_T\}$; that is, we want to infer $q(s_T|\overrightarrow{\boldsymbol{\rho}_T})$ to closely approximate the true posterior distribution $p(s_T|\overrightarrow{\boldsymbol{\xi}_T})$.

The approximating probability distribution over stimulus trajectories $q(s_T|\overrightarrow{\boldsymbol{\rho}_T})$ is interpreted from the spiking activity as

$$q(s_T|\overrightarrow{\boldsymbol{\rho}_T}) \propto \exp\left(-E\left(s, T, \overrightarrow{\boldsymbol{\rho}_T}\right)\right). \tag{2.7}$$

Here, $E\left(s, T, \overrightarrow{\boldsymbol{\rho}_T}\right)$ is the total effect of the spike train specified as a linear combination of the individual spikes,

$$E\left(s, T, \overrightarrow{\boldsymbol{\rho}_T}\right) = \sum_{j=1}^{M} \sum_{\tau=0}^{T-1} \kappa_{\text{std}}(j, s, \tau) \rho^j_{T-\tau}, \tag{2.8}$$

where $\kappa_{\text{std}}(j, s, \tau)$ is the spatiotemporal decoding kernel (defined below). Equations 2.7 and 2.8 define our decoding hypothesis. The decoder is illustrated in Figure 2.

Under this model, a downstream neuron simply has to add the responses of all the neurons that impinge on it; linear functions of these afferent spikes establish the information available in the population inputs (Hinton & Brown, 2000). The information is thus readily accessible: if the convolution of spikes with the spatiotemporal kernels $\kappa_{\text{std}}(j, s, \tau)$ can be considered as defining the postsynaptic potentials of neurons, then information about the relative probabilities of stimulus values can be thought of as being accessed by simply adding (and exponentiating) the neuronal excitatory postsynaptic potentials (EPSPs).

2.2.2 Spatiotemporal Kernels. We define the kernels $\kappa_{\text{std}}(j, s, \tau)$ (see equation 2.8) as being uniform and separable—uniform in that the shape of the kernel is the same for the entire population, and at all times, and separable as follows,

$$\kappa_{\text{std}}(j, s, t) = \phi_j(s)\psi(t), \tag{2.9}$$

where $\phi_j(s)$ is the spatial component and $\psi(t)$ is the temporal component. We will henceforth refer to these kernels as *standard kernels*. Then the spatiotemporal contribution of the spikes (see equation 2.8) can be rewritten as follows:

$$E\left(s, T, \overrightarrow{\boldsymbol{\rho}_T}\right) = \sum_j \left[\sum_{\tau=0}^{T-1} \psi(\tau) \rho^j_{T-\tau}\right] \phi_j(s). \tag{2.10}$$

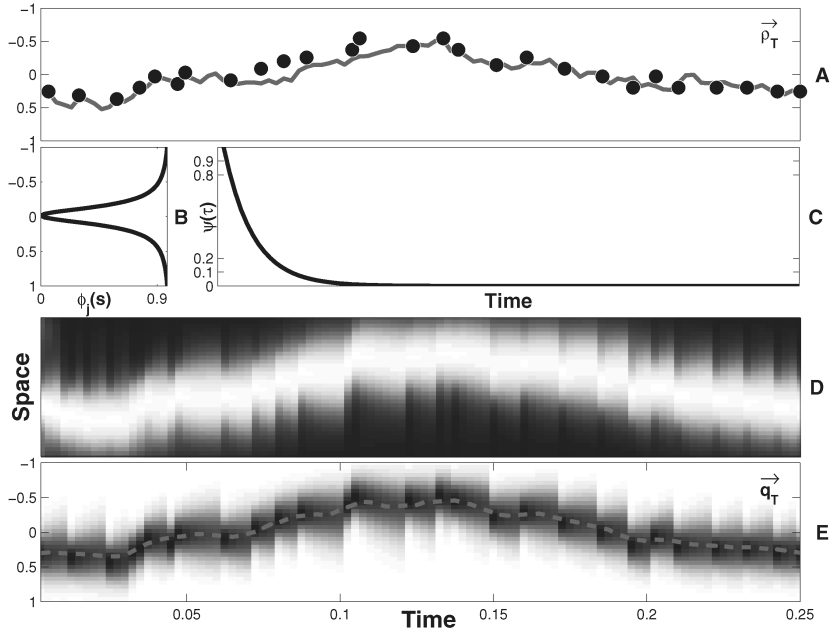

Figure 2: Independent decoding model. (A) Output spikes (solid black circles) of population $j = \{1, 2, \ldots, M\}$ convey information about the stimulus trajectory (solid gray line). (B, C) An example of the form of spatial and temporal components of the decoding kernel. (D) Linear contribution of the population spikes through time (as in equation 2.10). (E) Inferred posterior distribution (shaded in gray scale) over stimulus positions through time; the posterior mean is in the dashed gray line.

The spatial dimension of the kernel is parameterized as

$$\phi_j(s) = \frac{|s - s_j|^2}{\omega}, \tag{2.11}$$

where ω is the projective width of the neurons and s_j is the preferred stimulus value of neuron j. Thus, $\phi_j(s)$ defines the influence of the output of a particular neuron j in the population $j = \{1, \ldots, M\}$, on the spatial interpretation of the underlying variable.

The temporal dimension of the kernel is specified as an exponential decay (analogous to the postsynaptic potential of the neuron, η in equation 2.5):

$$\psi(\tau) = \exp(-\gamma\tau). \tag{2.12}$$

The parameter γ, which we refer to as the *temporal integration constant*, specifies the timescale for synaptic integration in the readout. It effectively controls what effect the spiking of a neuron at one time has on the interpretation at future times. Hence, the kernels are very simple, specified using just three parameters: the preferred stimulus value s_j and spatial projective width ω for neuron j and the temporal integration constant γ.

3 Learning to Recode

Independent decoding of the input spikes $\overrightarrow{\boldsymbol{\xi}_T}$ is far from optimal, especially for the smooth case. This decoder cannot by itself access the information in the input spikes that is due to the stimulus autocorrelations. Thus, this choice of output decoder shifts the onus of providing the information about the stimulus correlations from the decoder onto the network. In essence, the task of the network is to produce spikes that represent the information contained in the stimulus autocorrelations to make up for the fact that the decoder of its output spikes $\overrightarrow{\boldsymbol{\rho}_T}$ will, by itself, neglect any such information in the inputs $\overrightarrow{\boldsymbol{\xi}_T}$.

The overall aim of recoding is to form output spikes such that the posterior obtained by decoding these spikes according to equations 2.7 and 2.8 will faithfully approximate the optimal decoding of input spikes. We adopt a learning approach here, under the assumption that effective recoding can be achieved by learning from patterns of input spikes the spatial and temporal regularities governing the stimulus trajectories, that is, the prior over trajectories. In this section, we elaborate how the weight vectors **W**, **U** and the temporal decay constant β are learned using an online procedure.

3.1 Learning Algorithm. Learning occurs through the observation of a large number of trajectories that have the same spatial and temporal regularities. Since the observed spikes are nondeterministically related to the population inputs (see equation 2.6), we employ a form of reinforcement learning to improve the stochastic spiking behavior of the output neurons. The population spiking probabilities $P(\boldsymbol{\rho}_T|\overrightarrow{\boldsymbol{\xi}_T})$ can be thought of as defining a stochastic policy that maps the observations $(\boldsymbol{h}_{T-1}; \boldsymbol{\rho}_{T-1}; \boldsymbol{\xi}_T)$ into the probability distributions over the output neurons. That is, the spiking policy defines how each neuron should make the choice of being in one of two states (spike or not spike) at each time step, based on the observations. The policy is directly parameterized by the weights **W** and **U** and the temporal decay constant β, which we summarize as Ω.

The learning objective is to maximize long-term average reward,

$$J(T) = \frac{1}{T}\sum_{t=1}^{T} v(\Omega, t), \tag{3.1}$$

where $v(\Omega, t)$ is the reward signal at a particular time t.

This reward signal could take the form of some delayed reward based on behavior governed in some manner by the population spikes. In this letter we consider a more informative reward signal, a real-valued global signal broadcast to all neurons at time T,

$$v(\Omega, T) = D_{KL}\left(p(s_T|\overrightarrow{\boldsymbol{\xi}_T})||q(s_T|\overrightarrow{\boldsymbol{\rho}_T})\right), \tag{3.2}$$

where $D_{KL}(p(s)||q(s)) = \sum_s p(s)\log\frac{p(s)}{q(s)}$ is the Kullback-Leibler (KL) divergence, or relative entropy between the true distribution $p(s)$ and the model distribution $q(s)$. This is a natural way to measure the discrepancy between two probability distributions, having an extensive, information-theoretic justification (MacKay, 2003; Cover & Thomas, 2006). This signal entails knowing the posterior distribution $p(s_T|\overrightarrow{\boldsymbol{\xi}_T})$, based on the information available in the input spikes $\overrightarrow{\boldsymbol{\xi}_T}$. We adopt this very informative learning signal in order to assess whether a simple network can effectively combine temporal priors with the population inputs to generate an independently decodable representation.

The gradient with respect to Ω of the expected reward is not computable in closed form since the output spikes are stochastically sampled. Therefore, we resort to a stochastic gradient approach (Baxter & Bartlett, 2001), where the gradient is estimated via simulation (see section A.1) and the policy is improved by adjusting the parameters in the gradient direction. The update rule for all the parameters in Ω has a simple gradient ascent form:

$$\Omega \leftarrow \Omega + \frac{\partial J}{\partial \Omega}. \tag{3.3}$$

The dominant term in the gradients of J with respect to the weights $\mathbf{W}$ and $\mathbf{U}$, respectively, takes the form (see equations A.13 and A.18 in the appendix):

$$v(T)\xi_T^i\left[\rho_T^j - \sigma\left(h_T^j\right)\right] \qquad v(T)\rho_{T-1}^k\left[\rho_T^j - \sigma\left(h_T^j\right)\right], \tag{3.4}$$

where we have dropped the dependence on Ω for ease of exposition. The details are provided in sections A.2 and A.3. The update rule resembles contrastive Hebbian learning (Ackley, Hinton, & Sejnowski, 1985). Here, $\sigma(h_T^j)$ is the predicted activity, and ρ_T^j is the true stochastic activity. The update rule incorporates activity-dependent synaptic modification and a mechanism that induces competition between synapses so that when certain synapses to a postsynaptic neuron are strengthened, others are weakened. The gradient with respect to β is derived in section A.4.

The reward signal $v(T)$ conveys information on the state of all the synapses at time T. The competition between synapses can be thought of as

being caused by the term $(\rho_T^j - \sigma(h_T^j))$ that modifies the synaptic strengths. This term equilibrates when the expected predicted activity is equal to the expected true activity. Therefore, the unpredictability in spiking drives learning; learning stops when the distribution of responses under the model matches their empirical distribution.

In this formulation, the output units are conditionally independent given the input spikes $\boldsymbol{\xi}_T$; they are, however, marginally dependent (see equation 2.3). The network learns to optimize the forward and lateral connections under this independence assumption, and since the decoder also follows this assumption, the system consistently operates as if spikes from each neuron can be independently interpreted by downstream neurons. The fidelity of independent decoding on recoded spikes is evaluated using the following measure for information loss:

$$I_L = \frac{1}{T}\sum_{t=1}^{T} \frac{D_{KL}(p(s_T|\overrightarrow{\boldsymbol{\xi}_T})||q(s_T|\overrightarrow{\boldsymbol{\rho}_T}))}{H(p(s_T|\overrightarrow{\boldsymbol{\xi}_T}))}, \tag{3.5}$$

where $H(p)$ is the entropy of p.

4 Network Instantiation

This formulation is not specific to any particular encoding or decoding model, provided that there is an appropriate ideal observer that reports $p(s_T|\overrightarrow{\boldsymbol{\xi}_T})$. We simulated the special case that the input spikes $\overrightarrow{\boldsymbol{\xi}_T}$ are generated by a Poisson-gaussian spike generation model, that leads to a particularly simple ideal observer (Huys et al., 2007) with a revealing structure.

4.1 Spike Generation Model. We adopt a standard (tuning curve plus noise) spike generation model for the set of all the J spikes $\boldsymbol{\xi}_\varsigma \equiv \{\xi_{t_j}^i\}_{j=1}^J$ at times $0 < \{t_j\}_{j=1}^J \leq T$ evoked by the trajectory $\overrightarrow{s_T}$, where ς is a collection of all spike times. Under this model, the population response is governed by neurons whose expected responses are defined by partially overlapping gaussian tuning functions and whose stochastic observed responses are modeled as Poisson variables. The spikes are probabilistically related to the stimulus by means of a tuning function defined for each neuron i in the population as follows:

$$f_i(s_{t_j}) = r_{max} \exp\left(-\frac{(s_{t_j} - \theta_i)^2}{2\sigma^2}\right), \tag{4.1}$$

where s_{t_j} is the value of the stimulus variable at time t_j, θ_i is the preferred stimulus value of neuron i, $r_{\max}$ is the maximum input firing rate, and σ is the tuning width. By this simple gaussian definition, each neuron fires

maximally at its preferred value θ_i, and the activity drops off monotonically according to σ as the stimulus s_{t_j} drifts away from θ_i.

The actual observed spikes are generated as inhomogeneous and instantaneous Poisson processes governed by the tuning functions (Brown, Frank, Tang, Quirk, & Wilson, 1998; Barbieri et al., 2004):

$$P(\boldsymbol{\xi}_\varsigma|\overrightarrow{s_T}) \propto \left(\prod_j f_{i(j)}(s_{t_j})\right) \exp\left(-\sum_i \sum_t f_i(s_t)\right). \tag{4.2}$$

This spiking model entails some assumptions about the tuning properties and response of the input neurons. In our abstraction, the spikes are conditionally independent events given the stimulus, and the current response is independent of past activity. The tuning properties of the input population are constrained such that each neuron has the same maximum firing rate and tuning width. Furthermore, the tuning functions are assumed to span the stimulus state-space evenly and densely such that $\sum_i \sum_t f_i(s_t)$ is approximately a constant. This leads to

$$P(\boldsymbol{\xi}_\varsigma|\overrightarrow{s_T}) \propto \left(\prod_j f_{i(j)}(s_{t_j})\right). \tag{4.3}$$

When the tuning function from equation 4.1 is substituted, the input spike generation model is then

$$P(\boldsymbol{\xi}_\varsigma|\overrightarrow{s_T}) \propto r_{max} \exp\left(-\frac{(\boldsymbol{s}_\varsigma - \boldsymbol{\theta}(T))^T(\boldsymbol{s}_\varsigma - \boldsymbol{\theta}(T))}{2\sigma^2}\right), \tag{4.4}$$

where $\boldsymbol{s}_\varsigma$ is the vector of all the stimulus positions ordered by spike time and $\boldsymbol{\theta}(T)$ is the corresponding vector of preferred stimulus values of the spiking neurons.

4.2 Ideal Observer. In Huys et al. (2007), we provided the details for deriving the optimal posterior distribution as a Bayesian inverse of the above encoding model, together with a prior distribution over the stimulus trajectories. The posterior distribution provides the target for training the network parameters. Stimulus trajectories $\boldsymbol{s}_{(0,T]}$ were defined over continuous rather than discrete time and were assumed to be drawn from a gaussian process (GP). This means that for any finite collection of times $\mathcal{T} = \{t_i\}$, the stimulus values $\boldsymbol{s}_\mathcal{T}$ are drawn from a multivariate gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathcal{C}$,

$$p(\boldsymbol{s}_{(0,T]}) \sim \mathcal{N}(\mathbf{m}, \mathcal{C}) \qquad \mathcal{C}_{t_i t_j} = c \exp\left(-\alpha \|t_i - t_j\|^\zeta\right), \tag{4.5}$$

where $\mathcal{C}$ is the matrix of $\mathcal{C}_{t_i t_j}$ at the discretized times. Different values of ζ determine the different classes of prior distributions, and c parameterizes the overall scale of the process. The value of α scales the temporal extent of the interactions in the stimulus.

Under the assumptions described above, the posterior distribution for an observed population spike train can be derived as a gaussian distribution with a mean $\mu(T)$ that is a weighted sum of the preferred positions of neurons that fired and a variance $v^2(T)$ that depends on only $\mathcal{C}$ and the tuning width σ^2,

$$\mu(T) = \mathbf{k}(\boldsymbol{\xi}_\varsigma, T) \cdot \boldsymbol{\theta}(T) \tag{4.6}$$

$$v^2(T) = \mathcal{C}_{TT} - \mathbf{k}(\boldsymbol{\xi}_\varsigma, T) \cdot \mathcal{C}_{\varsigma T}, \tag{4.7}$$

where $\mathcal{C}_{TT} = c$ is the static stimulus variance at the observation time and $\mathcal{C}_{\varsigma T}$ is a vector of the cross-covariance between the spike times and observation time T. The weight on each spike depends strongly on the time at which the spike occurred:

$$\mathbf{k}(\boldsymbol{\xi}_\varsigma, T) = \mathcal{C}_{T\varsigma}(\mathcal{C}_{\varsigma\varsigma} + \mathbf{I}\sigma^2)^{-1}. \tag{4.8}$$

Here, $\mathcal{C}_{\varsigma\varsigma}$ is the covariance matrix of the stimulus at all the spike times.

4.3 Decoding Kernels. In this letter, we focus on generating a new set of output spikes $\overrightarrow{\boldsymbol{\rho}_T}$ by keeping the decoding kernels fixed and adapting just the network's synaptic weights, such that decoding the output spikes with the fixed kernels can approximate the optimal distribution. In a realistic network architecture, both the kernels and the weights can be adapted, since the kernels and the weights together determine the population responses. While adapting the weights at one level has an effect on subsequent representation of the input, adapting the kernels has an effect on the interpretation of responses at that level. Then any change to the kernels at one level affects proper spike interpretation at the next, since it entails possibly complicated coordination of kernels across levels. Therefore, in this letter, we concentrate exclusively on fixed kernels.

However, there is flexibility in the choice of these decoding kernels. In section 2.2.2, we described simple separable kernels defined by only three parameters: the preferred stimulus value s_j of an output neuron j; the projective width ω, which is the same for all the neurons; and a temporal integration constant γ. It is possible to have more complex kernels that match the specific spatiotemporal dynamics of the stimulus trajectories. In Huys et al. (2007), we explored one such case: we directly inferred a set of kernels to match the spatiotemporal dynamics that governed the smooth stimulus trajectories. We refer to these as *optimized kernels* $\kappa_{\text{opt}}(i, s, t)$.

The kernels were divided into discrete time and space intervals, defined as a discrete array. The values for the bins in this nonparametric kernel representation were inferred for the full input spike train by minimizing the KL divergence between the optimal distribution and an approximate distribution derived by independent decoding with these kernels:

$$\kappa_{\text{opt}}(i,s,t) \leftarrow \kappa_{\text{opt}}(i,s,t) + \epsilon \Delta_{\kappa_{\text{opt}}(i,s,t)} D_{KL}(p(s_T|\overrightarrow{\boldsymbol{\xi}_T})||\hat{p}(s_T|\overrightarrow{\boldsymbol{\xi}_T})). \tag{4.9}$$

The gradient has the following form:

$$\Delta_{\kappa_{\text{opt}}(i,s,t)} D_{KL}(p(s_T|\overrightarrow{\boldsymbol{\xi}_T})||\hat{p}(s_T|\overrightarrow{\boldsymbol{\xi}_T})) = \sum_T [\hat{p}(s_T|\overrightarrow{\boldsymbol{\xi}_T}) - p(s_T|\overrightarrow{\boldsymbol{\xi}_T})]\xi^i_{(T-t)}. \tag{4.10}$$

Learning was carried out by generating stimulus trajectories and corresponding spike trains $\overrightarrow{\boldsymbol{\xi}_T}$, and continued until the update equations converged. Although these kernels were optimized to fit the stimulus dynamics, once learned, they were held fixed during decoding (in lieu of $\kappa_{\text{std}}(j,s,\tau)$ in equation 2.8).

5 Simulations and Results

In this section we examine the extent to which recoding can obviate the need to maintain a spiking history for smooth stimuli and whether simple independent decoding of the recoded spikes can be near optimal. All the illustrations presenting the results below have the same format: true stimulus trajectory in a dashed line and population response in white circles (with the output population spikes in panel A and input spikes in B). As noted earlier, the ordinate represents the one-dimensional stimulus space and the abscissa, time. Each neuron (say, an input neuron i) has a preferred stimulus value (θ_i); if it emits a spike at time t, a shaded circle is drawn at position $(t;\theta_i)$. The preferred stimulus positions of the neurons span the stimulus space in a regular manner, and the neurons are ordered by their preferred value. The shaded circles in the following figures represent the spiking activity of the entire population of neurons over time. The approximate independently decoded distribution is shaded in gray scale in panel A of all the comparison plots and the optimal distribution (ideal observer) in panel B unless otherwise noted; the approximate and optimal posterior mean are in solid lines in the respective panels.

5.1 Stimulus Trajectories and Network Parameters. Stimulus trajectories $\overrightarrow{s_T}$ are generated by sampling from the gaussian process prior distribution defined in equation 4.5, and representing the spikes in a discrete time format. The parameter ζ of the covariance matrix $\mathcal{C}$ determines the

smoothness of the trajectories. By changing the value of ζ, we can consider different classes of GP priors, each defining the density over a set of trajectories with similar spatiotemporal dynamics. In our simulations, we test the efficacy of the network implementation of the coding scheme in recovering the optimal posterior for various classes of dynamics, but we focus primarily on ecologically relevant smoothly varying stimuli where the trajectories are non-Markovian. The parameter values used in the simulation are $\zeta = 2$, $\alpha = 0.05$, $c = 0.2$.

The neural population $i = \{1, \ldots, N\}$ is defined such that the preferred stimulus positions $\boldsymbol{\theta}$ of the neurons are evenly distributed in the stimulus space. We concentrate on a sparse spiking regime in our simulations since input sparsity makes a more challenging case for recoding. It allows us to evaluate empirically when and how the recoding population relies on the prior information about stimulus dynamics to decode effectively. Sparse inputs are generated in the simulations by assigning a low value to the maximum input firing rate $r_{\max} = 0.144$ and tuning width $\sigma = 0.1$ in equation 4.1. The preferred stimulus positions of the recoding recurrent population $j = \{1, 2, \ldots, M\}$ are also distributed evenly in the stimulus space; projective width $\omega = 0.2$. The temporal integration constant γ of the decoding kernel (see equation 2.12) tracks the value of the learned parameter β (see equation 2.4) which specifies the dynamics of the spiking activity. The input and recurrent output populations connect by feedforward $\mathbf{W}$ and lateral synapses $\mathbf{U}$ to form a fully connected network. In the simulations that follow, $N = 100$, $M = 100$.

Training time is a function of the number of input and output neurons and initialization strategy used for the weight matrices $\mathbf{W}$ and $\mathbf{U}$. The weights could be initialized randomly; however, initializing the weights based on a deterministic formulation of the network (detailed in Zemel, Huys, Natarajan, & Dayan, 2005) resulted in faster convergence. For each set of simulations, a network is trained on an average of 500 stimuli of length 200 each (discrete time bins), all generated from a gaussian process with the same parameters. Learning is stopped once the gradient magnitudes drop below a prespecified threshold $= 0.25$.

5.2 Decoding Smooth Trajectories. Figure 3 presents the results from decoding a particular smooth stimulus; the approximate distribution inferred by log-linear decoding on the recoded spikes (see Figure 3A) is compared with the optimal distribution (see Figure 3B). Even in this sparse input spike regime, the network is able to approximate the posterior distribution over stimulus trajectory with high fidelity, in this particular example with $I_L = 0.061$. The gray arrows in Figure 3A point to two interesting regions with respect to the dynamics of the posterior mean. In the absence of spikes, the mean continues in the general direction of the stimulus as observed from previous spiking activity. The decoded distribution also accurately represents the uncertainty in the input. This

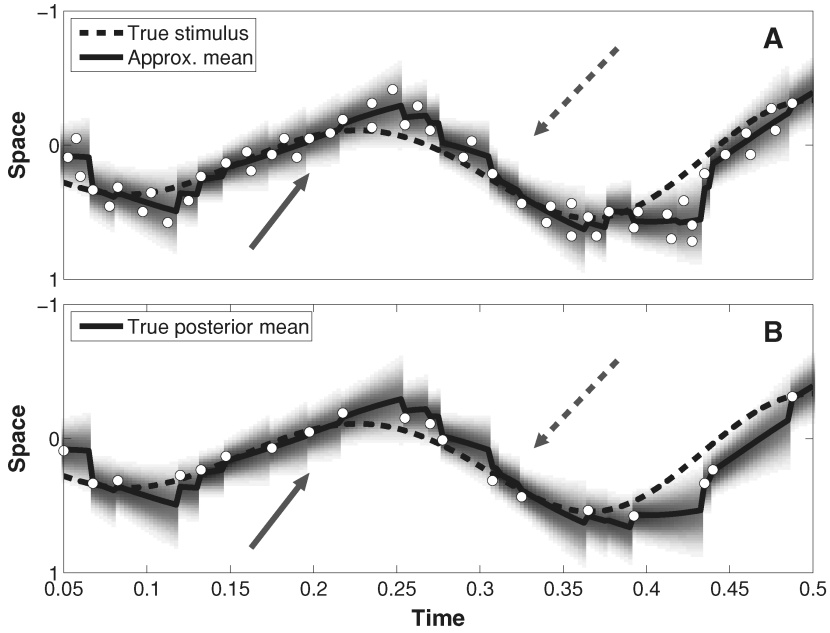

Figure 3: Recoding a smooth stimulus, example 1. (A) Log-linear decoding on network recoded spikes. (B) Decoding by ideal observer on input spikes. Gray arrows point to two example instances showing that in the absence of spikes, the recurrent network has captured the appropriate spatiotemporal regularities corresponding to this class of smooth stimuli, as evident from the effective approximation of the optimal distribution. The mean of the approximate distribution is also a good fit to the smooth trajectory.

is demonstrated by the fact that the variance of the decoded distribution very closely approximates that of the optimal distribution (see Figure 3B).

We show another example of decoding a smooth trajectory in Figure 4. Here again, the network produces a recoded representation such that the resulting spikes can be considered independently even if the information in the input was encoded in combinations of spikes. In the absence of evidence from the inputs, the network relies on the learned prior to produce a pattern of activity such that the approximate posterior distribution is a good estimate of the optimal distribution with $I_L = 0.072$. The posterior mean also matches the optimal mean quite well. The network correctly weighs the history and the new input according to their associated uncertainties. This is evident, as pointed to by the gray arrow, in how the posterior mean suddenly shifts to the position indicated by the input spike.

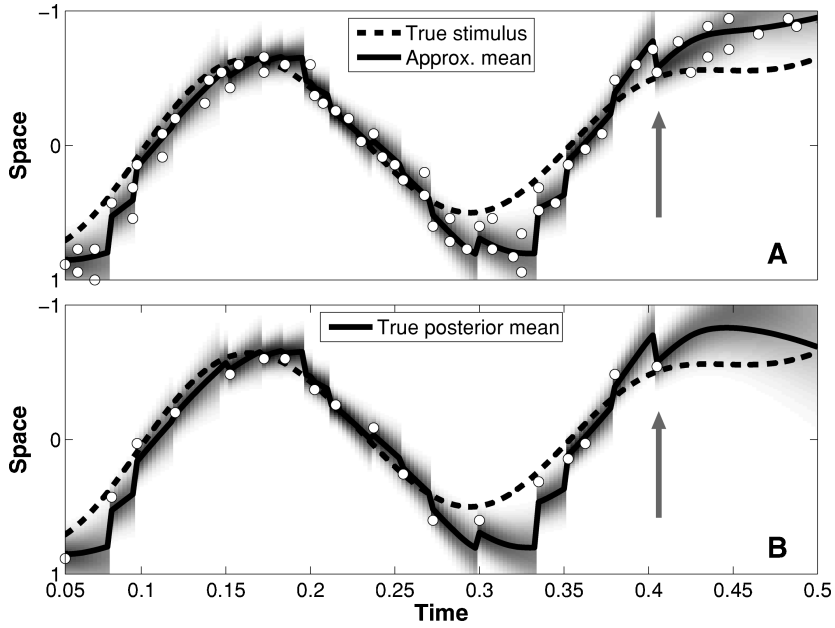

Figure 4: Recoding a smooth stimulus, example 2. (A) Log-linear decoding on network recoded spikes. (B) Decoding by ideal observer on input spikes. When a new input spike arrives, the posterior mean suddenly shifts to the new position indicated by the spike, suggesting that the network correctly weighs the history and the new evidence according to their associated uncertainties. The gray arrow points to one example of this behavior.

5.3 Understanding the Recoding Mechanism. We can gain some insight into how the network achieves these results by examining its learned feedforward and lateral weights, temporal decay constant, and the decoding kernels. The learned feedforward connection strengths **W** are illustrated in Figure 5A in the form of a matrix where the rows $i = \{1, 2, \ldots, N\}$ and columns $j = \{1, 2, \ldots, M\}$ correspond to the input and output neurons, respectively. Each cell of the weight matrix is shaded according to the strength of the synapse W_{ij}. These weights show that the network has learned a strong local connectivity, with each input neuron having strong excitatory connections to neurons in the recurrent population having the same or very similar stimulus preference and inhibitory connections to their neighboring neurons.

The learned lateral connection strengths **U** are illustrated in Figure 5B in a similar matrix form with rows $j = \{1, 2, \ldots, M\}$ and columns $k = \{1, 2, \ldots, M\}$ corresponding to the output neurons spatially laid out in the stimulus space. Each cell of the weight matrix **U** is shaded according to

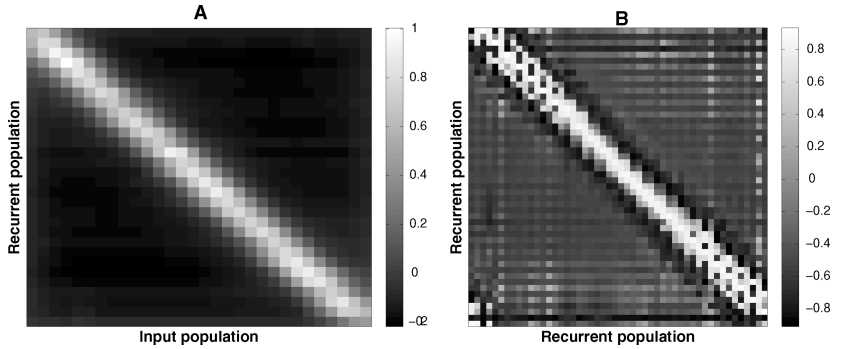

Figure 5: Smooth stimulus: understanding the synaptic weights. (A) Feedforward weights from neurons of the input population to output neurons of the recurrent population. (B) Lateral weights of 50 neurons of the recurrent population involved in recoding its input representation of a smooth stimulus trajectory.

the strength of the synapse U_{jk}; each neuron is seen to strongly excite its immediate neighbors and also inhibit the ones farther away. The negative weights allow one neuron's spikes to decrease the membrane potential of another. So the population activity not only indicates where the stimulus is at any given time (by spiking when the stimulus is at the neuron's preferred position); it also actively indicates where the stimulus may not be.

Here the learned temporal decay constant $\beta = 2.13$ (see equation 2.5). This allows the postsynaptic potential of neurons in the recoding population to decay slowly with time; each spike will have half its potential after about 30 time-steps. This means that temporally distant spikes have very slowly diminishing effects on the current population activity. The decay constant reflects the temporal extent of the interactions in the stimulus, specified by α in equation 4.5. This same rate of decay is applied to γ in the temporal kernel (see equation 2.12) to decode the spikes; it implies that the influence of a spike persists for a while, having a slowly decaying effect on the representation at future times.

The standard spatial and temporal decoding kernels used in this simulation are shown in Figure 6A; they are convolved together in the figure for the purpose of illustration. Only the kernel centered at the mean of the state-space is shown here; kernels for all the other neurons are alike, centered at their respective preferred stimulus positions. They have a very simple structure that is only exponentially decaying over time, and they are uniform across all neurons.

Since the lateral weights and the decay constant are learned together in an iterative manner with the decoding kernels fixed, they influence each other very closely. Combined with the input spikes $\boldsymbol{\xi}_t$, they influence the

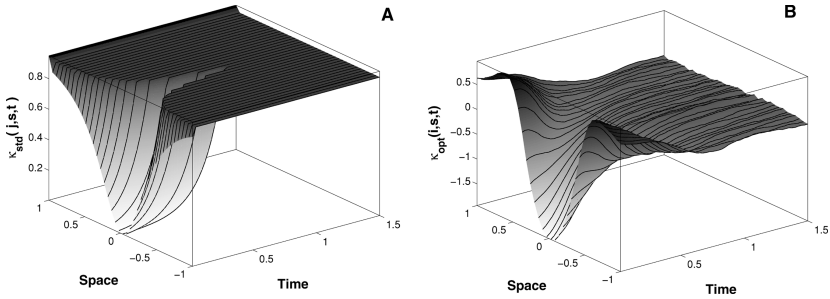

Figure 6: Smooth stimulus: Understanding the decoding kernels. (A) Simple spatial and temporal decoding kernels for a neuron j with a preferred value $s_j = 0$, convolved here for the purpose of illustration. A spike in this neuron at time t adds negative log probability to s values in proportion to their distance from $s = 0$ (see equation 2.8), making the estimated posterior more peaked around 0 at that time (see equation 2.7). This contribution decays over time. (B) Adapted spatiotemporal kernel inferred for smooth trajectories for a neuron with preferred stimulus value $s_i = 0$. The kernels have the shape of difference of gaussians for $t = 0$, falling off exponentially with time.

membrane potential $\boldsymbol{h}_t$ (see equation 2.5) of neurons in the recurrently connected population according to how the input population connects to the recurrent population via the feedforward weights $\mathbf{W}$.

In the absence of input spikes, recurrent activity contributes to the persistent effect of previously observed spikes. This triggers a cascaded pattern of activation among neurons of the recurrent population. As a result, the approximate posterior distribution is a faithful estimate of the optimal distribution. This is evident in Figure 3, especially in the region indicated by the gray arrows.

So what is it about the recoded representation that affects simple readout of a distributional code that the input representation is unable to achieve? One possible explanation is that the recoded spikes are first-order Markov, that is, recoding produces temporally independent spikes and the decoding kernels convolve them in such a way that information originally encoded in temporal spike patterns can now be independently decoded to produce a smooth, inferred distribution.

5.4 Decoding Input Spikes Versus Recoded Spikes. The results in section 5.2 illustrate that independent decoding of the recoded representation (see Figure 3A) can closely match the ideal observer (see Figure 3B). However, this alone does not show that recoding the input spikes is necessary for smooth trajectory dynamics. An alternative approach is to find decoding kernels directly for the input spikes such that the estimated posterior closely approximates the optimal distribution. To evaluate this, we use

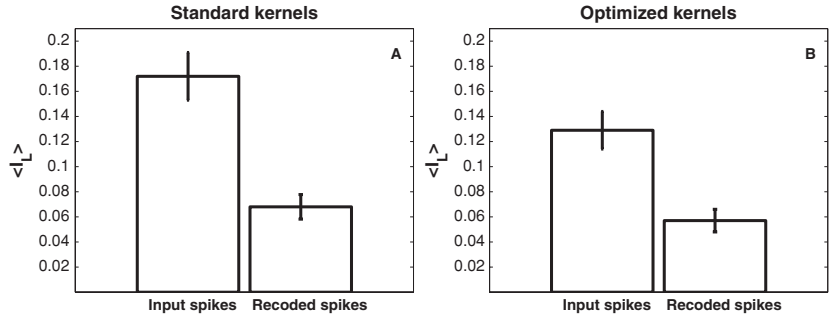

Figure 7: Cumulative statistics from testing the recurrent network model on 250 different smooth trajectories. $\langle I_L \rangle$ indicates the information loss averaged over all the test cases. The effect of using standard parameterized kernels (A) versus optimized nonparameterized kernels (B) is compared on the population inputs and the recoded spikes. A proper combination of embedding the prior in the decoding kernels as well in the lateral weights yields the lowest I_L with an optimally decoded distribution. The error bars provide an indication of the uncertainty in the information loss estimates.

the optimized kernels (see section 4.3) to decode the input spike trains separately. We chose these particular kernels since unlike the standard kernels, they were inferred to match the smooth dynamics of the trajectories.

An example optimized kernel is illustrated in Figure 6B for a neuron with preferred stimulus at the center of the state-space. These kernels specify prior-dependent dynamics for the posterior distribution and take on a complex shape for the smooth trajectories, nonseparable in space and time. For $t = 0$, the kernels have the shape of difference of gaussians. As the time from the observation of the last spike increases, the shape of the kernel changes slightly.

We decoded the same input spikes shown in Figure 3B with these optimized kernels and obtained an information loss value of $I_L = 0.129$. By comparison, we reported in section 5.2 that our recoding procedure (and even using nonoptimized kernels) achieved a much lower information loss $I_L = 0.061$. Recoding is important for smooth trajectories, since there is no Markov property to license a simple representation of the whole spiking history. It is not possible to reverse the loss of information using fixed kernels, even ones that had been adapted to the correct trajectories.

5.5 Cumulative Results. We now analyze and compare the cumulative statistics derived from averaging over 250 test cases of decoding for smooth stimuli, directly from both the input spikes and the recoded spikes; the averaged I_L values compared to the ideal observer are presented in Figure 7. In all these tests, recoding with a particular choice of kernels meant that

the network parameters were learned using a reward signal based on a comparison of the ideal observer and decoded output spikes using those kernels. Figure 7A shows that recoding with standard kernels and decoding the resulting spikes far outperforms direct decoding of input spikes with the same kernels.

In a second set of simulations, we examine the effect of the kernel on our recoding and decoding results. We compare decoding of input spikes versus recoded spikes using a different choice of decoding kernel, the optimized kernels, in Figure 7B. Even with such kernels adapted to the stimulus dynamics, recoding improves the fidelity of independent decoding compared to decoding the input spikes directly.

A third interesting result involves comparing Figures 7A and 7B. Recoding with the standard kernels in Figure 7A lends a much better approximation to the optimal distribution than decoding with even optimized kernels on the input spikes in Figure 7B. This result further supports the claim that despite information being encoded in combinations of spikes, recoding is capable of producing spikes that can be interpreted independently downstream using simply specified kernels.

The fourth and final result here concerns comparing recoding with the two different kernels. A natural approach would be to use apt kernels adapted to the trajectories and recode. Comparing the I_L for this kind of recoding in Figure 7B with all others, we see that this strategy leads to some improvement over either form of adaptation alone. The difference between decoding using optimized kernels of the recoded spikes versus input spikes (see figure 7B) is statistically significant with a p-value of 0.01. The difference between decoding using optimized versus standard kernels of the recoded spikes (second bar in Figures 7A and 7B) is also statistically significant with a p-value of 0.01. Thus, regardless of the form of decoding kernels used, recoding significantly improves the fidelity of independent decoding of smooth trajectories, a particularly difficult case for decoding.

5.6 Recoding with the Wrong Prior. To investigate further the extent to which the network learns dynamics from an ensemble of distributions over stimulus trajectories, the input spike train for another smooth trajectory was recoded by a network whose parameters were adapted to different, random walk trajectories. Figure 8 illustrates some sample trajectories with first-order Markov dynamics drawn from the prior distribution in equation 4.5 with $\zeta = 1$. These trajectories are characterized by a slow drift to the prior mean ($m = 0$ here for simplicity) of the state-space.

The feedforward weights (see Figure 9A) adapted to these dynamics are very similar to those learned for smooth trajectories and simply relay information to the output neurons. The learned lateral weights are shown in Figure 9B. Compared with the weight matrix for the smooth case in Figure 5B, the lateral connection pattern for the random walks is significantly different. Each neuron has strong positive connections to

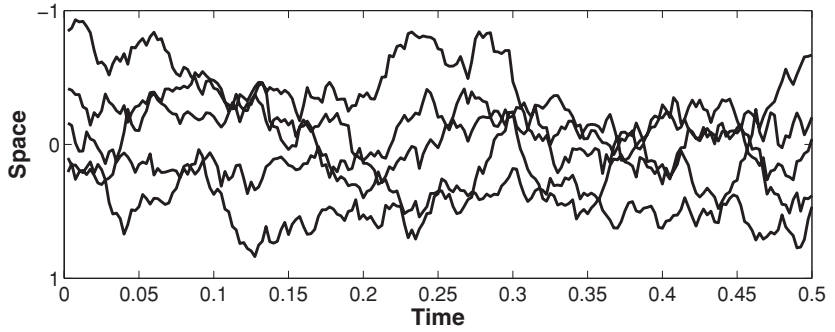

Figure 8: Random walk stimuli. Sample trajectories with first-order Markov dynamics, characterized by a slow drift to mean $m = 0$.

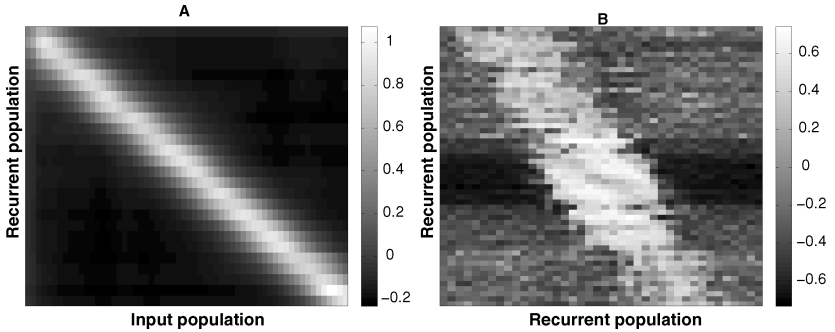

Figure 9: Network weights adapted to random walks. (A) Feedforward weights from 50 input neurons to output neurons of the recurrent population. (B) Lateral connections between 50 neurons learned for an ensemble of random walk stimuli. Contrast with the lateral connections from Figure 5B.

some (but not its immediate) neighbors, evident in the positive weights shaded white in the matrix. However, the learned temporal decay constant $\beta = 0.85$ is much lower than that of smooth dynamics.

Figure 10 compares the posterior distribution when this mismatched network is used for recoding a smooth trajectory (panel A) with the optimal distribution (panel B). The result shows that the decoding is suboptimal (especially around the regions indicated by the gray arrows) since in the absence of spikes, the distribution decays to the local mean right away instead of predicting that the stimulus will continue in its trend first before drifting back to the mean. The approximate posterior mean provides a poor match to the optimal posterior mean (panel B). Information loss from decoding with the wrong prior is high, with $I_L = 0.13$.

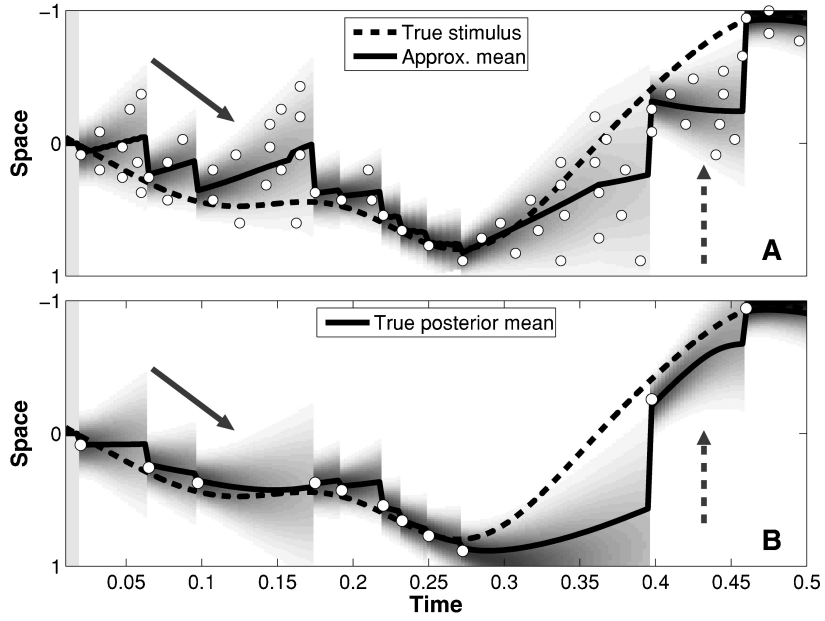

Figure 10: Recoding with the wrong prior. (A): Recoding a stimulus with a network adapted to different dynamics. In this case, the trajectory is smooth, but the network is trained on random walks. The decoding is suboptimal compared with the ideal observer in B, indicating that prior information about the dynamics is indeed embedded in learned network parameters.

When the value of β is substituted in equation 2.4, the postsynaptic potential of neurons in the recoding population decays much more rapidly with time, which does not match the slow variation in the smooth stimulus. Temporally distant spikes have very little effect on the current population activity. Since the lateral weights and the decay constant were adapted together to the random walk dynamics in an iterative manner, they interact to allow the recurrent population to forget quickly. Therefore, we do not observe the persistent effect that was characteristic of the network recoding for smooth stimuli. These decoding results strongly suggest that the prior information about spatiotemporal regularities is embedded in the learned network parameters, allowing linear mechanisms for efficient readout of a distributional code.

5.7 Effect of Tuning Curve Manipulations. We examine the effect of manipulations of the input population tuning curves on the fidelity of the decoded distribution. Theoretical studies have concluded that the

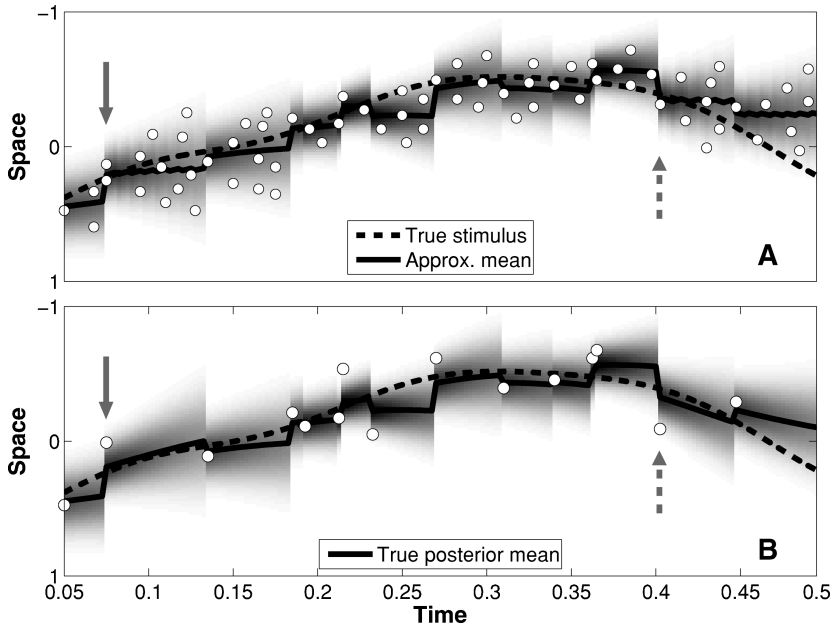

Figure 11: Effect of manipulations of input tuning curves. (A) Recoding input spikes generated using broader tuning curves having a higher $r_{\max}$ than earlier simulations. (B) Decoding by ideal observer on input spikes. Even when input spikes carry imprecise information on the stimulus position due to wider tuning curves, the network learns to correctly weigh the prediction given the past spikes and the information from the current spikes in accordance with their uncertainties. Solid and dashed gray arrows show evidence for this in the sloth with which new evidence leads to changes in the distribution.

optimal values for tuning width σ and gain $r_{\max}$ for a population code (see equation 4.1) depend critically on the covariance of noise in the input (Pouget, Deneve, Ducom, & Latham, 1999). Given that we have chosen to ignore noise correlations in this scheme, we are not concerned with the effects of tuning widths on input representations $\overrightarrow{\boldsymbol{\xi}_T}$ in this architecture. Instead we examine whether and how the learned network parameters and the recoded representation $\overrightarrow{\boldsymbol{\rho}_T}$ are sensitive to the different tuning widths.

We consider a case in which the the maximum input firing rate $r_{\max} = 0.25$ as well as the tuning width $\sigma = 0.25$ in equation 4.1 take on values higher than those examined in the simulations so far. The individual input spikes (solid white circles in Figure 11B) provide imprecise information on the stimulus position. But the posterior distribution decoded using our

scheme (see Figure 11A) is still a good approximation to the optimal distribution in Figure 11B from the ideal observer. The estimated posterior mean and variance closely match that of the ideal observer, and the information loss in this experiment is $I_L = 0.072$, similar to the previous cases using sharper tuning curves.

The learned network weights indicate that the recoding population is indeed sensitive to the differences in tuning curve parameters; the pattern of weights is similar to that in Figure 5 but much broader across the diagonal. The lateral weights have stronger positive and negative connections. Much like the case of narrow tuning curves in Figure 4, here too the network correctly weighs the information from past spikes and the new evidence according to their associated uncertainties; the solid and dashed gray arrows in the figure point to two such instances. The network learns stronger lateral weights than the feedforward weights, which indicates that it assigns higher weight to the prior information than the likelihood from input spikes in this condition. Also notable is that the variance of the decoded distributions is generally higher with wider input tuning curves. Overall, this result suggests that the network can be trained on inputs from neurons having a variety of tuning parameters and still learn to accurately capture underlying spatiotemporal dynamics.

5.8 Limitations of the Network. We first refer to the result in Figure 12A. In the absence of input spikes starting from the time indicated by the gray arrow, the network relies on the prior information about trajectories learned in the temporal decay constant and the lateral weights. The variance of the approximate distribution correctly grows with time until a new spike is encountered. However, the approximate posterior mean during this time is a poor fit to the posterior mean of the optimal distribution (see Figure 12B), which gradually moves away from the center of the state-space before reversing back. This limitation is largely attributable to the simple dynamics and recurrence in the network, which allows only a coarse account of population spike history to be retained. Although the network seems not to capture subtle changes in the optimal posterior mean, the approximate mean is in fact still within the optimal uncertainty cloud.

The second limitation has to do with the density of recoded spikes. The output firing rate is much higher than the inputs, especially during periods of limited input, when encoding a high-variance distribution. Since the network parameters are learned with the standard decoding kernels, this could be a consequence of the simplicity of the decoder, stemming from the interaction with the magnitude of the kernels. With the ideal observer, the contribution of each spike (i.e., the effective number of bits of each spike) is higher because the decoder is complex; hence, the same information could be communicated with fewer spikes.

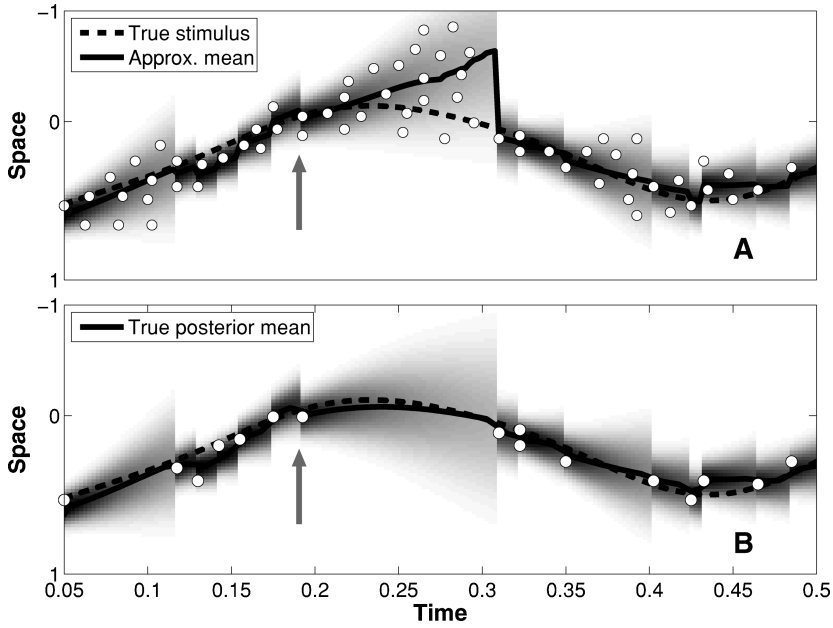

Figure 12: Network limitations. In some cases, the network is unable to capture subtle changes in the stimulus dynamics. Encoding produces dense output spikes to code a high-variance distribution.

Our cumulative results in section 5.5 suggested that learning the network parameters using the optimized kernels instead of standard kernels gave a slight improvement in the I_L values. Hence, we evaluated the efficacy of this strategy in encoding input distributions with high variance. We present the result in Figure 13, recoding the same trajectory and input spikes as those in Figure 12.

The variance of the optimal distribution, as seen in Figure 12B, grows steadily during the times when there are no input spikes. The variance of the approximate distribution in Figure 13 matches this closely, and the reconstructed posterior mean is also a good fit. More important, the recoded spikes coding the high-variance distribution in Figure 13 are not as dense as those observed in Figure 12A, meaning that each spike conveys more bits. The information loss here is $I_L = 0.056$.

These results further suggest that searching in the space of decoding kernels to find an appropriate set might improve the recoding efficacy. One option is to allow varieties of spatial kernels, perhaps through adaptation of the kernels, so that a single spike in a neuron with a broad projective width could directly encode a high-variance posterior.

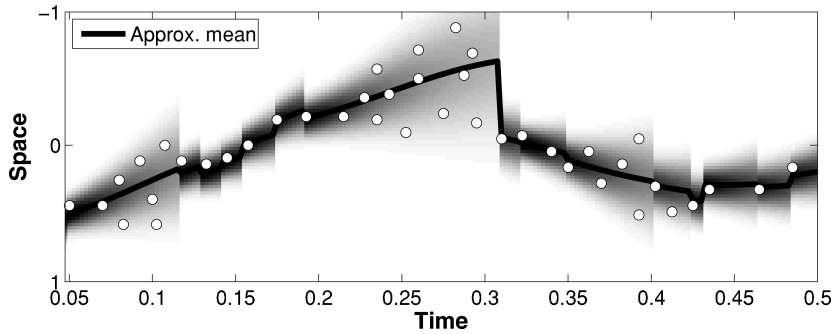

Figure 13: Recoding with optimized kernels. The input spikes for trajectory in Figure 12 are recoded here using a network, the parameters of which were learned by decoding using optimized kernels. The network produces output spikes that are less dense than using standard kernels. Adapting both the network weights and the decoding kernels may be a better strategy for recoding.

6 Summary and Discussion

We have studied the spiking population code representation of transient stimulus variables whose values follow a trajectory through time. Earlier studies have shown that neurons in a variety of sensory systems can faithfully encode information about stimuli that vary on ecologically relevant timescales (approximately 30 ms) with high efficiency (Vickers et al., 2001; Dean, Harper, & McAlpine, 2005). Maintaining an accurate representation of rapidly varying stimuli provides a difficult computational challenge for spiking population codes. In our model, the network's output spikes convey information associated not only with the mean of the stimulus trajectory but also with the full posterior distribution, and in particular the variance reflecting the uncertainty arising from noise in sensation.

Accurate perception requires this dynamically changing uncertainty to be represented, manipulated, and learned about correctly. Thus, the main biological motivation for our work is to investigate how populations of neurons can code stable representations of the stimulus value along with the uncertainty, allowing consistent semantics of inputs and outputs at every level of the neural hierarchy. As we show using a simple two-layer recurrent network in our simulations, this can be achieved by employing a fixed decoder (i.e., using kernels that are fixed at the output) and adapting the network parameters to recode the inputs into a stable representation that can be faithfully interpreted under the specified decoding scheme.

This form of temporal recoding can be considered a natural extension of the line attractor scheme of Pouget, Zhang, Deneve, and Latham (1998). Both methods formulate feedforward and recurrent connections so that a simple

decoding of the output can match optimal but complex decoding applied to the inputs. The model presented here extends this approach to spiking networks and, more important, to dynamically varying inputs associated with a prior that must be learned. The latter part of our discussion below outlines some potential ways of applying our model to experiments on real neural systems.

6.1 Summary. The highlights of our proposal as to how neural populations might encode continuous dynamic stimulus variables are as follows. We first specified a biologically motivated decoder for downstream neural access to information encoded in the population inputs. The decoder did not require any difficult assumptions about neuronal selectivity, the distribution, or tuning of neural reponses. It is based only on the idea that downstream neurons must be able to interpret the afferent inputs in a simple and fast manner; to this end, the decoder ignored all correlations in the input and treated each spike independently.

Second, we described how this independent treatment of input spikes during decoding is an unfaithful model for naturalistic stimuli that have smooth spatiotemporal dynamics. In Huys et al. (2007), we showed that under such ecologically relevant dynamics, the input spike history needs to be maintained in memory for accurate linear decoding. In this letter, we suggested that a neural population, rather than employing different decoding strategies for different stimulus dynamics, can recode the input representation into one that can be decoded independently regardless of stimulus dynamics. This recoding allows simple and faithful access to the encoded information and thus may facilitate downstream processing.

Third, we proposed a neurally plausible implementation of the recoding scheme using a simple nonlinear recurrent network that could learn the spatiotemporal regularities in stimulus dynamics using a local learning rule. The learning rule was based on the correspondence between the inferred distribution over stimulus values and the Bayes-optimal distribution encoded in the neural responses, under one of the conventional spike generation models. Finally, we illustrated using many simulations with smooth stimulus trajectories that the efficacy of the neural implementation of our proposed scheme closely matches that of optimal decoding.

6.2 Related Work. Several aspects of our framework bear interesting relationships to earlier work. First, our independent decoding scheme can be compared to other proposals for decoding trajectories from neural population responses. Bayesian methods have been employed previously to infer the 2D location of a rat from hippocampal place-cell activity (Brown et al., 1998; Twum-Danso & Brockett, 2001). These methods adopt gaussian approximations to a nonlinear, recursive Bayesian decoding scheme and provide a framework for a sequence of predictions needed for dynamically evolving behavioral applications such as neural prosthetics.

The application of Bayesian decoding to motor cortical data was proposed in Gao, Black, Bienenstock, Shoham, and Donoghue (2002) with various Kalman filter formulations. While these formulations are valid for stimulus trajectories with simple first-order Markovian dynamics (the generative model underlying Kalman filters), it may not be appropriate for stimuli with smooth autocorrelations. Our results in section 5.6 indicate qualitatively what information is lost by applying Kalman-filter-like formulations to decoding smooth stimuli.

Another approach extends the Kalman filter to reformulate the spike generation model (likelihood) as a probabilistic mixture of linear gaussian models and uses a switching Kalman filter for decoding with the mixture model (Wu et al., 2004). This approach naturally generalizes the posterior distribution since it can produce multimodal posteriors but requires a specification of the number of components in the mixture.

Other studies have used particle filtering to solve the recursive Bayesian decoding task with nonlinear, nongaussian likelihoods (Gao et al., 2002; Brockwell, Rojas, & Kass, 2004). Brockwell et al. (2004), for example, constructed a statistical coding model for monkey hand movements during ellipse-tracing experiments. They used adapted parametric tuning functions with nonuniform density to model directional tuning in motor cortex. Although the experiments generated trajectories with smoothly varying acceleration, the movement trajectories were modeled as a first-order Markov process in which states corresponded to velocity values, and the states were constrained to evolve smoothly from one time-step to the next. Optimal Bayesian decoding was approximated with particle filters. The formulation also assumed that motor cortex codes movement in terms of firing rates. By contrast, our approach employed a simple model of neural responses that nonetheless handled spikes rather than firing rates and a more complicated trajectory model, directly representing the smoothness constraint with a gaussian process. We then approximated optimal Bayesian decoding with a recurrent neural network and a simple log-linear spike decoder.

The choice of decoder applied to the output spikes is crucial, especially for ecologically relevant, smoothly varying trajectories. If we allow a complex decoder, for example, the Bayesian decoder discussed in section 4.2, then the network needs access to all the input spikes. To prevent infinite regress, the decoder applied to the output spikes should access the type of information that biological neurons might reasonably be expected to extract. In our case, this is particularly a decoder that does not require access to all spikes at all times. We chose an independent decoder that integrates the information provided by spikes recursively, which is likely the way real neurons integrate information and has the advantage of being well understood in its own right.

The second highlight of our work, the recoding objective, can be considered in the light of earlier attempts to formulate the accuracy of simple decoding as an objective for population responses. Pouget et al.

(1998) proposed that the processing in a recurrently connected population can be viewed as facilitating the ability of a linear decoder to faithfully match maximum likelihood decoding of the population inputs. Similarly, Wu, Nakahara, and Amari (2001) examined the efficiency of an unfaithful decoder that neglected pair-wise correlations between neuronal activities against that of an optimal maximum-likelihood inference. Their results showed that the simple decoder reduced computational complexity remarkably, maintained high decoding accuracy and efficiency in the asymptotic range when neuronal correlation is uniform, and was biologically implementable in a recurrent network. Our approach extends this same objective along two dimensions: to apply to spiking networks and dynamic input variables.

A third line of related work concerns other attempts to learn spiking networks that can faithfully represent dynamic stimuli. There are relatively few proposals specifying how the strength of synaptic connections within and between spiking populations can be learned to carry out difficult information processing tasks, particularly those that change on the fast timescales that make timing important. Notable examples such as Smith and Lewicki (2005) and Hinton and Brown (2000), develop methods for learning representations of complex signals in a population of spiking neurons. These proposals, however, lack a method of performing online inference; that is, the mapping from the input signal to the spikes is not causal. Instead of adapting the kernels to the stimulus statistics, the recoding framework adapts the synaptic weights to re-represent the information in inputs into spike trains that can be easily interpreted downstream using a fixed set of kernels; causality of decoding is maintained throughout.

6.3 Limitations. Our proposal has several relevant limitations. First, the system was tested using a very constrained input spiking scenario. The population inputs are produced by a Poisson-gaussian spike generation model, where the model neurons exhibit only roughly realistic smooth unimodal tuning curves, each having the same shape and asymptoting at 0, differing only in their centers (preferred stimulus values) being separated by a fixed distance in the stimulus state-space. While several authors (Pouget et al., 1998; Zhang & Sejnowski, 1999; Seung & Sompolinsky, 1993) have made the same assumptions, various issues about the model are actively debated. The associated criticism stems from our use of the work of Huys et al. (2007) as providing a suitable test case for which a close approximation to the true posterior distribution is straightforward to calculate.

Our proposed framework can readily generalize to other spike generation processes with more realistic tuning curves. Our characterization of the distribution of the neurons can be seen as one particular case of a more general functional description by Twum-Danso and Brockett (2001), where the geometry of hippocampal place cell distribution is parameterized such that the space of neural preferences is a function of the higher-dimensional

stimulus state-space and the spacing between the neurons can be adjusted. Such nonhomogeneity in tuning curves might improve the accuracy of input representation. One consequence of this generalization would be that the variance of the optimal posterior distribution in our scheme would then depend not just on the spike timing, as it does now, but also on the relative tuning preferences of the spiking neurons.

Another assumption is that the input spikes are independent across time and neurons, with no dependence on stimulus history. However, the stimulus-induced correlations that we assumed in the neural activity depend only on the nature of the temporal prior (ζ value in equation 4.5) and not on the particular spiking model assumed; this argument was elaborated in Huys et al. (2007). Thus, the complexity of the decoding task (in having to retain past spikes in memory to account for smooth autocorrelations) primarily derives from the dynamics of the underlying variable, particularly since we focus only on the sparse spiking regime. The dependence on stimulus history could be approximated in some manner similar to the linear-nonlinear-Poisson model neurons (Paninski, 2003). In that case, we suppose that the network would still be useful for accumulating information about past spiking activity on longer timescales.

A second limitation in the model is the fact that the learning procedure is driven by supervision. The network assumes access to a global reward signal: knowledge of the optimal posterior based on the available information in the input spikes. The objective of learning in this setup was to assess whether any set of synaptic weights could be found that could recode the inputs to an independently decodable representation. If the aim extends to making the learning plausible in a biological context, then some feedback about the optimal distribution over stimulus values would be required. While information about the true underlying value can be plausibly derived, it is more challenging to envision how information about the optimal distribution could be obtained.

Finally, we consider the ability of our framework to represent more general distributions in the light of earlier proposals that populations can represent arbitrary probability distributions over stimulus values, and even multiple values (Zemel, Dayan, & Pouget, 1998; Pouget, Dayan, & Zemel, 2003). In all the simulations in this letter, the independent decoder always aimed to form gaussian posterior distributions. This stems from the limitation of the optimal decoder under a gaussian process prior, which can derive only gaussian posterior distributions with a mean that is a weighted average of the preferred values of the spiking neurons. However, our framework can construct general forms of unimodal and multimodal posterior distributions in a causal manner. Yet representing multiplicity is not as straightforward. There is evidence that sensory neurons can encode several environmental stimuli simultaneously in their activity. It is not readily obvious how the model can encode more than one stimulus, allowing for an internal representation of this form of uncertainty as well.

6.4 Current Directions. A factor that potentially complicates recoding concerns the computations carried out by the population as a function of its inputs: the computations can have a considerable effect on the representation of the relevant input variable. However, computational issues are orthogonal to our recoding hypothesis, and our recoding and decoding scheme as well as the results do not address the network's ability to manipulate information in such a way as to achieve a particular computation. While a network will in general face both of these issues (i.e., both issues will have to be solved by any realistic network architecture), we concentrated purely on the issue of information access in this letter. Hence, we have focused on a case where the same input variable is recoded and then decoded, ignoring any kind of computation performed on the input.

We are currently extending the model in two directions. First, we are developing a recursive, hierarchical formulation of the system in which recoded information can be manipulated easily and integrated efficiently in a hierarchical manner to perform neural computations through time. Instead of generating the input spikes using a gaussian-Poisson model as in our current simulations, we apply the recoding scheme and then use the resulting spikes as one of the inputs to another downstream population. This latter population could in turn integrate recoded spikes from this and several other disparate input populations characterized by varying temporal dynamics, each conveying information on different aspects of stimulus. As mentioned at the beginning of this section, the decoding or interpretation of the the spikes would be fixed at each level of the hierarchy.

Second, we are applying the framework to a specific statistical computation, dynamic cue combination, in which information from sensory cues must be mapped from representations specific to each cue to a target representation, such as motor output. Particularly, we consider how a neural system might employ the proposed coding scheme recursively to dynamically reweight and integrate information about the current state from different sensory modalities. For example, one population can encode dynamic information from one modality such as proprioception, while another represents visual information. Both utilize the recoding scheme to efficiently represent a state variable such as position. Then a third population takes these recoded spikes as inputs, combines them, and, using the same representational scheme, produces a spike-based representation of the posterior distribution over current position. The aim is then to relate the model to experimental results, such as those of Körding and Wolpert (2004) on probabilistic computation during sensorimotor processing.

Unfortunately, there are as yet few experimental constraints on our, or indeed other, models of probabilistic representation and processing. However, the stronger claims we are making about the temporal statistics of stimuli and their sensory inputs should admit more powerful experimental tests. Our study makes several predictions that may help initiate such experiments. First, for smooth stimuli, the learned value of β was higher

than for random walk stimuli, which implies that the temporal dynamics of spiking reflect the temporal dynamics of the stimuli themselves. Second, lateral connections play a critical role in encoding the temporal prior over stimulus dynamics, so that spikes during periods of limited input can be read out as making predictions about the underlying stimulus. In particular, during such periods, we expect to observe the way that the increased uncertainty (i.e., posterior variance) is coded. In the hierarchical computation described above, when the inputs to a multisensory population (such as model superior colliculus neurons) from disparate sources (such as visual and somatosensory systems) are combined together, we expect to observe the influence of the sensory cues on the final estimate following dynamic reweighting of each cue based on its reliability. We expect to observe this under various experimental conditions, such as multimodal prior distributions and varying levels of noise both over time and across sensory modalities.

Appendix: Stochastic Gradient Estimation

A.1 Gradient Estimation via Simulation. The probability of generating the spike vector $\boldsymbol{\rho}_t$ at any time t given the population spike vector $\boldsymbol{\rho}_{(t-1)}$ at time $t-1$ and the inputs $\boldsymbol{\xi}_t$ at time t was specified as

$$P(t) \equiv P(\boldsymbol{\rho}_t \mid \overrightarrow{\boldsymbol{\xi}_t}) = P(\boldsymbol{\rho}_t \mid \boldsymbol{h}_{t-1}; \boldsymbol{\rho}_{t-1}; \boldsymbol{\xi}_t) = \prod_j \sigma\left(h_t^j\right)^{\rho_t^j} \left(1 - \sigma\left(h_t^j\right)\right)^{(1-\rho_t^j)}. \tag{A.1}$$

The global reward signal at time t is defined as

$$v(\Omega, t) = D_{KL}\left(p(s_t \mid \overrightarrow{\boldsymbol{\xi}_t}) || q(s_t \mid \overrightarrow{\boldsymbol{\rho}_t})\right). \tag{A.2}$$

The reward directly depends on the parameters Ω. Given an objective function $J(t)$ that maximizes the average reward, the gradient with respect to Ω of the expected reward cannot be computed in closed form since the output spikes are stochastically sampled. We resort to approximate gradient approaches where the gradient is estimated by simulation and the policy is improved by adjusting the parameters in the gradient direction (Baxter & Bartlett, 2001). We employ a version of the GPOMDP algorithm to produce gradient estimates.

The approximate gradient with respect to Ω takes the form

$$\frac{\partial J(t)}{\partial \Omega} = \frac{(t-1)}{t} \frac{\partial J(t-1)}{\partial \Omega} + \frac{1}{t} v(t) z(t), \tag{A.3}$$

where the eligibility trace $z(t)$ that relates the output spikes $\boldsymbol{\rho}_{(0,T]}$ with weights $\mathbf{W}$ and $\mathbf{U}$ and integration constant β, is defined as

$$z(t) = \tau z(t-1) + \frac{1}{P(t)} \frac{\partial P(t)}{\partial \Omega}. \tag{A.4}$$

The second term in the eligibility trace is derived to be

$$\frac{1}{P(t)} \frac{\partial P(t)}{\partial \Omega} = \frac{\partial \log P(t)}{\partial \Omega}, \tag{A.5}$$

where the log probability is

$$\log P(t) = \sum_j \rho_t^j \sigma\left(h_t^j\right) + \left(1 - \rho_t^j\right)\left(1 - \sigma\left(h_t^j\right)\right). \tag{A.6}$$

This derivative of this log with respect to the parameters Ω is found to be

$$\begin{aligned} \frac{\partial \log P(t)}{\partial \Omega} = & \sum_j \rho_t^j \frac{1}{\sigma\left(h_t^j\right)} \sigma\left(h_t^j\right)\left(1 - \sigma\left(h_t^j\right)\right) \frac{\partial h_t^j}{\partial \Omega} \\ & + \left(1 - \rho_t^j\right) \frac{1}{\left(1 - \sigma\left(h_t^j\right)\right)} \left(-\sigma\left(h_t^j\right)\right)\left(1 - \sigma\left(h_t^j\right)\right) \frac{\partial h_t^j}{\partial \Omega}, \end{aligned} \tag{A.7}$$

which can be simplified as

$$\frac{\partial \log P(t)}{\partial \Omega} = \sum_j \frac{\partial h_t^j}{\partial \Omega} \left(\rho_t^j - \sigma\left(h_t^j\right)\right). \tag{A.8}$$

Now, the gradient with respect to each of the parameters in Ω can be derived as shown in the following sections.

A.2 Gradient with Respect to W. The approximate gradient of the objective function with respect to the feed-forward weights $\mathbf{W}$ is

$$\frac{\partial J(t)}{\partial W_{ij}} = \frac{(t-1)}{t} \frac{\partial J(t-1)}{\partial W_{ij}} + \frac{1}{t} v(t) z(t) \tag{A.9}$$

$$z(t) = \tau z(t-1) + \frac{1}{P(t)} \frac{\partial P(t)}{\partial W_{ij}} \tag{A.10}$$

$$\frac{\partial \log P(t)}{\partial W_{ij}} = \sum_j \frac{\partial h_t^j}{\partial W_{ij}} \left(\rho_t^j - \sigma\left(h_t^j\right)\right) \tag{A.11}$$

$$\frac{\partial h_t^j}{\partial W_{ij}} = \xi_t^i + \eta(1) \frac{\partial h_{t-1}^j}{\partial W_{ij}}. \tag{A.12}$$

Plugging these values back into equation A.9 yields

$$\frac{\partial J(t)}{\partial W_{ij}} = \frac{(t-1)}{t}\frac{\partial J(t-1)}{\partial W_{ij}} + \frac{1}{t}v(t) \times \left(\tau z(t-1) + \sum_j \left(\xi_t^i + \eta(1)\frac{\partial h_{t-1}^j}{\partial W_{ij}}\right)\left(\rho_t^j - \sigma\left(h_t^j\right)\right)\right). \quad \text{(A.13)}$$

A.3 Gradient with Respect to U. The approximate gradient of the objective function with respect to the lateral weights U is

$$\frac{\partial J(t)}{\partial U_{kj}} = \frac{(t-1)}{t}\frac{\partial J(t-1)}{\partial U_{kj}} + \frac{1}{t}v(t)z(t) \quad \text{(A.14)}$$

$$z(t) = \tau z(t-1) + \frac{1}{P(t)}\frac{\partial P(t)}{\partial U_{kj}} \quad \text{(A.15)}$$

$$\frac{\partial \log P(t)}{\partial U_{kj}} = \sum_j \frac{\partial h_t^j}{\partial U_{kj}}\left(\rho_t^j - \sigma\left(h_t^j\right)\right) \quad \text{(A.16)}$$

$$\frac{\partial h_t^j}{\partial U_{kj}} = \rho_{t-1}^k + \eta(1)\frac{\partial h_{t-1}^j}{\partial U_{kj}}. \quad \text{(A.17)}$$

Plugging these values back into equation A.14 yields

$$\frac{\partial J(t)}{\partial U_{kj}} = \frac{(t-1)}{t}\frac{\partial J(t-1)}{\partial U_{kj}} + \frac{1}{t}v(t) \times \left(\tau z(t-1) + \sum_j \left(\rho_{t-1}^k + \eta(1)\frac{\partial h_{t-1}^j}{\partial U_{kj}}\right)\left(\rho_t^j - \sigma(h_t^j)\right)\right). \quad \text{(A.18)}$$

A.4 Gradient with Respect to β. The approximate gradient of the objective function with respect to the temporal integration constant β is

$$\frac{\partial J(t)}{\partial \beta} = \frac{(t-1)}{t}\frac{\partial J(t-1)}{\partial \beta} + \frac{1}{t}v(t)z(t) \quad \text{(A.19)}$$

$$z(t) = \tau z(t-1) + \frac{1}{P(t)}\frac{\partial P(t)}{\partial \beta} \quad \text{(A.20)}$$

$$\frac{\partial \log P(t)}{\partial \beta} = \sum_j \frac{\partial h_t^j}{\partial \beta}\left(\rho_t^j - \sigma\left(h_t^j\right)\right). \quad \text{(A.21)}$$

$$\frac{\partial h_t^j}{\partial \beta} = -\eta(1)h_{t-1}^j + \eta(1)\frac{\partial h_{t-1}^j}{\partial \beta}. \quad \text{(A.22)}$$

Acknowledgments

This work was supported by NSERC and CIHRC NET program (R.N., R.Z.), the BIBA consortium (Q.H., P.D.), the UCL Medical School M.B./Ph.D. program (Q.H.), and the Gatsby Charitable Foundation (Q.H., P.D.). We thank two anonymous reviewers for their extensive and helpful comments.

References

Ackley, D. H., Hinton, G. F., & Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, *9*, 147–169.

Barbieri, R., Frank, L. M., Nguyen, D. P., Quirk, M. C., Solo, V., Wilson, M. A., & Brown, E. N. (2004). Dynamic analyses of information encoding in neural ensembles. *Neural Comp.*, *16*, 277–307.

Baxter, J., & Bartlett, P. L. (2001). Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, *15*, 319–350.

Brockwell, A. E., Rojas, A. L., & Kass, R. E. (2004). Recursive Bayesian decoding of motor cortical signals by particle filtering. *J. Neurophysiol.*, *91*, 1899–1907.

Brown, E. N., Frank, L. M., Tang, D., Quirk, M. C., & Wilson, M. A. (1998). A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *J. Neurosci.*, *18*(18), 7411–7425.

Cover, T., & Thomas, J. (2006). *Elements of information theory* (2nd ed.). New York: Wiley.

Dean, I., Harper, N. S., & McAlpine, D. (2005). Neural population coding of sound level adapts to stimulus statistics. *Nat. Neurosci.*, *8*(12), 1097–6256.

Gao, Y., Black, M. J., Bienenstock, E., Shoham, S., & Donoghue, J. P. (2002). Probabilistic inference of hand motion from neural activity in motor cortex. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems, 14*. Cambridge, MA: MIT Press.

Hinton, G. E., & Brown, A. D. (2000). Spiking Boltzmann machines. In S. A. Solla, T. K. Leen, & K.-R. Miller (Eds.), *Advances in neural information processing systems, 12*. Cambridge, MA: MIT Press.

Huys, Q. J. M., Zemel, R., Natarajan, R., & Dayan, P. (2007). Fast population coding. *Neural Comp.*, *19*(2), 404–441.

Kistler, W., Gerstner, W., & van Hemmen, J. L. (1997). Reduction of Hodgkin-Huxley equations to a threshold model. *Neural Comput.*, *9*, 1069–1100.

Körding, K., & Wolpert, D. M. (2004). Bayesian integration in sensorimotor learning. *Nature*, *427*(6971), 244–247.

MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge: Cambridge University Press.

Nirenberg, S., Carcieri, S. M., Jacobs, A. L., & Latham, P. E. (2001). Retinal ganglion cells act largely as independent encoders. *Nature*, *411*, 698–701.

Paninski, L. (2003). Convergence properties of three spike-triggered analysis techniques. *Network*, *14*(3), 437–464.

Pola, G., Thiele, A., Hoffmann, K.-P., & Panzeri, S. (2003). An exact method to quantify the information transmitted by different mechanisms of correlational coding. *Network: Comput. Neural Syst.*, *14*(1), 35–60.

Pouget, A., Dayan, P., & Zemel, R. S. (2003). Inference and computation with population codes. *Annu. Rev. Neurosci.*, *26*, 318–410.

Pouget, A., Deneve, S., Ducom, J.-C., & Latham, P. E. (1999). Narrow vs. wide tuning curves: What's best for a population code? *Neural Computation*, *11*(1), 85–90.

Pouget, A., Zhang, K., Deneve, S., & Latham, P. E. (1998). Statistically efficient estimation using population coding. *Neural Comp.*, *10*(2), 373–401.

Seung, S. H., & Sompolinsky, H. (1993). Simple models for reading neuronal population codes. *Proc. Natl. Acad. Sci. USA*, *90*(22), 10749–10753.

Smith, E., & Lewicki, M. S. (2005). Efficient coding of time-relative structure using spikes. *Neural Comp.*, *17*, 19–45.

Twum-Danso, N., & Brockett, R. (2001). Trajectory estimation from place cell data. *Neural Networks*, *14*, 835–844.

Vickers, N. J., Christensen, T. A., Baker, T., & Hildebrand, J. G. (2001). Odour-plume dynamics influence the brain's olfactory code. *Nature*, *410*, 466–470.

Wu, S., Nakahara, H., & Amari, S. (2001). Population coding with correlation and an unfaithful model. *Neural Comp.*, *13*(4), 775–797.

Wu, W., Black, M. J., Mumford, D., Gao, Y., Bienenstock, E., & Donoghue, J. P. (2004). Modeling and decoding motor cortical activity using a switching Kalman filter. *IEEE Trans. Biomedical Engineering*, *51*(6), 933–942.

Zemel, R. S., Dayan, P., & Pouget, A. (1998). Probabilistic interpretation of population codes. *Neural Comp.*, *10*(2), 403–430.

Zemel, R. S., Huys, Q. J. M., Natarajan, R., & Dayan, P. (2005). Probabilistic computation in spiking populations. In L. K. Saul, Y. Weiss, Bottou (Eds.), *Advances in neural information processing systems, 17* (pp. 1609–1616). Cambridge, MA: MIT Press.

Zhang, K., & Sejnowski, T. J. (1999). Neuronal tuning: To sharpen or To broaden. *Neural Comp.*, *11*(1), 75–84.

Received January 14, 2007; accepted December 26, 2007.