Title: Probability Density Methods for Smooth Function-Approximation and Learning in Populations of Tuned Spiking Neurons

Author: Terence David Sanger

Institution: Massachusetts Institute of Technology Department of Brain and Cognitive Sciences, and Boston Children's Hospital Department of Neurology.

Contact Address: Terence D. Sanger

MIT, Room E25-526, Cambridge, MA 02139

tel: (617)253-5769, fax: (617)734-5164, email(preferred): tds@ai.mit.edu

Running Title: Function Approximation in Spiking Neurons

Keywords: Spiking Neurons, Population code, Population vector, Probability, Density Estimation, Motor Cortex, Tuning Curves, Poisson process, Cortical information processing, Function Approximation, Learning, Neural Networks

Submitted: *Neural Computation*

Abstract

This paper proposes a new method for interpreting computations performed by populations of spiking neurons. Neural firing is modelled as a rate-modulated random process for which the behavior of a neuron in response to external input can be completely described by its tuning function. I show that under certain conditions a cell with any desired tuning function can be approximated using only spike coincidence detectors and linear operations on the spike output of existing cells. I further show examples of adaptive algorithms based only on spike data that cause the underlying cell tuning curves to converge according to standard supervised and unsupervised learning algorithms. Unsupervised learning based on Principal Components Analysis leads to independent cell spike trains. These results suggest a duality relationship between the random discrete behavior of spiking cells and the deterministic smooth behavior of their tuning functions. Classical Neural Network approximation methods and learning algorithms based on continuous variables can thus be represented implicitly within networks of spiking neurons without the need to make estimates of the actual cell firing rates.

1 Introduction

As electrophysiological recording data from multiple cortical cells becomes available, it is increasingly important that Neural Network models are able to interpret the information and computations performed by the "Neural Code" (Sejnowski 1995). However, an important gap exists between the physiological representation of information and much of the large body of theoretical work on Neural Networks. Many Neural Network models are defined in terms of smooth function approximation (Poggio and Girosi 1990, for example) under the assumption that continuous variables can be represented in the firing rate of cells (Judd and Aihara 1993). Cell behavior is described by a "tuning curve" (Rieke *et al.* 1997, for discussion) that gives the average firing rate as a function of some external variable. It is assumed that once firing rates are known, they can be added, multiplied, and otherwise manipulated according to the rules of continuous mathematics. In fact, this assumption is so basic to such models that it is rarely stated explicitly. Simulated implementations use continuous variables as inputs and hidden units, and compute continuous functions at the outputs without attempting to relate these variables to the actual underlying spike data.

The difficulty with this approach is that in order to extract the continuous rate variable from the spike data, we must count the number of spikes that occur within some fixed time interval. Since a rapidly (and regularly) firing cell might fire at most 100 spikes per second, we would need to count over at least 1 second in order to have an estimated error less than 1 percent (although (Tovee *et al.* 1993) suggests that shorter time-intervals may be possible). If the cell's firing is described by a Poisson process with average rate of 100 spikes/second, then we will need to count over a significantly longer interval in order to make an accurate estimate of the rate (Stein 1967). In either case, the error in rate estimation means that variables can only be approximately represented (Softky 1996) (or equivalently, that these variables are represented with low precision). This situation becomes worse for cells with low firing rates or if the average rate is chaotic (van Vreeswijk and Sompolinsky 1996) or varying with time (August and Levy 1996, Gabbiani and Koch 1996). As experimenters, we frequently have the option of observing a biological network over a long period of time under static conditions, so that we can in fact compute accurate average rates. But a behaving animal cannot afford to wait 1 second or longer at each stage of computation.

There are several ways to address this problem. One is to assume that there are many copies of each cell, so that we can use spatial averaging or a population code to increase the accuracy of rate estimation for short time-windows (Shadlen and Newsome 1994, Maass 1995, Shadlen *et al.* 1996, Theunissen *et al.* 1996). Although this appears to be wasteful of neural resources, it might be feasible given the large number of cortical cells available. Population averaging may fail if there is significant correlation between the output of different cells (Zohary *et al.* 1994). Other methods note the remarkable predictability of spike firing times (Bair and Koch 1996), and can thus use the inter-spike interval (Judd and Aihara 1993, Softky 1994) or the filtered spike train (Bialek and Rieke 1992, August and Levy 1996, Gabbiani and Koch 1996) as an "instantaneous" estimate of the relevant variable. There is considerable theoretical and experimental evidence to support either a rate code (Shadlen and Newsome 1994, Shadlen and Newsome 1995) or single spike codes (Murthy and Fetz 1994, Hopfield 1995, Mainen and Sejnowski 1995, Konig *et al.* 1996), and some comparisons are given in (Stein 1967, Gerstner and van Hemmen 1994, Softky 1995, Softky 1996). A detailed review can be found in (Rieke *et al.* 1997).

The method proposed in this paper is related to both rate codes and single-spike codes, and attempts to incorporate the rapid read-out of spike codes with the computational simplicity of rate codes. The underlying problem for rate codes is that although the firing rates are indeed continuous and often deterministic functions of the cell inputs, it is difficult to calculate these rates explicitly from the pattern of spike data. I propose to avoid this problem entirely by assuming that data is encoded in the instantaneous *probability* of firing. I show that it is possible to connect a network of randomly spiking neurons such that the firing probability of the output neurons is any desired smooth function of the firing probability of the input neurons. The connections depend upon the probability of transmission of single spikes, but there is no delay while spike rates are averaged.

For example, consider computing the identity function such that an output neuron has the same firing rate as an input neuron. In order to do this computation using explicit firing rates, the network could wait for several seconds and count the input neuron spikes, and use this total spike count to modulate the output neuron's rate. Alternatively, many copies of the input neuron could be provided and the total number of spikes over the input population could determine the output neuron's rate. In any finite time period (or finite number of copies of the input neuron) these two methods could only approximate the input firing rate and thus the calculation of the output rate will not be exact. I propose instead that if the output neuron is connected so that it fires every time the (single) input neuron fires, then the average firing rate of the two neurons is identical. This involves no approximation. If the average firing rate of the input neuron changes, then so does the firing rate of the output neuron. Their tuning curves are identical.

The above extremely simple example illustrates an important and subtle point that is the foundation for the rest of this paper: It is possible to compute an output cell's tuning curve *exactly* and deterministically using connections that transmit only the randomly firing spikes. This is because the *probability* of a spike is a continuous deterministic variable that is only implicitly linked to whether or not a spike actually occurred. Since the tuning curve is the average firing rate for each value of a measured variable, it will be proportional to the conditional probability of firing given that variable. Each cell has an implicit probability of firing, and computations on the probabilities can be performed exactly without ever needing to extract these probabilities from the pattern of spike firing. Only at the final output stage of the system when the cells must actually drive the muscles is it necessary to use time or space-averaging to estimate the firing probabilities from the spike data (Stein 1967).

2 Experimental Interpretation of Population Codes

The mathematical foundations of the computational method proposed here come from the Probability Density Estimation method for interpretation of experimental results from population codes (Sanger 1996). This method was proposed as an efficient way for an experimenter to use electrophysiological data to estimate the value of variables represented within populations of spiking neurons. It was not intended to model the extraction of signals within the brain, but rather to indicate the extent to which information is implicit within the neural code. It is this implicit information which is essential to understanding the method proposed here.

To summarize the results in (Sanger 1996), consider a very simple model for neural firing described by

$$P[\mathrm{spike}|x] \propto \sigma(x) \tag{1}$$

meaning that the probability of firing is proportional to a smooth function of an externally measurable variable x. The function $\sigma(x)$ is referred to as the "tuning curve" of the cell with respect to the variable x, and it describes the average firing rate for each value of x. If the probability of firing within any fixed-length time interval is the same, then firing is governed by a Poisson distribution

$$P[n \text{ spikes}|x] = \frac{\sigma(x)^n e^{-\sigma(x)}}{n!} \tag{2}$$

This model is quite general, since x can be a vector if the cell is dependent on multiple variables, and it can include time if the firing rate is time-dependent. The model does not include any representation of refractory periods or the possibility of explicitly coding elements of x using the relative spike times or interspike intervals.

From Bayes' law, we have

$$P[x|\text{spike}] = \sigma(x)^n e^{-\sigma(x)} P[x]/\mathcal{N} \tag{3}$$

where $P[x]$ is the probability distribution of the external variable x (usually controlled by the experimenter) and $\mathcal{N}$ is a normalization constant calculated to make the total probability equal to 1. This equation (for $n = 1$) states what we learn about the variable x from a single spike. Under this model, the conditional density $P[x|\text{spike}]$ completely describes the behavior of this cell.

The probability density estimation method (Sanger 1996) makes the assumption that the spike generating units are independent (see also (Maass 1995)). In other words, for any given value of x, whether cell i with tuning curve $\sigma_i(x)$ fires or not is independent of whether any other cell fired:

$$P[\text{spike}_i \text{ and } \text{spike}_j|x] = P[\text{spike}_i|x]P[\text{spike}_j|x] \tag{4}$$

for all values of x. If each cell i fires s_i times during some time interval, then we can immediately calculate the conditional density of x given the entire population of cells as (Sanger 1996)

$$P[x|\text{population}] = \frac{1}{\mathcal{N}} P[x] \prod_i e^{-\sigma_i(x)} \sigma_i(x)^{s_i} \tag{5}$$

The conditional density of x is implicit in the population firing response, and this is the *most* information about x that can be extracted from this population. If we wish to extract a best estimate of a single value of x from the density information, then we can take the peak of the conditional density which will give the *Maximum A Posteriori* (MAP) estimate of x given the population spike data. (Similarly, the minimum mean-squared error (MMSE) estimate is given by the mean of the conditional density.)

The significance of equation 5 is that we can *in principle* extract this estimate from the electrophysiological data. Therefore the complete density of x is implicit in the spike information. However, it is not suggested that a behaving brain actually performs this calculation. I will show below that the brain can make use of this information without extracting explicit estimates of x. The information is present, but it does not need to be manipulated into a single continuous variable in order to perform computations.

Other methods for estimating the value of a variable x from population spike data exist, but they do not attempt to calculate the complete density information. To compare these methods, define the column vector S to be a vector of integers s_i where each s_i gives the number of times that cell i fired during some measurement interval. Assume for simplicity that the externally measurable variable x is a scalar. Over many sample measurements, we can compute the autocorrelation matrix

$$K_{SS} = E[SS^T]$$

where the ij'th element of K_{SS} is the average value of the product $s_i s_j$. We can also compute the cross-correlation vector

$$K_{xS} = E[xS]$$

where the i'th element of K_{xS} is the average value of the product xs_i. Then the method proposed by (Georgopoulos *et al.* 1988) estimates x using

$$\hat{x} = K_{xS}S \tag{6}$$

and the Optimal Linear Estimator proposed by (Salinas and Abbott 1994) uses

$$\hat{x} = K_{xS}K_{SS}^{-1}S \tag{7}$$

As shown in (Sanger 1994) for estimation of arm movement direction in motor cortex, the different estimates for x are the same if the "preferred directions" of the cell tuning curves are uncorrelated so that K_{SS} is the identity.

There is another class of algorithms that attempts *explicitly* to estimate the probability density for x (Anderson 1995, Zemel *et al.* 1997). In certain cases, if the tuning curves $\sigma_i(x)$ are local in the sense that they respond only to a small range of contiguous values of x, then the pattern of firing rates (averaged over time) gives an explicit estimate of the probability density of x. These methods differ from the method proposed here in that they depend upon knowledge of the average firing rates, and the goal is the explicit representation of the density in the neural firing pattern. The probability density estimation method, in contrast, extracts the probability density of x given whatever pattern the cells happen to produce, for arbitrary cell tuning curves that might have been designed for a very different purpose.

3 Implicit Computation with Spiking Neurons

The method described in section 2 above provides a technique by which an experimenter can extract the maximal information from a pattern of spiking neurons, and it indicates the amount of information present in the population. However, it is not intended to be a model of information processing in the brain. In this section I propose a related model that shows how simple connections between spiking neurons can be arranged in such a way that tuning curves can be calculated to approximate a large number of desired functions. Although the calculated tuning curves will determine the spiking behavior of their respective cells, the value of the tuning curve is never explicitly computed.

Under the models given by equations 1 and 2, a cell's behavior is completely described by its tuning curve, and the tuning curve can (eventually) be extracted as the long-time average firing rate of the cell for each value of x. Therefore, the long-time spike data and the tuning curve essentially are two different ways of describing the same cell. Thus, in order to create a cell with any particular behavior, we only need to choose a desired tuning curve and synthesize a cell with this tuning curve using the spike data from existing cells. For example, a motor neuron that controls a muscle which is intended to fire for movements that extend a particular joint needs to be connected to other joint-related neurons (with a variety of different tuning curves) in such a way that its tuning curve is a monotonically increasing function of joint angle.

Consider a neuron k that is a "coincidence detector" for neurons i and j and thus has a nonzero probability of firing only if i and j both fire within a small time-window. In this situation, we can write

$$\sigma_k = \lambda s_i s_j$$

where σ_k is the probability of neuron k firing, s_i and s_j are either one or zero depending on whether or not each neuron fired, and λ is a scaling factor. (If the neurons fire more than once in the time-window, then we can allow s_i and s_j to be larger integers.) Note that σ_k is a function of the two random variables s_i and s_j, and thus the event of neuron k firing is a Poisson process modulated by a random variable. For Poisson neurons, the expected firing rate given x is

$$E[\sigma_k(x)] = \lambda E[s_i s_j | x] \tag{8}$$

and since $P[s_i s_j|x] = P[s_i|x]P[s_j|x]$ by the independence assumption, we have

$$E[\sigma_k(x)] = \lambda E[s_i|x]E[s_j|x] = \lambda \sigma_i(x)\sigma_j(x) \tag{9}$$

(Note that the expected firing rate $E[\sigma_k(x)]$ needs to be used here since $\sigma_k(x)$ is a random variable.) So the expected tuning curve (and thus the average firing rate) for k is proportional to the product of the tuning curves for i and j. This relationship was created simply by making k a coincidence detector. There was no need to compute the firing rates or tuning curves of i or j explicitly. Figure 1 show a simulated example.

$\boxed{\text{Figure 1 near here}}$

Now, consider a cell whose firing rate is a weighted linear combination of the spike outputs of a set of other cells, so that

$$\sigma_k = \sum_i w_i s_i \tag{10}$$

where the "synaptic weights" w_i are real-valued, and indicate the incremental probability that neuron k will fire given that neuron i fired. Then

$$E[\sigma_k(x)] = \sum_i w_i E[s_i|x] = \sum_i w_i \sigma_i(x) \tag{11}$$

so the average firing rate is a linear combination of the firing rates of the input neurons $\sigma_i(x)$. Figure 2 shows a simulated example. Note that the tuning curve σ_k was produced exactly as a linear combination of the σ_i's without ever calculating the actual firing rates of the input neurons. In fact, σ_k itself remains implicit and cannot be extracted without time-averaging. Since we can now calculate linear combinations as well as products of input tuning curves, we have the ability to calculate polynomials and thus can approximate any desired function of the input probability distributions with arbitrary accuracy. (Polynomial computations and multiplicative interactions in neural networks are discussed in (Koch and Poggio 1992, Mel 1993, Tal and Schwartz 1997).)

Figure 2 near here

It is important to note some interesting properties of this form of implicit computation. For one thing, the actual spike rates do not need to be calculated until they are needed at the output stage to control muscles. The probability values propagate implicitly. Although the spikes are random binary events, the underlying probabilities are continuous variables that can be computed exactly. The computational delay is exactly equal to the synaptic delay; an output cell has the correct probability of firing as soon as an input cell spike *could have* affected it, even if the input cell has not yet actually fired. Time- or Population-averaging may be needed to read out the information or to control muscles, but it is not needed for intermediate computations. An alternative model that has very short synaptic delays and is based on relative spike timing can be found in (Maass 1994, Maass 1997, Maass 1996).

4 Learning Algorithms

In the previous section, I showed how it is possible to perform smooth function approximation on the implicit firing probability of cells. The question naturally arises as to whether desired functions can be learned from examples. In this section I show that it is possible to find "spike-equivalent" forms of common neural network algorithms such that the underlying implicit probabilities converge according to the (smooth) algorithms while the synaptic weights change in response to cell spiking. The algorithms are closely related to neural network models of stochastic convergence, since it turns out that the expected change in weights from the spike-based algorithms is equal to the weight change for the smooth implicit algorithms. There are many examples in biology of adaptation in response to spike coincidence, although the relationship to current neural network models can be difficult to elucidate (Magee and Johnston 1997, Markram *et al.* 1997).

First, I show an example of a supervised learning algorithm whose average behavior approximates the Widrow-Hoff "LMS" algorithm (Widrow and Hoff 1960, Widrow *et al.* 1976). Let p be a "teaching" neuron with tuning curve $\sigma_p(x)$, and let k be a neuron that we would like to train so that $E[\sigma_k(x)] = \sigma_p(x)$. The firing rate of k is a linear combination of the spikes of other cells i according to equation 10. The training algorithm has access only to the spike data, and it must adapt the synaptic weights w_i from the set of input neurons i to the output neuron k. Let s_p, s_k, and s_i take the value 1 if a spike occurred in cells p, k, or i. Consider the adaptation algorithm

$$\Delta w_i = \gamma(s_p - s_k)s_i \tag{12}$$

which means that the weight increases or decreases by zero or γ depending on which cells fired. The expected change in weight is

$$\begin{aligned} E[\Delta w_i|x] &= \gamma E[(s_p - s_k)s_i|x] && (13) \\ &= \gamma(P[s_p s_i|x] - P[s_k s_i|x]) && (14) \\ &= \gamma(P[s_p|x]P[s_i|x] - P[s_k s_i|x]) && (15) \end{aligned}$$

since neurons p and i have independent firing by assumption. Neurons k and i are not independent, since the firing rate of k is directly modulated by whether or not neuron i fired. We have

$$\begin{aligned} P[s_k s_i|x] &= P[s_k|s_i, x]P[s_i|x] && (16) \\ &= \left(w_i + \sum_{j \neq i} w_j P[s_j|x] \right) P[s_i|x] && (17) \\ &= \left(w_i(1 - P[s_i|x]) + \sum_{j} w_j P[s_j|x] \right) P[s_i|x] && (18) \\ &= (w_i(1 - P[s_i|x]) + P[s_k|x])P[s_i|x] && (19) \end{aligned}$$

and combining gives

$$E[\Delta w_i|x] = \gamma(P[s_p|x] - P[s_k|x])P[s_i|x] - w_i(1 - P[s_i|x])P[s_i|x] \qquad (20)$$

The first term on the right of equation 20 is the Widrow-Hoff supervised linear learning rule. The last term is always negative, and gives a decay term for w_i that is most significant if the probability of firing for neuron i is close to 1/2. An example of convergence of the tuning curve when synapses are trained according to equation 12 is shown in figure 3.

Figure 3 near here

I now show an example of an unsupervised Hebbian learning algorithm that causes the tuning curves to converge to the eigenfunctions of the input probability density. Previous examples of Hebbian learning for spike data are found in (Brown *et al.* 1992, Gerstner *et al.* 1996), and the relationship between principal components analysis and information maximization is elegantly derived in (Fry 1995).

Assume there are multiple outputs σ_k for many values of k, and let

$$\sigma_k = \sum_i w_{ki} s_i$$

and consider the learning algorithm

$$\Delta w_{ki} = \gamma \left(s_i - \sum_{j \leq k} w_{kj} s_j \right) s_i \tag{21}$$

Then we can compute the average weight update as

$$\begin{aligned}
E[\Delta w_{ki}|x] &= \gamma E \left[\left(s_i - \sum_{j \leq k} w_{kj} s_j \right) s_k | x \right] && (22) \\
&= \gamma \left(E[s_i|x] - \sum_{j \leq k} w_{kj} E[s_j|x] \right) E[s_k|x] \\
&\quad - w_{ki}(1 - P[s_i|x])P[s_i|x] - \sum_{j \leq k} w_{kj}^2 (1 - P[s_j|x])P[s_j|x] && (23) \\
&= \gamma \left(\sigma_i(x) - \sum_{j \leq k} wkj \sigma_j(x) \right) \sigma_k(x) \\
&\quad - w_{ki}(1 - \sigma_i(x))\sigma_i(x) - \sum_{j \leq k} w_{kj}^2 (1 - \sigma_j(x))\sigma_j(x) && (24)
\end{aligned}$$

and it should be noted that the first term of equation 24 describes the Generalized Hebbian Learning Algorithm that finds the principal components of the input distribution (Sanger 1989). The second term in equation 24 is a weight decay term, and the third term is always less than zero and thus also contributes to weight decay, although not proportionately to the weight w_{ki} itself. figure 4 shows an example of the convergence of the unsupervised spike algorithm for simulated data. The graph gives the "diagonalization score" which is a measure of the extent to which the matrix of cross-products of network outputs $E[\sigma_k(x)\sigma_m(x)]$ is diagonal. The score is the sum of the magnitudes of the off diagonal elements divided by the sum of the magnitudes of the diagonal elements. Low numbers indicate a more diagonal matrix, and therefore a set of uncorrelated tuning curves.

Figure 4 near here

Both the supervised and unsupervised algorithms make use of "random" spike data to perform learning whose long-time average behavior approximates smooth learning algorithms. These algorithms demonstrate the possibility of performing standard learning tasks in the single-spike domain, but there is not yet evidence to suggest them as a literal model of biological information processing.

5 Unsupervised Learning and Information

After the unsupervised algorithm has been trained, the tuning curves are orthogonal. In this section I show that this implies that the spike trains then become independent. This is important, since it implies that orthogonality of the tuning curves is the most "separate" that we can make the behavior of two cells. To see this, note that the mean of the conditional density $P[s_i|x]$ is given by

$$\int P[s_i|x]P[x]dx = P[s_i]$$

and define the zero-mean function

$$f_i(x) = P[s_i|x] - P[s_i]$$

for which

$$\int f_i(x)P[x]dx = 0.$$

Now under the independence assumption we can write

$$P[s_i s_j] = \int P[s_i s_j|x]P[x]dx \tag{25}$$

$$= \int P[s_i|x]P[s_j|x]P[x]dx \tag{26}$$

$$= \int (f_i(x) + P[s_i])\,(f_j(x) + P[s_j])\,P[x]dx \tag{27}$$

$$= P[s_i]P[s_j] + \int f_i(x)f_j(x)P[x]dx \tag{28}$$

and we see that $P[s_i s_j] = P[s_i]P[s_j]$ (meaning that the spikes are independent) if and only if

$$\int f_i(x)f_j(x)P[x]dx = 0$$

so that the (zero-mean) tuning curves are orthogonal. The converse also holds, so that maximum dependence occurs for maximally cross-correlated tuning curves. If we set $f_i(x) = f_j(x)$, then this shows that maximizing the variance of the tuning curve

$$\int f_i^2(x)P[x]dx$$

will maximize

$$P[s_i^2] - P[s_i]^2 = P[s_i](1 - P[s_i])$$

which is greatest for $P[s_i]$ close to $1/2$. Since this value maximizes the spike entropy, there is a relationship between maximizing the variance of the tuning curve and maximizing the entropy of the spike firing probability.

It is surprising to find such relationships between linear properties of the zero-mean tuning curves such as cross-correlation or variance, and nonlinear properties of the spike trains such as independence or entropy. Yet the existence of such relationships shows that simple linear algorithms on the underlying probability distributions can lead to optimal nonlinear behavior at the level of the spikes. This fact can considerably simplify the types of calculations that are necessary, and gives neural networks an important set of tools for performing nonlinear computations.

6 Duality

The results above demonstrate a close link between the behavior of spiking neurons and the underlying implicit tuning curves that describe their conditional probability of firing for different values of x. This link is a natural and unavoidable property of probabilistic models of spiking neurons. It is not specific to Poisson firing statistics, but will arise for any neuron model in which the probability of firing is smoothly modulated by a continuous tuning curve. The spike behavior and the tuning curve properties are two different yet completely equivalent ways of describing a neuron's response, and we can take advantage of their different mathematical properties to simplify certain types of computations.

The equivalence of these two descriptions leads to a duality relationship between the space of probabilistic events (spikes) and the space of deterministic bounded integrable functions (tuning curves) of a random variable x. In the spike "space", maximum entropy and independence are related to maximum variance and orthogonality in the tuning curve "space". Polynomial functions in the two spaces are equivalent (under the assumption of independent spike generators). I hypothesize that for any learning algorithm in the tuning curve space there is a spike learning algorithm with similar convergence properties, as I have shown above for the LMS and Generalized Hebbian Algorithms.

7 Conclusion

The analysis given above shows the possibility of a direct link between familiar continuous neural network algorithms and learning algorithms performed by spiking neurons. It is not necessary to postulate that unit "activity levels" in a neural network are somehow formed from average neural firing rates (which require a long time to estimate), since these activity levels can be directly represented by the instantaneous *probability* of firing. Computations can be performed directly on firing probabilities without the necessity for first estimating these probabilities by estimating the average firing rate. This saves considerable time in synaptic transmission, and provides a better model for the rapid transmission of information in large biological neural networks.

Although the computation is essentially noise-free and occurs with minimal delays, the model still begs the question of how the data is eventually read out. (Stein 1967) gives a detailed discussion of approximation error for reading out rate codes. As for any neuronal processing, the code must necessarily be mapped from the final layer of processing onto the relevant muscles. The significant advantage of the model proposed here is that the rate code needs to be interpreted only at the final step of computation, so that it does not contribute to approximation errors at every step.

The model proposed here is perhaps most closely related to rate coding, since it provides a rapid computational method based upon average firing rates, although without explicitly computing those rates. It is different from codes which base processing on explicit differences in spike firing times or interspike intervals. For example, (Softky 1996) argues for the use of a binary pulse code for maximizing the transmission of information. Such a code is definitely superior at the time of read-out, but it has considerable disadvantages in terms of the complexity of intermediate computations and learning. The distinction between "simple" rate codes and "efficient" binary spike codes (Softky 1995) is blurred by the model proposed here, since we have both the advantages of simple computation based on average firing rate without the disadvantage of slow and error-prone computations.

It is important to realize that the results above are inherent properties of probabilistic models of spiking neurons. Whether or not a system chooses to make use of these properties, they are necessarily present. A cell with a tuning curve has an implicit conditional probability of firing, and when connected to other cells will have a direct effect on their probabilities of firing. This effect requires only the synaptic transmission time, and is not dependent on temporal integration of spike rates. There is evidence that cell tuning is present within the first few spikes, thus suggesting that rapid processing of average rate information is indeed occurring in biological systems (Celebrini *et al.* 1993, Tovee *et al.* 1993).

The work above has shown probabilistic methods for interpreting the patterns of activity in populations of spiking neurons, computing new neurons with desired tuning curves, and learning in both supervised and unsupervised modes. It is hoped that such methods may form the basis for a probabilistic "Neural Calculus" that both allows interpretation of biological neural systems, as well as explains some of the ways in which those systems perform internal computations.

8 Acknowledgements

I wish to thank Emilio Bizzi, Simon Giszter, Peter Dayan, Barak Pearlmutter, and the reviewers for very helpful comments and discussions. The author was supported by a McDonnell-Pew Postdoctoral Fellowship. Dr. Bizzi's laboratory is supported by grants from the National Institute of Health (NS09343) and the Office of Naval Research (N00014/90/J/1946).

References

Anderson C. H., 1995, Unifying perspectives on neuronal codes and processing, Proc. 19th Int'l workshop on Condensed Matter Theories, Caracas, Venezuela.

August D. A., Levy W. B., 1996, A simple spike train decoder inspired by the sampling theorem, *Neural Computation*, 8:67–84.

Bair W., Koch C., 1996, Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey, *Neural Computation*, 8:1185–1202.

Bialek W., Rieke F., 1992, Reliability and information transmission in spiking neurons, *Trends in Neuroscience*, 15(11):428–434.

Brown T. H., Zador A. M., Mainen Z. F., Claiborne B. J., 1992, Hebbian computations in hippocampal dendrites and spines, In *Single Neuron Computation*, pages 81–116, Academic Press, Boston, MA.

Celebrini S., Thorpe S., Trotter Y., Imbert M., 1993, Dynamics of orientation coding in area V1 of the awake primate, *Visual Neuroscience*, 10:811–825.

Fry R. L., 1995, Observer-participant models of neural processing, *IEEE Trans. Neural Networks*, 6(4):918–928.

Gabbiani F., Koch C., 1996, Coding of time-varying signals in spike trains of integrate-and-fire neurons with random threshold, *Neural Computation*, 8:44–66.

Georgopoulos A. P., Kettner R. E., Schwartz A. B., 1988, Primate motor cortex and free arm movements to visual targets in three-dimensional space. II. Coding of the direction of movement by a neuronal population, *J. Neurosci*, 8(8):2928–2937.

Gerstner W., van Hemmen J. L., 1994, How to describe neuronal activity: Spikes, rates, or assemblies?, In *Advances in Neural Information Processing Systems*, volume 6, pages 463–470, Morgan Kaufmann, San Mateo, CA.

Gerstner W., Kempter R., van Hemmen J. L., Wagner H., 1996, A neuronal learning rule for sub-millisecond temporal coding, *Nature*, 383:76–78.

Hopfield J. J., 1995, Pattern recognition computation using action potential timing for stimulus representation, *Nature*, 376:33–36.

Judd K. T., Aihara K., 1993, Pulse propagation networks: A neural network model that uses temporal coding by action potentials, *Neural Networks*, 6:203–215.

Koch C., Poggio T., 1992, Multiplying with synapses and neurons, In *Single Neuron Computation*, pages 315–345, Academic Press, Boston, MA.

Konig P., Engel A. K., Singer W., 1996, Integrator or coincidence detector? the role of the cortical neuron revisited, *Trends in Neuroscience*, 19:130–137.

Maass W., 1994, On the computational complexity of networks of spiking neurons, In *Advances in Neural Information Processing Systems*, volume 7, pages 183–190, MIT Press, Cambridge, MA.

Maass W., 1995, On the computational power of noisy spiking neurons, In *Advances in Neural Information Processing Systems*, volume 7, pages 183–190, MIT Press, Cambridge, MA.

Maass W., 1996, Lower bounds for the computational power of networks of spiking neurons, *Neural Computation*, 8:1–40.

Maass W., 1997, Fast sigmoidal networks via spiking neurons, *Neural Computation*, 9:279–304.

Magee J. C., Johnston D., 1997, A synaptically controlled, associative signal for Hebbian plasticity in hippocampal neurons, *Science*, 275:209–213.

Mainen Z. F., Sejnowski T. J., 1995, Reliability of spike timing in neocortical neurons, *Science*, 268:1503–1506.

Markram H., Lubke J., Frotscher M., Sakmann B., 1997, Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, *Science*, 275:213–215.

Mel B. W., 1993, Synaptic integration in an excitable dendritic tree, *J. Neurophysiology*, 70(3):1086–1101.

Murthy V. N., Fetz E. E., 1994, Effects of input synchrony on the firing rate of a three-conductance cortical neuron model, *Neural Computation*, 6:1111–1126.

Poggio T., Girosi F., 1990, Regularization algorithms for learning that are equivalent to multilayer networks, *Science*, 247:978–982.

Rieke F., Warland D., van Steveninck R., Bialek W., 1997, *Spikes: Exploring the Neural Code*, MIT Press, Cambridge, MA.

Salinas E., Abbott L. F., 1994, Vector reconstruction from firing rates, *J. Computational Neuroscience*, 1:89–107.

Sanger T. D., 1989, Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks*, 2:459–473.

Sanger T. D., 1994, Theoretical considerations for the analysis of population coding in motor cortex, *Neural Computation*, 6(1):12–21.

Sanger T. D., 1996, Probability density estimation for the interpretation of neural population codes, *J. Neurophysiology*, 76(4):2790–2793.

Sejnowski T. J., 1995, Time for a new neural code?, *Nature*, 376:21–22.

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

Shadlen M. N., Newsome W. T., 1995, Is there a signal in the noise?, *Current Opinion in Neurobiology*, 5:248–250.

Shadlen M. N., Britten K. H., Newsome W. T., Movshon J. A., 1996, A computational analysis of the relationship between neuronal and behavioral responses to visual motion, *J. Neuroscience*, 16(4):1486–1510.

Softky W. R., 1994, Sub-millisecond coincidence detection in active dendritic trees, *Neuroscience*, 58(1):13–41.

Softky W. R., 1995, Simple codes versus efficient codes, *Current Opinion in Neurobiology*, 5:239–247.

Softky W. R., 1996, Fine analog coding minimizes information transmission, *Neural Networks*, 9(1):15–24.

Stein R. B., 1967, The information capacity of nerve cells using a frequency code, *Biophysical Journal*, 7:797–826.

Tal D., Schwartz E. L., 1997, Computing with the leaky integrate-and-fire neuron: Logarithmic computation and multiplication, *Neural Computation*, 9.

Theunissen F., Roddey J. C., Stufflebeam S., Clague H., Miller J. P., 1996, Information theoretic analysis of dynamical encoding by four identified primary sensory interneurons in the cricket cercal system, *J. Neurophysiology*, 75(4):1345–1364.

Tovee M. J., Rolls E. T., Treves A., Bellis R. P., 1993, Information encoding and the responses of single neurons in the primate temporal visual cortex, *J. Neurophysiology*, 70(2):640–654.

van Vreeswijk C., Sompolinsky H., 1996, Chaos in neuronal networks with balanced excitatory and inhibitory activity, *Science*, 274:1724–1726.

Widrow B., Hoff M. E., 1960, Adaptive switching circuits, In *IRE WESCON Conv. Record, Part 4*, pages 96–104.

Widrow B., McCool J. M., Larimore M. G., Johnson C. R., 1976, Stationary and nonstationary learning characteristics of the LMS adaptive filter, *Proc. IEEE*, 64(8):1151–1162.

Zemel R. S., Dayan P., Pouget A., 1997, Probabilistic interpretation of population codes, to appear in *Advances in Neural Information Processing Systems 9*.

Zohary E., Shadlen M. N., Newsome W. T., 1994, Correlated neuronal discharge rate and its implications for psychophysical performance, *Nature*, 370:140–143.

List of Figures

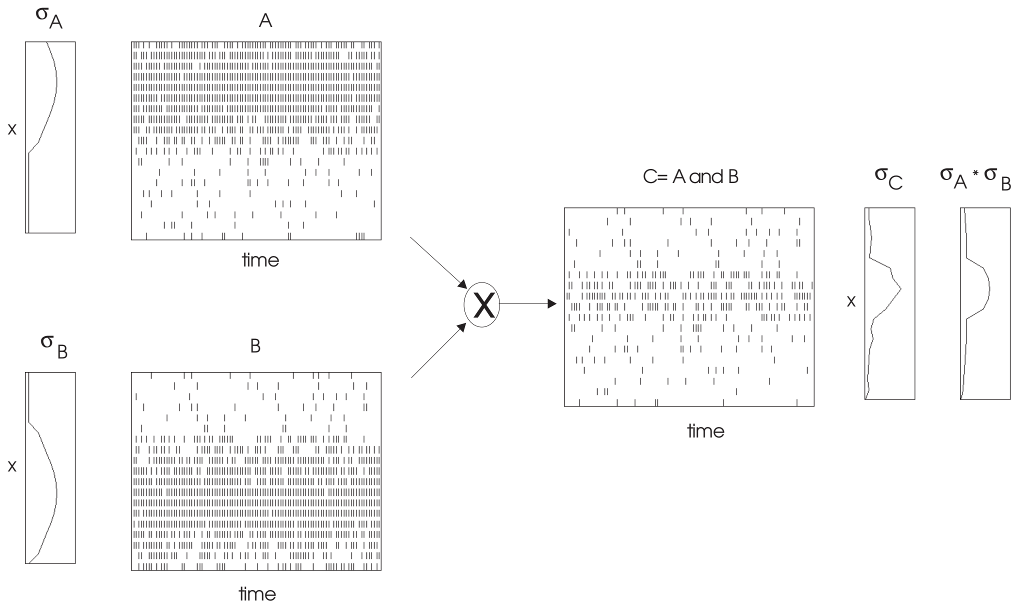

Figure 1: Simulation of tuning curve multiplication. The probability that cell C fires is nonzero only when both cells A and B fire simultaneously. Each row of the simulated spike plots shows the simulated spikes over time for one fixed value of x. $\sigma_A(x)$ and $\sigma_B(x)$ are the tuning curves for cells A and B, and $\sigma_C(x)$ is the tuning curve for C estimated from the spike data. On the right side of the figure, σ_C is compared to the product of σ_A and σ_B.

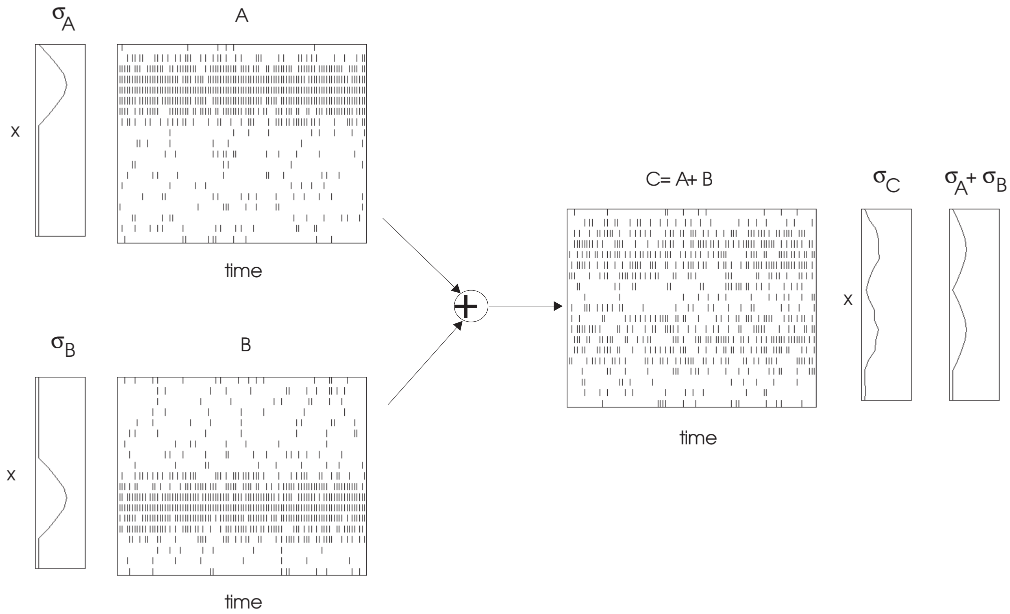

Figure 2: Simulation of tuning curve addition. The probability that cell C fires is proportional to the sum of the total number of spikes of cells A and B in any time interval. On the right side of the figure, $\sigma_C(x)$ is compared to the sum of σ_A and σ_B.

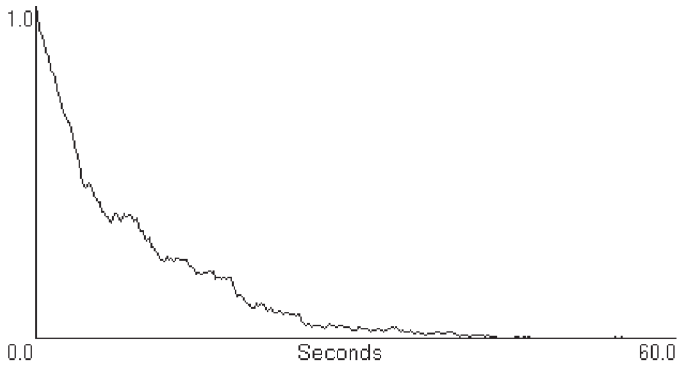

Figure 3: The normalized mean squared error as a function of time for a neuron trained with the supervised spike learning algorithm. There were 20 input cells with a maximum spike rate of 50/second and randomly-chosen smooth tuning curves. The target output cell tuning curve was also chosen randomly. The learning rate is 0.001, and spike coincidence was calculated within 20msec time windows.

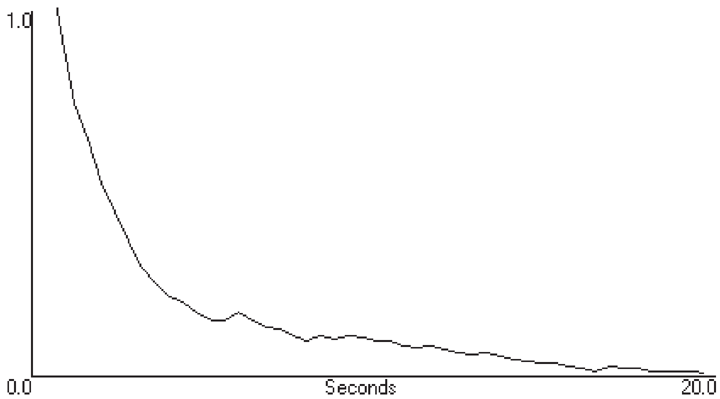

Figure 4: The diagonalization score as a function of time for the spike unsupervised Generalized Hebbian Algorithm. There are 20 input cells, 4 output cells, and 1000 time steps of 20msec each. The learning rate decreased from 0.015 to 0.0055 over the learning.