# A Probability Interpretation of Neural Population Coding for Movement

*Terence D. Sanger*
Massachusetts Institute of Technology
Room E25-526, Cambridge, MA 02139
tds@ai.mit.edu

### Abstract

Although theoretical neural network models frequently assume that the firing rates of individual neurons carry the analog signals used for computations, it is becoming clear that in many sensory and motor systems, information is encoded as patterns of activity over large populations of cells. This type of distributed encoding has many useful computational and representational properties. Under certain conditions, information can be extracted from such a cell population by forming a "population vector" defined to be a linear combination of the cell firing rates weighted by the "preferred" stimulus for each cell (Georgopoulos *et al.* 1988). However, it can be shown that the population vector is not unique and that any population of cells can be taken to represent many different possible input variables, depending on how the component cells are labelled (Mussa-Ivaldi 1988, Sanger 1994). This leads naturally to the question of whether there is a "best" variable represented by a given cell population.

By identifying a cell's receptive field profile with the posterior distribution of the input data or motor response given that the cell has fired, it is possible to interpret the average firing rates of the cells in a population as providing samples from the probability distribution of the represented data (Anderson 1995, Sanger 1996). I derive the maximum *a posteriori* estimate of the input variable given the population firing pattern, and I show the existence of a simple method for extracting its conditional distribution given the population data. This probabilistic interpretation of population codes allows us to use maximum entropy arguments to determine both the optimal population code for representing a given variable, as well as the variable that is represented best by a given population code.

# 1 Distributed Representations and Population Codes

The common language of mathematics makes extensive use of continuous-valued variables that directly represent quantities of interest. The most direct physical implementation of computations on such variables leads to analog computers for which the directly-measurable voltage or current values represent the variables of interest. The success of the digital computer is due in part to the fact that it is much easier and less expensive to implement computational hardware when continuous values are not represented by voltages, but rather by bits in a binary number system. In this way, digital computers provide a "distributed" representation of internal variables where any variable is coded using 8, 16, or more voltages whose exact values are relatively unimportant. A similar argument might be applied to the brain. Although it is possible to represent continuous values directly by the firing rate of neurons or the inter-spike interval, it is becoming clear that in many sensory and motor systems these values exist in a distributed representation for which the pattern of activity is more important than the individual cell spikes. We might expect that, as for digital computers, the evolutionary pressure favoring such an implementation would be driven by the interaction of "hardware" constraints with the set of tasks that the system must perform. Different brain regions may well have evolved different representations for internal information that are particularly suited to their required tasks.

As with digital computers, we might expect that the types of computational tasks that are easy or difficult for biological systems will vary depending upon the representation used for the underlying variables. For example, in digital systems multiplication or exponentiation by powers of 2 is easily accomplished, while arbitrary multiplication is considerably harder. This is in contrast to analog systems for which multipliers consist of a single MOS component, but exponentiation can be difficult. I will show below that distributed neural representations have little difficulty computing arbitrary nonlinear functions of a scalar variable, but that it may require significant resources to compute scalar functions of several variables, such as addition or subtraction.

Certain types of operation that we consider "natural" on continuous variables behave very differently in distributed representations. For example, I will show below that it is a very difficult question to determine exactly *what* is represented by a population code. Distributed representations do not have a simple concept of a coordinate system, and in fact they can simultaneously represent many different coordinate systems. Unlike continuous variables, distributed representations in certain cases can give information about confidence in the values they represent. These ideas will be developed in detail below.

This chapter begins to address the problem of determining a consistent mathematical formalism within which we can discuss certain types of distributed representations (Sejnowski 1995). This formalism is based upon the use of probability theory, which arises naturally in the discussion of neural coding due to the apparently random nature of neural spike timing. Recent evidence suggests that this timing may not be nearly as random as first believed (Mainen and Sejnowski 1995). However, the probabilistic nature of external environmental information and neural recording data makes it likely that probability theory will lead to useful insights.

A particularly useful property of probability theory is the association of a density function with a random variable. This means that the behavior of any probabilistic scalar is described by a function. (Anderson 1995) has suggested that there may be a close relationship between population codes and probability density functions, and thus we might expect that the relationship between scalar random variables and their densities may provide a useful model for the relationship between variables coded by cortex and the sets of neural firing patterns that represent them.

In this chapter, I discuss a particular type of distributed representation that has been suggested by recent experiments in primate motor cortex (Schwartz *et al.* 1988). I will first provide some general definitions and terminology, and then go on to describe current Population Vector models and some difficulties that arise with these models. In section 2, I will then propose a probability model based upon the information content of single spikes. This model leads to a simple method for reading out population code data from a set of cells. In section 3, I will show how similar models can be used to read out information from the average spike rates of a set of cells that has been observed for a long time. I will also show how nonlinear computations can be easily performed on time-averaged spike rates. In section 4, I will use the probability formulation and information theory to derive the optimal population code for representing any particular variable. This derivation will be performed both for the single-spike case, as well as the time-averaged spike rate case. I show the "synaptic diffusion" algorithm which is an unsupervised algorithm that optimizes population codes in the time-averaged case under certain circumstances. Finally, in section 5, I show how we can determine what variable is represented best by a given neural population. This derivation is in some ways the inverse of the derivation in section 4, and it allows a partial answer to the lingering question of the role of coordinate systems in neural computation.

## 1.1 Definitions

The most general definition of a "distributed representation" is a set of neural units for which the pattern of activity indicates the value of an underlying coded variable. This is in contrast to an "analog representation" in which the underlying variable is represented directly by the firing rates of one or more cells. The difference is illustrated in figure 1 (see (van Gelder 1991, Snippe 1996) for discussion). The distinction is less clear than it might seem, however, when we include the possibility of a population of many units with identical "analog" representations. If there is independent noise on each unit, then the population gives more information than any individual unit, yet information is coded by the intensity of the firing rather than the pattern. Nevertheless, figure 1 provides a useful intuitive distinction between the classical mathematical representation of variables and population-coded distributed representations.

Neural network concepts of distributed representations frequently

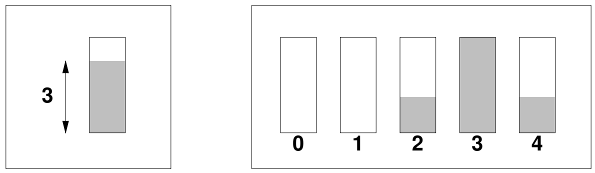


Figure 1: a. “Analog”, and b. “Population” coding of the number 3.

arise from observations of “receptive fields”. A neuron’s receptive field is usually defined as the set of values of sensory variables for which the neuron’s firing rate increases or decreases from background. For “simple cell” neurons in primary visual cortex, for instance, the receptive field is the area of visual space in which small flashes of light cause an observable change in firing rate (Maffei and Fiorentini 1973). The analysis of receptive fields usually focuses on a specific subset of experimentally controllable variables. For example, a visual receptive field may be defined in terms of its response to points of light or drifting gratings, but the stimulus color or depth may be held constant when this is not the variable of interest.

In the motor system, the concept of a receptive field is not as well defined. This occurs because cells’ behavior can be investigated both by extracellular recording techniques, as well as by microstimulation techniques. The consequences of microstimulation can be difficult to estimate, since *in vivo* a particular cell is unlikely to fire alone, but rather will most likely fire as part of a set of activations over a population. Thus many researchers have chosen to use extracellular recording during awake behavior as a method for determining the pattern of usage of motor system neurons. The primary difficulty with this approach lies in the inability to determine whether a given neuron has a causative role in production of the movements to which it responds.

I will ignore this difficulty here and study only the *response* properties of cells. In a behaving animal, the cell firing rate reflects the way in which the animal chooses to stimulate each cell in order to accomplish a desired task. Since this choice will in general depend upon sensory information, it is not surprising that many motor cortical cells have extensive sensory receptive fields. The type of question that is raised by investigating such response properties involves *what* is represented and *how well* it is represented. These questions are easily answered for analog representations, but can be significantly more difficult to answer clearly for distributed representations. The discussion below will attempt to address such issues.

Georgopolous and others (Georgopoulos *et al.* 1983, Georgopoulos *et al.* 1985, Schwartz *et al.* 1988) have performed recording studies in awake behaving monkeys, and have been able to characterize the response of motor cortical cells to changes in the direction of movement during upper-extremity reaching to lighted targets. Many cells have a broadly-tuned response with a single peak that is the "preferred" direction of movement. This defines the motor equivalent of a sensory receptive field. It is important to realize that, as for sensory receptive fields, motor receptive fields are investigated only for certain ranges of input variables. For example, a cell that is broadly-tuned for arm movement direction may also have significant modulations in firing rate as a function of endpoint force (Kalaska *et al.* 1989), but these modulations may not be included in the description of the cell's receptive field unless the experiment is specifically designed to test this. In general, such experiments assume that all other environmental variables are held fixed so that a cell's differential response to particular environmental variables can be investigated.

## 1.2 Linear Methods

Consider a particular form of distributed representation in which each unit is active for only a small contiguous range of input values. A radial basis function representation with local basis functions has this form, as does the unary (but not the binary) number system. Many of the motor receptive fields found in monkey primary motor cortex have this form as well (Schwartz *et al.* 1988). I will refer to such representations as having "local receptive fields". This type of receptive field supports the intuitive concept of each cell indicating how close the data is to its preferred value, and the population of cells as a whole can then "vote" to determine the true data value.

(Georgopoulos *et al.* 1988) showed that this technique can be used to extract the true direction of arm movement from the cortical representation. A "population vector"[1] $p$ is defined as

$$p = \sum c_i d_i \tag{1}$$

where $d_i$ is the (scalar) firing rate of each cell during a movement $m$, and $c_i$ is the (vector) direction of movement (in Cartesian coordinates) that produces the maximal response of cell $i$. This equation represents each cell voting for a particular direction of movement, in which all votes are weighted by the cell firing rate. The population code is "read-out" using a linear combination of the cell firing rates. Under fairly general conditions, the population vector $p$ will approximate the true direction of movement $m$ (Georgopoulos *et al.* 1988, Mussa-Ivaldi 1988, Sanger 1994).

Although the population vector method will correctly predict the direction of movement under these conditions, it is not able to give us much information about the coordinate system in which the population data is coded. If we can extract $p$ linearly from the cell firing rates, then it is as easy to extract $Tp$ for any linear transformation $T$. If the firing rates are a linear function of the movement direction $m$ so that $d_i = c_i^T m$ (as is the case for cosine-tuned cells), then the set of linear transformations of $m$ is the complete set of functions that can be computed linearly based on the population. However, if the firing rates include higher-order (non-cosine) terms, then it is possible to compute nonlinear transformations of the movement direction $m$. In other words, if $d_i = \sigma_i(m)$ for a set of nonlinear functions $\sigma_i$, then a linear combination of the form of equation 1 can calculate any nonlinear function in the Hilbert subspace spanned by the $\sigma_i$'s. This

[1] I use the term "population vector" to refer to the linear read-out of a population of cells using equation 1, and I use the term "population code" to refer to any collection of cells whose response represents one or more analog values

is the computational basis for artificial neural network "Radial Basis Function" techniques (Poggio and Girosi 1990, for example). We can generate an even larger set of possible functions by allowing nonlinear computations to be performed on the firing rates, rather than restricting read-out methods to linear combinations. So although we can say that "movement direction" is coded, it may not make sense to talk about a unique "coordinate system" for movements represented by distributed representations, since many such coordinate systems are possible. We can thus consider many distributed representations to be "coordinate-free" (Sanger 1994).

It is important to realize that the population vector is in general not the most efficient or accurate way to estimate the movement direction from the cell firing rates, and a variety of standard statistical and nonlinear methods exist that can perform better (Snippe 1996). For example, we can find the least-squares linear approximation that minimizes the expected error from the true movement $E[(p-m)^2]$ (Salinas and Abbott 1994). In this case, the solution is given by

$$p = Ad \tag{2}$$

where $p$ is the population vector, $d$ is a vector of all the cell firing rates, and $A$ is a $3 \times N$ matrix of coefficients calculated as

$$A = (C^T C)^{-1} C^T \tag{3}$$

where $C$ is the $N \times 3$ matrix whose rows are the preferred direction vectors $c_i$. The population vector method sets $A = C^T$, and it will work if and only if $C^T C = I$. Nonlinear best-estimation methods are also possible and may yield even better results, depending on the amount and type of noise in the measurements.

The measurement of preferred directions and population codes intentionally ignores or holds fixed many aspects of the environment that may contribute significantly to modulations in cell firing rates. This becomes particularly relevant when we consider that, in theory, 3 cells would be sufficient to specify the direction of hand movement, so that the approximately 600 direction-modulated cells found in (Schwartz *et al.* 1988) may well be performing other tasks at the same time. Despite the limitations of linear methods, the existence of population vector analysis demonstrates the sufficiency of population codes for the representation of certain features of movement.

A theoretical discussion of issues in population coded representations can be found in (Salinas and Abbott 1995). These authors show that for noisy neurons, the population vector reconstruction error decreases as the number of neurons in the population increases. They apply their results to multi-modal sensory information and show how sensory-motor maps can be learned adaptively. These issues will be discussed further below.

### 1.3 Difficulties with Population Code models

The above formulation leaves us with several difficult issues (Loeb *et al.* 1996). A question of central importance to brain researchers is the precise representation of signals within the central nervous system, and this question often takes the form of a discussion of coordinate systems. Much effort has been expended in attempts to determine, for example, whether arm movement direction is coded in the endpoint position Cartesian coordinates, joint angle coordinates, muscle length coordinates, body-centered polar coordinates, or visually-relevant coordinates. As shown above in section 1.2, this question can be ill-posed in the sense that all linearly-related coordinate systems are equivalent. Further, a nonlinear "read-out" of the population code may extract significantly more information or more accurate estimates than the linear method, so that analysis by linear methods can underestimate the true accuracy of the population-coded variables.

Although no single coordinate system can uniquely describe the internal representation, it may be possible to determine a "best" coordinate system. For example, suppose that a representation for hand position exists with many cells active for movements near the body but far fewer cells active for distant movements. In this hypothetical case, a body-centered polar coordinate system would be well-represented at all points in space, whereas a Cartesian coordinate system would have significantly increased error for points located far from the body. The best coordinate system would be determined both by the distribution of cell response properties, as well as by the read-out methods available to the system for interpreting the population code.

Another problem with the population code formulation is that it is difficult to understand the need for a large population of cells to encode a small number of relevant variables. Clearly, one can make arguments about reduction of noise sensitivity, but for linear reconstruction of three directions of movement it would be as efficient to have multiple copies of the same three cells as it is to have multiple different types of cell. We might presume that these cells accomplish a myriad of different tasks, but this removes the inherent interest of the population vector method. There is a need to formulate population codes in such a way as to capture the intuition that they allow precise parameter estimation from possibly imprecise neural firing data. For example, in (Salinas and Abbott 1995) the population vector approximation error decreases as the number of cells in the population increases. However, it is difficult to attribute this phenomenon to inherent properties of population codes, since not all such codes would exhibit this phenomenon. Consider a population of two cells with triangular receptive field profiles $\sigma_1(x) = 1 - x$ for $x < 1$ and $\sigma_2(x) = x$ for $x > 0$. Then any affine function $f(x) = mx + b$ can be easily computed as $b\sigma_1(x) + (m+b)\sigma_2(x)$. In particular, the identity $f(x) = x$ can be exactly computed using only these two cells. No further increase in accuracy would occur with more than two cells.

If the cells are noisy, then making $N$ copies of each (not necessarily spatially displaced) yields a reduction of noise variance proportional to $1/N$ according to convergence of the sample mean. If the cell receptive fields are curved so that $\sigma_1(x) + \sigma_2(x)$ is not everywhere affine, then it may not be possible to approximate linear functions well (although note that certain curved functions such as $\sin^2(x)$ and $\cos^2(x)$ do have the property that linear functions can be exactly approximated). It is likely that the decrease in error seen in (Salinas and Abbott 1995) is due to a combination of noise averaging and spatial interpolation for receptive fields that do not add to linearity.

Related to the question of the effect of increasing numbers of cells is the significance of broadly-tuned receptive fields. For movement variables, intuition might suggest that the more broadly tuned a neuron is, the "fuzzier" its data and the more dependent we are on using a population of such cells to extract accurate data. But consider the following situation: Suppose we have a single movement variable (say, movement amplitude) that is coded using only two broadly-tuned cells. If we know the exact firing rate of these cells (by waiting a few seconds and counting spikes), there is no noise, and we know the exact shapes of the receptive fields, then each of the two cells will have exactly that firing rate for only a very small number of possible inputs. The combination of the two cells determines the input value exactly. If the estimates of firing rates are noisy, we need only gather information for a longer time or from a large set of identical cells, to average out the noise and recover a good estimate of the input. No population vector needs to be computed, and the shape of the tuning curves has only a minimal effect on the accuracy of computation.

Central to many neural network models is the assumption that it is firing *rate* that carries the neural information. Given the vast number of different neuronal architectures, it is quite possible that such an assumption is valid for certain brain regions. However, for infrequently-spiking neurons the computation of spike rates may require a very long sample time. Even for neurons spiking at 100 spikes/second it would require 0.1 seconds to estimate the true firing rate to within 10%, and 1 second to estimate to within 1%. A behaving animal would have difficulty functioning in an active environment with this type of delay. It is possible to use a "pipelined" architecture in which the firing rate carries the information but is never computed internally. For instance, to multiply a firing rate by 0.5 it would only be necessary to block every second spike, without any need to compute the true firing rate in advance. It might also be possible to use the inter-spike interval as a measure of "instantaneous" firing rate, assuming a method for measuring this interval accurately. In the single-spike model presented below, I will use the approach of calculating based upon the spike data in a large population of cells, so that in any small time interval although most cells do not fire, at least some will. This allows good estimates to be calculated rapidly.

### 1.4 Goals of the Probability Model

In the discussion below, I propose a new way of interpreting neural spike trains that seeks to address some of these issues. The central idea is that if a neuron fires *probabilistically* depending on the value of input data, then every time the neuron is firing we can obtain a probability distribution for the input. In this way, neural spikes can be interpreted as modifying a prior distribution for the input, and neural spike rates can be interpreted as giving approximate values of input probabilities. I show that computations can be easily performed that allow probability information to be propagated throughout a network of neurons, in much the same way that spike rates can be propagated. This method can be used to re-phrase many commonly accepted methods for describing neural function.

The power of the probabilistic model is that we can answer questions such as

1. How can the system extract *instantaneous* information from a population of firing and non-firing cells?

2. How can nonlinear computations be learned from examples?

3. How is uncertainty represented internally, and how is it used? Are broadly-tuned response functions more uncertain than narrowly-tuned ones?

4. How can information from multiple sensory modalities be combined, and how can multiple motor modalities be coordinated?

5. How can the system optimize its use of cellular resources? What rules might govern use-dependent changes in cortical maps?

6. How well are different variables represented, and what is the "best" coordinate frame for representing internal variables?

## 2 Probability Model

The probability model presented here follows the development originally presented in (Sanger 1996). Consider a scalar variable $x$ that represents some measurable external quantity, and a set of cells $i$ that are modulated by changes in $x$. I model each cell by a positive receptive field function $\sigma_i(x)$ that modulates the rate of a Poisson process spike train. This is clearly an oversimplification, and does not take into account such important features as the refractory period of neuronal firing, or the possibility of inhibitory regions within the receptive field. However, it is an adequate model for the development which follows, and can be extended at the expense of significant complexity of notation. Note that each cell's receptive field may have the same shape but be at a different location (so that $\sigma_i(x) = \sigma(x + m_i)$), or all the receptive field shapes may be different (and perhaps at the same location).

The probability of firing $n$ times during a time interval $\Delta t$ in response to any fixed input $x$ is given by

$$\text{Poisson}(\sigma_i(x)\Delta t, n) = \frac{(\sigma_i(x)\Delta t)^n e^{-\sigma_i(x)\Delta t}}{n!} \tag{4}$$

where Poisson($\sigma_i(x)$) indicates a Poisson process with rate $\lambda = \sigma_i(x)$. The average firing rate $\sigma_i(x)$ will be the "response" of the receptive field at $x$. This is the same as in the standard formulation in terms of average firing rates. If we examine the system during a very short time-window for small $\Delta t$, then the probability of a single spike within time $\Delta t$ is approximated by

$$P[\text{spike}_i|x] \approx \sigma_i(x)\Delta t \tag{5}$$

### 2.1 Posterior distribution for a single spike

We can use Bayes' law to find the posterior distribution for the input $x$ given that a single spike occurred within a small time-window $\Delta t$:

$$P[x|\text{spike}_i] = \left(\frac{\sigma_i(x)\Delta t}{P[\text{spike}_i]}\right) P[x] \tag{6}$$

and note that for small $\Delta t$

$$P[\text{spike}_i] = \Delta t \int \sigma_i(x) dPx \tag{7}$$

so that

$$P[x|\text{spike}_i] = \left( \frac{\sigma_i(x)}{\int \sigma_i(x) dPx} \right) P[x] \tag{8}$$

I interpret this to mean that the occurrence of a spike in cell $i$ leads to a conditional probability density for the input that is equal to cell $i$'s receptive field shape normalized to have unit integral.

Note that the shape of $\sigma_i$ is arbitrary, and can be constructed to take into account many different possible concepts of distributed encoding. For instance, if $\sigma_i(x) = x$ (for bounded $x$) then this yields analog coding, since the rate is directly proportional to the input value. If $\sigma_i(x) = \delta(x - x_i)$ then we obtain "place coding" that is specific for a particular set of discrete values $x_i$.

Equation 8 is the foundation for the development which follows. It encapsulates the intuition that a narrow receptive field gives more detailed information about the input, since in such a case the occurrence of a spike would limit the possible input values to lie within the narrow receptive field. This result also shows how information can be read out "instantaneously" from single neurons based on single-spike events. If we calculate the number of spikes occurring within a much longer time interval $\Delta t$ so that $\sigma_i(x)\Delta t$ is large, and if the value of $x$ remains constant over this interval, then the Poisson distribution in equation 4 approaches a normal distribution with mean $\sigma_i(x)$ and variance $\sigma_i(x)$. If we calculate the number of spikes over successive long intervals of length $\Delta t$ and average $N$ times, then the variance will decrease by $1/N$ and we can make accurate estimates of the possible values of $x$ as the inverse image of the firing rate $\sigma_i^{-1}(x)$.

### 2.2 Posterior distribution for no spike

The probability that cell $i$ does not fire within time $\Delta t$ is given by

$$\text{Poisson}(\sigma_i(x)\Delta t, 0) = e^{-\sigma_i(x)\Delta t} \tag{9}$$

so that

$$P[x|\text{nospike}_i] = \left( \frac{e^{-\sigma_i(x)\Delta t}}{P[\text{nospike}_i]} \right) P[x] \tag{10}$$

where

$$P[\text{nospike}_i] = \int e^{-\sigma_i(x)\Delta t} dPx \tag{11}$$

which can be difficult to calculate explicitly, but also independent of whether or not cell $i$ fires. We can now calculate the posterior distribution as

$$P[x|\text{nospike}_i] = \left( \frac{e^{-\sigma_i(x)\Delta t}}{\int e^{-\sigma_i(x)\Delta t} dPx} \right) P[x] \tag{12}$$

The first-order approximation for small $\Delta t$ is given by

$$P[x|\text{nospike}_i] \approx \left( \frac{1 - \Delta t \sigma_i(x)}{1 - \Delta t \int \sigma_i(x) dPx} \right) P[x] \approx P[x] \tag{13}$$

so for small $\Delta t$ this will have only a minimal effect on the posterior probability distribution for $x$.

### 2.3 Posterior distribution for multiple spikes

Now, assume that cell firing rates for different cells are conditionally independent given the input $x$, so that

$$P[\text{spike}_i, \text{spike}_j|x] = P[\text{spike}_i|x] P[\text{spike}_j|x] \tag{14}$$

Then from Bayes' law we have

$$\begin{aligned} P[x|\text{spike}_i, \text{spike}_j] &= \left( \frac{P[\text{spike}_i|x] P[\text{spike}_j|x]}{P[\text{spike}_i, \text{spike}_j]} \right) P[x] && (15) \\ &\approx \left( \frac{\sigma_i(x)\sigma_j(x)}{\int \sigma_i(x)\sigma_j(x) dPx} \right) P[x] && (16) \end{aligned}$$

where the approximation holds for small $\Delta t$. Two cells with identical receptive fields that fire will multiply their receptive fields together and may produce a "sharper" distribution for the input. If the receptive fields are different, the input will lie within the product of the receptive fields, or roughly their intersection. Note that equation 16 holds for any two spikes occurring within the same time interval $\Delta t$, whether they occur exactly simultaneously or not.

The distribution for one spike that fires within $\Delta t$ and one that does not is proportional to (with normalization terms omitted)

$$P[x|\text{spike}_i, \text{nospike}_j] \sim \sigma_i(x) e^{-\Delta t \sigma_j(x)} P[x] \tag{17}$$

The generalization to multiple spikes over a population is straightforward. Because of conditional independence, the posterior probability distribution for $x$ given a population of cells is proportional to (Sanger 1996)

$$P[x|\text{population}] \sim P[x] e^{-\Delta t \sum_j \sigma_j(x)} \prod_{i \in \text{spike}} \sigma_i(x) \tag{18}$$

where a normalization term has been omitted. This can also be written

$$P[x|\text{population}] \sim \left( P[x] \prod_j e^{-\Delta t \sigma_j(x)} \right) \prod_{i \in \text{spike}} \sigma_i(x) \tag{19}$$

The first product term gives a "lateral inhibition" such that neurons which are not firing reduce the probability in a local region. This term takes into account the possibility for an uneven distribution of cell types or receptive fields over the possible values of $x$. The second product term, for centrally peaked receptive fields $\sigma$, gives local excitation (where $\sigma(x)$ is large) and long-range inhibition (where $\sigma(x)$ is small. Notice that the first term is independent of which cells are firing. This simplifies the population "read-out" considerably, since we need only multiply the receptive field shapes of the firing neurons (and normalize) in order to determine the posterior probability distribution.

### 2.4 Interpretation

The development above shows that we can consider a population code to represent implicitly the entire conditional density of the input data, rather than just a single value. In this way, it can indicate a "confidence" in the estimate of the underlying variable. A similar idea (with a different read-out method) has been proposed by (Anderson 1995).

Multiplicative combination of receptive fields as in equation 18 is unusual in population code analyses, but it encapsulates the intuition that increasing the number of cells leads to increasingly accurate estimates of the input, even if the cells have broadly-tuned receptive fields. It also shows how information can be extracted rapidly from a population by measuring the single spikes occurring within a very short time interval. And it shows that it is necessary to have a large population of cells in order to have a sufficient number firing at any time that we can accurately predict the input data or represent the movement parameters.

Given the posterior distribution for $x$, estimation of the true value of $x$ can be performed using Maximum a Posteriori (MAP) estimation by choosing the peak of the posterior distribution. Alternatively, a Minimum Mean Squared Error (MMSE) criterion could be used by choosing the mean of the posterior distribution. Note that the posterior distribution may be multi-lobed if there is uncertainty as to the range of possible values for $x$. Such a case would represent the existence of several discrete possible input values.

The posterior distribution for $x$ is implicit in the pattern of spike activity over the population. Many different spike patterns may lead to similar posterior distributions, and similar values of $x$ may lead to different spike patterns. The need for multiple cells arises from the relatively infrequent firing of each cell. If cells fired sufficiently fast that many spikes fall within any small time interval $\Delta t$, then average spike rate methods could be used. It is possible to compute posterior distributions for the input given a set of computed average firing rates, and (Anderson 1995) has provided a clear and thorough development of this technique. He shows that the posterior distribution is given as a linear combination of the average firing rates. This case is discussed further below.

Suppose we observe for a very long time $\Delta t$. Then the posterior probability of $x$ is

$$\frac{P[x]}{\text{normalization}} e^{-\Delta t \sum_i \sigma_i(x)} \prod_i \sigma_i^{N_i}(x) \tag{20}$$

where $N_i$ is the number of times that neuron $i$ fired in time interval $\Delta t$. If neuron $k$ fires almost continuously, and significantly more frequently than any other (perhaps because $x$ remains constant), so that $N_i/N_k \to 0$ for all $i \neq k$, then after normalization, equation 20 approximates

$$\frac{\sigma_k^{N_k}(x)}{\text{normalization}} \tag{21}$$

which will converge to a dirac delta function centered at the peak of $\sigma_k$. This is an unrealistic situation in general, since most of the time neuron $k$ will be silent and the exponential term will prevent it from approaching a narrow distribution. But we can still think of each cell as representing the "belief" that the sensory or motor data is located at the peak of its receptive field, since if we observe this cell for long enough and it fires significantly more than any other cell, then the data will indeed be confined to this value.

This form of multiplicative read-out of the population code addresses several of the problems noted above in section 1.3. Since we extract the true *a posteriori* distribution for $x$ given the population, we can use standard statistical methods to find the minimum-error estimate of $x$. We can represent the case of multiple discrete possible values, if the posterior distribution is bimodal. This case is particularly troublesome for linear read-out systems, since the resulting value would be the average of the two peaks' centers. It is clear that a large population of cells is needed in order to allow accurate estimates of $x$ to be computed in short amounts of time $\Delta t$. We will see below that it is important to have many different types of receptive fields, and that the fields need to be broad enough so that all possible input values lead to a significant number of cells firing. These results are a direct result of using the probability interpretation proposed here.

### 2.5 Combination of Cross-modal information

Essentially identical results hold for estimation of cross-modal or multi-dimensional information. In this case, we let $x$ be a vector quantity, and equation 18 continues to hold. The receptive fields $\sigma_i(x)$ are defined to represent the cell response to all relevant input information. If cell $i$ is not modulated by a dimension $x_k$ of the input, then $\sigma_i(x)$ will be a constant as that dimension varies, and the firing rate of cell $i$ will affect the marginal conditional distribution of $x_k$ only if $\sigma_i(x)$ and $x_k$ are not independent with respect to $P[x]$.

For example, suppose that cell $i$ responds only to visual information and not to auditory information. If a tiger appears in the field of view and growls, then the conditional distribution for auditory information will change as a function of the visual information. Computationally, this occurs because the marginal distribution for the auditory data is found by integrating the product $\sigma_i(x)P[x]$, and the natural correlation between visual and auditory information that is encoded in $P[x]$ yields a nonconstant function of auditory space. In this way, the visual data from cell $i$ yields a posterior probability distribution for auditory space that can be combined (by multiplication) with true auditory data from sound-responsive cells to produce an accurate estimate of the posterior probability of noise production by the tiger.

This type of calculation requires knowledge of the prior distribution $P[x]$ which can be obtained only through learning in the relevant environment. In the above example, if a tiger has never been seen (and heard) before, then there is no way to predict that a tiger in the field of view will lead to generation of a growl.

## 3 Time-averaged Spike Rate

As mentioned previously, many common interpretations of neural codes make use of an average spike rate as the essential mode of information transmission in neurons. We can perform a similar analysis on the probabilistic neurons defined above. From equation 4, the average firing rate for neuron $i$ is given by $\sigma_i(x)$. Following an analysis in (Anderson 1995), consider the current state of the input $x$ to be represented by a probability distribution such that

$$P[x(t)] = \delta(x - \bar{x}(t)) \qquad (22)$$

Then the average firing rate of each neuron at time $t$ will be given by $\sigma_i(\bar{x}(t))$. There will exist a set of "contravariant" functions $\tau_i(x)$ such that

$$P[x(t)] = \sum_i \tau_i(x)\sigma(\bar{x}(t)) \tag{23}$$

and the complete form of $P[x(t)]$ is implicit in the average firing rates of the population. Of course, $P[x(t)]$ is simply a delta function in this case, since it represents the fixed and known state of the experimental system, and in the absence of noise we could simply (piecewise) invert $\sigma_i$ to obtain an estimate of $x$ from the firing rates.

Now, suppose that $x$ is time-varying and has a sample distribution $P[x]$ that is smooth. The long-term average firing rate of neuron $i$ will be given by

$$\bar{\sigma}_i = \int \sigma_i(x)dPx = \int \sigma_i(x)P[x]dx \tag{24}$$

The contravariant functions $\tau_i(x)$ can be used to reconstruct $P[x]$ from the neuron average firing rates by using

$$P[x] = \sum_i \tau_i(x)\bar{\sigma}_i \tag{25}$$

The distinction between equation 23 and equation 25 is that the former is calculating short-time average firing rates for a single (approximately constant) value of the input state $x$, whereas the latter is calculating long-time average firing rates for many different values of the input state $x$. In the former case we estimate the exact value of the input variable at time $t$, whereas in the latter case we estimate the probability distribution for $x$ at all times. In both cases, the relevant information is implicit in the neural population average firing rates and can be extracted using the contravariant functions $\tau_i(x)$.

This method of density estimation is often referred to as "kernel density estimation", and a detailed treatment of several important issues is given in (Anderson 1995). The work here differs in that the underlying model is based on a Poisson-process model of neural firing that generates the probabilistic properties of the neurons as a direct consequence of this spike firing model. In (Anderson 1995), the hypothesis is made that neurons code probability density functions, and the consequences of this hypothesis are analyzed without reference to how such a coding might come about. In the time-averaged case, the two analyses give essentially the same results, although without a detailed spike-process model it is impossible to do short-time analyses such as those shown in section 2.

### 3.1 Computation with Population Codes

Consider a nonlinear smooth function between two variables $y$ and $x$ given by $y = f(x)$. If $x$ is a random variable with probability density $P_x[x]$, then this induces a probability density $P_y[y]$ and we seek the functional that maps $P_x$ into $P_y$. This immediately yields
**The Density Mapping Theorem:** *For any smooth function $y = f(x)$ there exists a piecewise linear transformation between probability representations for $x$ and $y$ given by*

$$P_y[y] = \int K(x,y) P_x[x] dx \tag{26}$$

*where $K(x,y) = \delta(y - f(x)) f'(x)^{-1}$ for each monotonic segment of $f$.*

This remarkably simple result shows that any nonlinear function between two scalar analog variables has a representation as a linear functional between their probability densities. In terms of population codes, this provides an important computational property since only linear operations will be needed in order to perform nonlinear operations on population code representations of probability densities. Essentially the same result was shown independently in (Anderson 1995), where the elements of the kernel $K(x,y)$ are identified with the conditional probabilities $P[y|x]$. Note that the kernel function performs essentially a "shift" operation in which particular input neurons are mapped to different output neurons with only minimal processing using local linear functions.

The theorem only applies to the original continuous densities $P_x$ and $P_y$, not to the population codes $\sigma_i(x)$ and $\phi_j(y)$. The optimal linear transformation between population codes can be described by

$\phi(y) \approx A\sigma(x)$ where the matrix $A$ is given by

$$A = \int \int \phi(y) K(y,x) \tau(x)^T dy dx, \tag{27}$$

$K(y,x)$ is as above, $\phi(y)$ and $\sigma(x)$ are column vectors, and $\tau(x)$ is the vector function that minimizes the Hilbert space norm $\|\delta(x' - x) - \tau(x')^T \sigma(x)\|$. ($\tau$ will be the contravariant function of $\sigma$.)

Given population codes $\phi_j(y)$ and $\sigma_i(x)$ we can easily learn the linear mapping between them using either direct computation (via matrix pseudo-inversion) or iterative learning algorithms. Suitable algorithms include the Widrow-Hoff delta rule (Widrow and Hoff 1960) or recursive least squares estimation (Ljung and Soderstrom 1983). A Hebbian-like algorithm is demonstrated in (Salinas and Abbott 1995).

### 3.2 Multi-Dimensional Computation

If we attempt to compute a function of two variables given by $z = f(x,y)$, then the density mapping theorem gives

$$P[z] = \int K(z,x,y) P[x,y] dx dy \tag{28}$$

where

$$K(z,x,y) = \delta(z - f(x,y)) / |J_f(x,y)| \tag{29}$$

and $|J_f(x,y)|$ is the determinant of the Jacobian matrix of $f$ at $(x,y)$. Although the linearity of the result holds, it is clear that the computational difficulties will increase exponentially with increasing dimensionality of the input space. In particular, to approximate the continuous kernel $K$ using $n$ values for each of $d$ dimensions we will need to store $n^d$ numbers which rapidly becomes prohibitive, both for learning time and the required storage capacity. While these expressions simplify considerably in the case that $f$ is separable (so that $f(x,y) = f_1(x) f_2(y)$) and $x$ and $y$ are independent ($P[x,y] = P[x]P[y]$), this case can not be counted on for all situations.

A particular learning algorithm that performs well under these circumstances is the "Basis-Function Tree" developed in (Sanger 1991c, Sanger 1991b, Sanger 1991a). This algorithm makes use of separable high-dimensional receptive fields in the input. An example would be a spherical Gaussian such as

$$\sigma(x) = e^{-x^T \Lambda_i x/2} \tag{30}$$

where $\Lambda_i$ is a positive diagonal covariance matrix, and $x$ is a vector. The algorithm takes advantage of the fact that when receptive fields are separable, they can be written as

$$\sigma_i(x) = \prod \sigma_i^k(x_k) \tag{31}$$

where the $x_k$ are the individual scalar components of the vector $x$. In the Gaussian case, each $\sigma_i^k$ is a one-dimensional Gaussian. The set of functions that are linear combinations of the high-dimensional receptive fields in equation 30

$$\sum_i \alpha_i \sigma_i(x) \tag{32}$$

is exactly equivalent to the set of polynomials over the one-dimensional receptive fields $\sigma_i^k$

$$\sum_{i,n} \alpha_{i,n} (\sigma_i^k(x_k))^n \tag{33}$$

where the $\alpha$'s are fixed coefficients. The basis-function tree learning algorithm attempts to find a sparse polynomial approximation to the full-complexity polynomial, and in so doing can reduce the complexity from exponential to (on average) linear. This result generalizes a suggestion in (Anderson 1995) that multiplicative interactions between receptive fields may have an important computational role. We have already seen other examples of multiplication in the derivation of the spike model for single-dimensional population codes.

In the case presented here, interactions are extended from simple multiplication to the full set of polynomials which can be shown to be the required set of functions for computing high-dimensional functions from one-dimensional population codes. This allows sensor fusion and the combination of information across multiple sensory modalities. In other words, if each scalar input variable $x_k$ is represented by a separate population code $\sigma_i^k(x_k)$, then these individual codes can be combined to compute desired functions by forming polynomials of the cell firing rates. From equation 18 we also know that the read-outs for the individual populations involve polynomials over the set of receptive fields, so the multiplicative interactions in polynomial computations provide a general computational framework. It is thus particularly relevant that simple adaptive algorithms exist for finding such polynomials (Sanger 1991c, Sanger 1991b, Sanger 1991a).

## 4 Optimal Population Codes

Given an input probability distribution $P[x]$, the question arises of whether it is possible to design a population code that is "optimal" for representing $x$, in some sense. This question is important to allow comparison with existing neural codes so that we can determine whether these codes are in fact optimal or whether there are additional constraints that prevent optimality. In terms of modelling, it would be worthwhile to design adaptive algorithms that can generate the optimal codes, as well as respond to changes in $P[x]$ over time. Such models might be able to explain the plasticity in adult cortical maps that occurs in response to changes in the sensory environment or motor plant (Donoghue and Sanes 1987, Donoghue and Sanes 1988, Sanes *et al.* 1988, Kaas *et al.* 1983, Merzenich *et al.* 1983a, Merzenich *et al.* 1983b, Allard *et al.* 1991, Jenkins *et al.* 1990).

Algorithms for plasticity operate at a relatively slower time-scale than that needed for sensory interpretation or motor action, so it is possible to use either the single-spike or the time-averaged formulation to develop optimal codes. The time-averaged form is considerably simpler and will be presented first, which will help to motivate the single-spike development.

### 4.1 Maximum Mutual Information for Time-Averaged Spike Rates

We seek to find a population code for a variable $y = f(x)$ such that the mutual information between $y$ and (bounded) $x$ is maximized, according to the "infomax" principle first suggested by Linsker (1988). In the noiseless case, we cannot use this principle, since $y$ will be exactly determined by $x$ with no residual uncertainty. We therefore need to introduce a noise model. Suppose that the true value of a variable is given by $x$, but we are only able to measure $\hat{x} = x + \epsilon$ where $\epsilon$ is a noise variable independent of $x$. If we observe many values of $\hat{x}$ for fixed $x$, then we can construct the conditional probability density $P[\hat{x}|x]$.

Using the development in (Anderson 1995) and the description above, we can interpret the average firing rates in a population code for a particular value of $x$ as a representation of the conditional probability distribution $P[\hat{x}|x]$, using equation 25:

$$P[\hat{x}|x] = \sum_i \tau_i(\hat{x}) E_{\hat{x}}[\sigma_i(\hat{x})] \tag{34}$$

where $E[\sigma_i(\hat{x})]$ gives the expected value of the firing rate as a response to many values of $\hat{x}$, and $\tau_i(\hat{x})$ is a set of contravariant functions that allows reconstruction of the density from the outputs of $\sigma_i$. We now have

$$P[x|\hat{x}] = \frac{P[\hat{x}|x]P[x]}{P[\hat{x}]} \tag{35}$$

by Bayes' law.

We seek to choose a population code for the variable $y$ which maximizes the mutual information $I(Y;X)$ between $y$ and $x$. Note that this is not the same as maximizing the mutual information between the population code itself (the firing rates) and the input variable $x$.

From section 3.1 and (Anderson 1995) we can identify the elements of the kernel $K(y,x)$ that maps $x$ into $y$ with the terms $P[y|x]$ if the populations are interpreted as probabilities. This allows us to apply the methods of information theory. $P[y|\hat{x}]$ is available by observing the average firing rates over the $y$ population so long as the true input $x$ does not change. If we observe over a very long time so that we see multiple values of $x$, then the average firing rates will represent $P[y]$. To extract these densities, we need to use the contravariant functions according to equation 25.

The mutual information between $y$ and $\hat{x}$ is given by

$$I(Y;\hat{X}) = H(Y) - H(Y|\hat{X}) \tag{36}$$

where $H$ is the entropy of the density of its argument. We can write

$$H(Y) = -\int P[y]\log(P[y])dy \tag{37}$$

and

$$H(Y|\hat{X}) = -\int P[y|\hat{x}]\log(P[y|\hat{x}])dydP[\hat{x}] \tag{38}$$

and note that $P[y]$ is available from the long-term average firing rates over the population for $y$ (for many values of $x$), while $p[y|\hat{x}]$ is available as the short-time averaged firing rates (for fixed values of $x$). Equation 37 is maximized when $P[y]$ is constant, and equation 38 is minimized when $P[y|\hat{x}]$ is as sharply peaked as possible for all values of $\hat{x}$. The combination of these two conditions will maximize the mutual information 36. We can interpret this to mean that for any input we want only a few neurons in the population to be active, so that we can unambiguously identify the stimulus or desired movement. On the other hand, over the set of all stimuli or movements, we want to use all the neurons approximately equally in order to maximize our use of available resources.

This formulation makes use of the probability encoding in the population to provide easy access to the distributions for $P[y]$ and $P[y|\hat{x}]$. It requires the assumption of noisy inputs in order to use maximum information methods. Under these assumptions, it gives an easy method for finding the variable $y$ and representing it with a neural population.

However, it does not tell us whether the population code itself is optimal. In particular, we are faced with the difficulty of why we do not simply allow the firing rate of a single neuron to be equal to $y$, once we have found it. Surely this is a better use of available resources! So although the conclusion seems to be appropriate for a good use of

Figure 2: Synaptic diffusion algorithm structure. The input neurons $\sigma$ and output neurons $\phi$ are connected by a much larger number of synapses $\psi$ whose strengths and locations are controlled by the variable matrix $B$. Synapses "diffuse" left and right in order to equalize the average firing rate of the output neurons.

resources, it is not clear from the development why a population code is needed at all. A similar difficulty occurred above in section 1.3 with the initial development of population codes as probability densities, in that the time-averaged form did not clearly demonstrate the need for more than two neurons. The single-spike formulation, however, showed that instantaneous estimation of population-coded parameters depends upon a large population of cells so that at any given time at least a few are likely to fire. Section 4.3 describes optimality arguments in terms of the single-spike model.

### 4.2 The Synaptic Diffusion Algorithm

We can use a form of "synaptic diffusion" to approximate the optimal time-averaged solution through an iterative algorithm. This algorithm is motivated by the fact that diffusion maximizes the entropy of a collection of movable particles (in this case, synapses). Consider the structure shown in figure 2. The sensory receptive fields are the func-

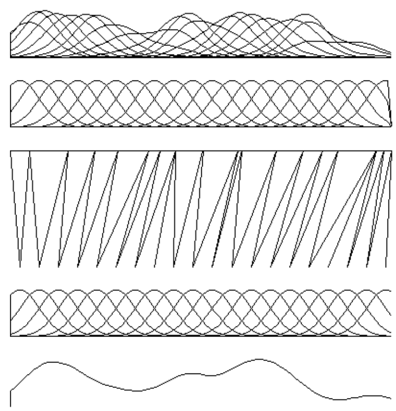


Figure 3: Synaptic diffusion algorithm results. The mapping input density is chosen randomly, and the learned mapping is computed iteratively.

tions $\sigma_i(x)$. These outputs are remapped onto a large vector $\psi = B\sigma$ via a nonsquare matrix $B$ whose column sums are all 1 and for which each row contains exactly one nonzero value. The number of rows $m$ of $B$ is much greater than the number of columns. This means that each input neuron maps onto several different output synapses, but that each output synapse is connected to only one input neuron. $B$ is chosen so that the expected values $E[\psi_k]$ are all equal, so the connections from $\sigma$ to $\psi$ spread out the activity of each $\sigma_i$ over a region of the vector $\psi$ proportional to the expected value $E[\sigma_i]$. A subsequent step of calculating the $\phi_j$'s as a smoothed and subsampled version of $\psi$ yields the new population code.

This structure corresponds to a set of output neurons $\phi_i$ each with multiple input synapses $\psi_k$, and a set of input neurons $\sigma_i$ each of which can lead to mutlple output synapses $\psi_k$. The number of synapses is much greater than the number of either input or output cells. Diffusion occurs among the synapses to spread out the activity evenly over each output cell $\phi_j$. The mapping from input to output is controlled by the adaptive matrix $B$, and the large number of synapses allows a smooth mapping to be computed.

The receptive field shapes for a randomly chosen input density are shown at the top of figure 3. Note that in terms of the original input coordinates the output receptive fields are wider in regions of low data density and narrower in regions of high data density. This is appropriate, since the narrower fields will convey more information about the input data and thus should be allocated to more likely regions. A similar result is shown in more detail in (Linsker 1989).

The algorithm for finding $B$ simply involves allowing the synapses from $\sigma$ to $\psi$ to "diffuse" down a gradient determined by the product of the local synaptic density $B_{ji}$ and the average input activity $E[\sigma_i]$. Suppose the number of output synapses $\psi$ is $m$, and the number of input synapses $\sigma$ is $n$, and $m = rn$ for some integer $r$. Then if a particular synapse connects from $\sigma_i$ to $\psi_j$, and $E[b_{j+1,i}\sigma_i] < E[b_{j,i}\sigma_i]$, the synapse will move to connect to the output $\psi_j$ in an attempt to equalize the output average firing rates. The use of diffusion will perform histogram equalization of the probability distribution over $\sigma$, as desired. A set of evenly-spaced receptive fields that uses $\psi$ as input will then yield a set of output units with equal variance. Note that this algorithm does not take the term $P[y|\hat{x}]$ into account, and thus converges to only an approximate solution. The one-dimensional example shown required 1.2 million diffusion steps (approximately 15 seconds on a 100Mhz 80486). Many other algorithms to solve this problem are possible and might perform faster.

The extension of this type of algorithm to high-dimensional input is theoretically straightforward but computationally difficult. Suppose that the input variable $x$ is a 2-dimensional vector. Then the optimal $y$ will also be 2-dimensional, with $P[y]$ a constant. If $y$ is bounded in a square region, then $P[y]$ is a separable function that can be written as $P_1[y_1]P_2[y_2]$ and thus the two components $y_1$ and $y_2$ of $y$ are independent. To achieve this, let $y_1 = f_1(x_1)$ and choose a monotonic $f_1$ such that $P[y_1]$ is constant. Now for each value of $x_1$ compute $P[x_2|x_1]$ and choose a function $f_2$ such that $y_2 = f_2(x_2, x_1)$ has $P[y_2|x_1] = P[y_2|y_1]$

constant. Then $P[y_2, y_1] = P[y_2|y_1]P[y_1]$ which is a constant as desired. The computational difficulty lies in the computation of $f_n(x_n, \ldots, x_1)$ when the number of dimensions $n$ becomes large. Future work will investigate efficient approximations for this procedure.

### 4.3 Maximum Mutual Information for the Single-Spike Model

The above optimality discussion showed an approximate method for using population codes to find a variable $y$ that had maximum mutual information with a noisy input variable. However, it did not determine whether or not the population code for $y$ was in any sense optimal. In particular, the same variable could be represented using many different types of collections of neurons, depending on how the value is read out.

Consider the state of the population over a very short time interval $\Delta t$ during which at most one (but probably none) of the neurons fires with significant probability. The number of likely states of the population is then $N + 1$ where $N$ is the number of neurons, and the extra 1 indicates that no neurons firing is an additional state. There is a small but nonzero Poisson-distributed probability that more than one neuron will fire during $\Delta t$, and I will ignore this possibility in the initial development below.

The mutual information between the population code and the variable $x$ is given by

$$I(S;X) = H[S] - H[S|X] \tag{39}$$

where the variable $S$ represents the state of firing or not-firing of all the neurons in the population. Since the population now has $N + 1$ discrete states, its entropy can be written

$$H[S] = -P[0] \log P[0] - \sum_{i=1}^{N} P[S_i] \log P[S_i] + \epsilon \tag{40}$$

where $S$ is the state of the population including all firing and non-firing neurons, $P[0]$ is the probability that no neuron fires, $S_i$ is the event that neuron $i$ fires, and $\epsilon$ is the contribution from terms due to multiple neurons firing during a chosen interval of length $\Delta t$. The conditional entropy can be written

$$H[S|X] = \int -P[0|x] \log P[0|x] - \sum_{i=1}^{N} P[S_i|x] \log P[S_i|x] + \epsilon(x) dP[x] \tag{41}$$

where $x$ is the current state of the input or desired movement.

Now, maximization of the mutual information (39) is accomplished by simultaneously maximizing (40) and minimizing (41). Equation 40 is maximized when $P[S_i]$ is equal for all neurons $i$. (Actually, if $P[S_i] = P[0]$ for all neurons, but this is in general not possible since the value of $P[0]$ varies with $\Delta t$.) This means that the probability of firing is equal for all neurons, and thus so is the average firing rate. Recall from equation 24 that the average rate of neuron $i$ is given by

$$\bar{\sigma}_i = \int \sigma_i(x) P[x] dx$$

For simplicity, let $S_0$ be the state of no neurons firing, so we can rewrite equation 41 as

$$\begin{aligned} H[S|X] &= -\int \sum_{i=0}^{N} (P[S_i|x] \log(P[S_i|x])) + \epsilon(x) dP[x] &(42)\\ &= -\int H[S|x] dP[x] &(43) \end{aligned}$$

which shows that we need to minimize the number of different neurons with significant probability of firing for each value of $x$. In other words, although we want all neurons to fire at approximately the same rate, we only want a few firing for any particular input value.

Substituting equation 5 ($P[S_i|X] \approx \sigma_i(x)\Delta t$) gives

$$\begin{aligned} H[S|X] &\approx -\int P[0|x] \log(P[0|x]) dP[x] \\ &\quad -\int \sum_{i=1}^{N} \sigma_i(x)\Delta t \log(\sigma_i(x)\Delta t) dP[x] &(44)\\ &= -\int P[x,0] \log\left(\frac{P[x,0]}{P[x]}\right) dx \end{aligned}$$

$$-\sum_{i=1}^{N}\int \sigma_i(x)\Delta t\log(\sigma_i(x)\Delta t)dP[x] \qquad (45)$$

The first term of equation 45 is the negative of the Kullback-Liebler distance from $P[x,0]$ to $P[x]$, and minimization of this term will thus maximize the distance between the distributions and ensure that the situation of no neurons firing is as unlikely as possible. Minimization of the second term minimizes the sum of the "spread" of the receptive fields of the neurons, ensuring that they respond to only a limited range of values $x$, particularly in regions where $P[x]$ is large. An alternate view is provided by equation 44, which shows that the second term is minimized when only a few of the neurons respond to any given value of $x$. This is important, since it means that we do not maximize the mutual information by having a large collection of identical neurons, but rather by having a set of varied neurons that respond differently to different inputs.

So maximization of mutual information can be accomplished by attempting simultaneously to

1. equalize the average firing rate of all neurons in the population code,

2. minimize the amount of time that no neuron is firing,

3. minimize the range of input values to which each neuron responds.

4. minimize the number of neurons that respond to any input value.

These goals capture a natural intuition about the efficient use of limited neural resources. They are derived directly from the use of single-spike information for estimation, and all uncertainty arises from the need to code continuous input data using a population of Poisson-firing neurons. Straightforward arguments arising from maximization of mutual information lead directly to optimality principles that describe the goals of an optimal neural coder in terms of the receptive fields of neurons in the population. This leads to "histogram equalization" of the probability density (Nadal and Parga 1994) and a large and varied population of cells.

The mutual information depends upon the distribution of the input data $P[x]$. In the laboratory, this is chosen by the experimenter. The "real" world imposes a specific choice of this distribution. An interesting consequence of these results is that we can test whether or not an animal has in fact optimized a distributed representation by computing the value of equation 39 for a hypothetical $P[x]$. If the value is low, it suggests that either our assumed prior $P[x]$ is wrong, or there are biological constraints preventing optimality.

## 5 Cortical Representations

In the investigation of biological population codes, one frequently seeks to identify the "coordinate system" in which some relevant behavioral parameter is coded. This question comes in two basic forms. In the first form, it is a question concerning what information is or is not present in a representation. For example, if cell firing rates vary with hand position but not with elbow position, then we can say that although hand position is clearly represented, joint angle is not.

The second form of the coordinate system question is the one to be investigated here. In this case, we know that a set of cells responds to a particular sensory or motor parameter, but we would like to determine the coordinate system that is used to code that parameter internally. For example, a set of cells in motor cortex that are known to respond differentially to different types of hand reaching movements (Georgopoulos *et al.* 1988, for example) could be hypothesized to code those movements in terms of the Cartesian coordinates of the hand, polar coordinates of the hand about the shoulder, muscle lengths for the arm, perspective projection on the retina of the hand position, or some other possible coordinate system.

Recent work (Sanger 1994) summarized in section 1.2 has shown that it is not possible to define a unique coordinate system for certain types of population-coded variables. In the current framework, this result can be summarized by noting that the density mapping theorem shows that different population codes can be mapped into each other using linear transformations that perform shift operations, so that it is relatively easy to change coordinate systems merely by re-labelling the values assigned to cell outputs.

Despite this result, it would seem plausible that the maximum-information criterion given above might allow us to define a "best" coordinate system. Choosing a coordinate system for a population code does not change the population's behavior, but it provides a method for interpreting the output. So, unlike the discussion above, information maximization will not indicate how to design the population (which is determined by the biology) but rather, it will indicate what interpretation gives the most information about the population (or conversely, about which interpretation the population gives the most information).

Clearly, the population gives the most information about itself. In other words, the best coordinate system is the set of neural firing rates themselves. However, this is unsatisfying in that we usually want to convert the population code into a few continuous variables that describe observed behavior such as the position or velocity of the hand. The concept of the "population vector" in equation 1 (Georgopoulos *et al.* 1988) is one method for extracting continuous variables from a population using a linear combination of time-averaged firing rates. Equations 8 and 18 give another alternative. Determination of the optimal code will depend on the details of which interpretation method is used.

## 5.1 Reading Population Codes

Following the probabilistic model for spiking neurons, I will use equation 18 to interpret the instantaneous activity of the population. The best estimate of $x$ is given (by MAP estimation) as the peak of the posterior distribution in equation 18. As seen in section 2.4, if a single cell fires sufficiently often, then the posterior distribution for the input will converge to a delta function at the peak of the cell's receptive field. Therefore, I will identify the value of $x$ at the peak of the receptive field

$\sigma_i(x)$ as the "value" assigned to cell $i$. This is similar to the idea of the "preferred direction" described in (Schwartz *et al.* 1988).

### 5.2 Population Codes and Coordinate Systems

There exist many coordinate systems in which an input or movement parameter $x$ could be measured. For example, we can describe a receptive field as $\sigma(x)$, $\sigma(\log x)$, or any other remapping of the input. This becomes more of an issue if the receptive field is a "two-dimensional" function of two variables, so that it could be $\sigma(x, y)$ (Cartesian coordinates), $\sigma(\sqrt(x^2 + y^2), \tan^{-1}(y/x))$ (polar coordinates), or some other function. In general, we can describe the receptive field as $\sigma(g(x))$ where $g(x)$ is an arbitrary invertible function that maps into a new coordinate system. The important thing to realize is that there is no "natural" $g(x)$. Any experiment implicitly determines a measurement system that may be polar, Cartesian, Spherical, or something else. The implicit coordinate system is the one that is used to determine the receptive field in the first place, by recording from cell $i$ while $g(x)$ is kept fixed at some value. The receptive field is described in terms of measurements in this coordinate system. For notation, I am using $g(x)$ to indicate the variability in the coordinate system mapping relative to a fixed variable $x$. This is not meant to imply that $x$ itself has some special meaning. $x$ is merely an arbitrary "default" coordinate system that allows $g(x)$ to be defined.

When we choose a coordinate system $g(x)$ to measure the input and then perform an experiment, we measure the value of the peak of $\sigma_i(g(x))$, so that neuron $i$ becomes labelled with the value of $g(x)$ at this peak. Conversely, if we assign a value $y_i$ to each neuron $i$ whose peak occurs at position $x_i$ in the "default" coordinate system, then that set of values determines the function $g(x)$ at a set of discrete points according to $y_i = g(x_i)$. So when we assign a label to the output of a given neuron we are really determining the value of the coordinate system mapping $g(x)$ at some point.

The danger that this causes is that we can assign any value we like to the neurons in the population. We will still be able to compute receptive fields and make MAP estimates of the input values, although these estimates will be computed within our arbitrarily-defined coordinate system. In the next section I show that there may be a "best" labelling that induces an "optimal" coordinate system for a given population.

### 5.3 The Optimal Coordinate System

Once again, I will use information maximization to induce a concept of "optimality":

**Definition:** *The "optimal" coordinate system $y = g(x)$ maximizes the mutual information $I(Y;S)$ between $y$ and the state $S$ of the population of neurons.*

The mutual information is given by

$$I(Y;S) = H[Y] - H[Y|S] \tag{46}$$

Going back to the short time interval $\Delta t$ case in which at most 1 neuron fires, we have from equation 8 that the posterior probability for $y$ given that neuron $i$ fired and no other neurons fired is

$$P[y|S_i] = P[y]\frac{\sigma_i(y)}{\int \sigma_i(y)dPy} \tag{47}$$

where $\sigma_i(y)$ is the receptive field plotted with respect to the variable $y$. Let $\bar{\sigma}_i(y) = \sigma_i(y)/\int \sigma_i(y)dPy$ be the normalized receptive field. Then

$$\begin{aligned} H[Y|S] &= \sum_i H[Y|S_i]P[S_i] && (48)\\ &= -\sum_i P[S_i]\int \bar{\sigma}_i(y)\log\bar{\sigma}_i(y)dPy && (49)\\ &= -\sum_i P[S_i]\int \bar{\sigma}_i(x)\log\bar{\sigma}_i(x)g'(x)dPx && (50)\\ &= -\int g'(x)\sum_i P[S_i]\bar{\sigma}_i(x)\log\bar{\sigma}_i(x)dPx && (51)\end{aligned}$$

where I have changed variables back to the default coordinate system. Similarly, we have

$$H[Y] = -\int P[y]\log P[y]dPy \tag{52}$$

$$= \quad -\int P[x] \log P[x] g'(x) dPx \tag{53}$$

so that we can write the mutual information as

$$\begin{aligned} I(Y;S) &= H[Y] - H[Y|S] \quad (54) \\ &= -\int P[x] \log P[x] g'(x) dx \\ &\quad + \int g'(x) \sum_i P[S_i] \bar{\sigma}_i(x) \log \bar{\sigma}_i(x) dPx \quad (55) \end{aligned}$$

$$= \int g'(x) \left( -P[x] \log P[x] + \sum_i P[S_i] \bar{\sigma}_i(x) \log \bar{\sigma}_i(x) \right) dPx \tag{56}$$

The term in parentheses is large when $P[x]$ is constant, while as few of the $\sigma_i$'s as possible are nonzero. This means that we want to obtain the most information about $x$ with the least uncertainty due to multiple neurons firing. As in section 4, $I(Y;S)$ is maximized when $g'(x)$ is large in regions where the term in parentheses is large. In other words, there are "more" values of $y$ wherever we will be able to determine $y$ accurately.

If $g'(x)$ is monotonic and restricted to be square integrable, then we can maximize $I(Y;S)$ for any arbitrary value of $\int g'(x)^2 dx$ by setting $g'(x)$ proportional to

$$-P[x] \log P[x] + \sum_i P[S_i] \bar{\sigma}_i(x) \log \bar{\sigma}_i(x) \tag{57}$$

The first term spreads out values of $y$ in regions where there is more uncertainty in the input value. The second term spreads out $y$ wherever the population tends to give unequivocal information about the value of $x$ (when a small number of very specific neurons have a high firing rate). Equation 57 gives us a constructive technique for finding the optimal mapping, and thus the optimal labelling for the cells in the population. Note that, unless $y$ is a circular variable (running 0 to $2\pi$), the scaling of $y$ is not specified in the sense that $\alpha y$ is as good for any fixed value of $\alpha$. And since only the derivative $g'$ is specified, we can add an arbitrary constant to obtain $\alpha y + \beta$.

For two-dimensional or higher receptive fields, set the determinant of the Jacobian to be equal to equation 57. Note that this may not be integrable; in particular there may be small neighborhoods that need to be excluded (such as near the origin in a polar coordinate system).

# 6 Conclusion

The re-interpretation of population codes as representing probability distributions allows a set of powerful techniques from probability theory to be applied to address current difficulties with population coding theory, including the need for multiple cells and cell-types and the definition of a best coordinate system. In particular, we can access mapping theorems on probability Hilbert spaces (The "Density Mapping Theorem"), as well as results from Information Theory. I propose that this re-interpretation allows the development of a computational theory of population codes within which we can formulate supervised and unsupervised learning algorithms, determine optimality, and interpret neurophysiological data.

This work leaves several unanswered questions. Probably one of the most important is how techniques for optimizing neural mappings can be found that operate efficiently on samples of data. The "synaptic diffusion" algorithm is only one possibility, and is only correct for the time-averaged spike rate case. In the single-spike case, it would be interesting to determine whether a recursive algorithm might lead to similar changes in cortical maps to those found by (Donoghue and Sanes 1987, Donoghue and Sanes 1988, Sanes *et al.* 1988) and others. It would also be important to determine how such an algorithm could handle high-dimensional or multi-modal information without succumbing to Bellman's "curse of dimensionality".

These results provide a possible framework for interpreting large population codes in biological neural systems. The probabilistic development allows the use of information theory and probability theory to encapsulate our natural intuition about the role of large collections of nerve cells. It also provides an explicit mechanism whereby a biological system can extract precise "real-time" information from the apparent imprecision of randomly spiking neurons.

## References


Allard T., Clark S. A., Jenkins W. M., Merzenich M. M., 1991, Reorganization of somatosensory area 3b representations in adult owl monkeys after digital syndactyly, *J. Neurophysiology*, 66(3):1048–1058.

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

Donoghue J. P., Sanes J. N., 1987, Peripheral nerve injury in developing rats reorganizes representation pattern in motor cortex, *Proc. Natl. Acad. Sci.*, 84:1123–1126.

Donoghue J. P., Sanes J. N., 1988, Organization of adult motor cortex representation patterns following neonatal forelimb nerve injury in rats, *J. Neuroscience*, 8(9):3221–3232.

Georgopoulos A. P., Caminiti R., Kalaska J. F., Massey J. T., 1983, Spatial coding of movement: A hypothesis concerning the coding of movement direction by motor cortical populations, *Exp. Brain Res. Suppl. 7*, pages 328–336.

Georgopoulos A. P., Kalaska J. F., Caminiti R., 1985, Relations between two-dimensional arm movements and Single-Cell discharge in motor cortex and area 5: Movement direction versus movement end point, *Exp. Brain Res. Suppl. 10*, pages 175–183.

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.

Jenkins W. M., Merzenich M. M., Ochs M. T., Allard T., Guic-Robles E., 1990, Functional reorganization of primary somatosensory cortex in adult owl monkeys after behaviorally controlled tactile stimulation, *J. Neurophysiology*, 63(1):82–104.

Kaas J. H., Merzenich M. M., Killackey H. P., 1983, The reorganization of somatosensory cortex following peripheral nerve damage in adult and developing mammals, *Annual Review of Neuroscience*, 6:325–56.

Kalaska J. F., Cohen D. A. D., Hyde M. L., Prud'homme M., 1989, A comparison of movement direction-Related versus load Direction-Related activity in primate motor cortex, using a two-dimensional reaching task, *J. Neurosci.*, 9(6):2080–2102.

Linsker R., 1988, Self-organization in a perceptual network, *Computer*, 21(3):105–117.

Linsker R., 1989, How to generate ordered maps by maximizing the mutual information between input and output signals, *Neural Computation*, 1:402–411.

Ljung L., Soderstrom T., 1983, *Theory and Practice of Recursive Identification*, MIT Press, Cambridge, MA.

Loeb G. E., Brown I. E., Scott S. H., 1996, Directional motor control, *TINS*, 19(4):137–138.

Maffei L., Fiorentini A., 1973, The visual cortex as a spatial frequency analyzer, *Vision Research*, 13:1255–1267.

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

Merzenich M. M., Kaas J. H., Wall J., Nelson R. J., Sur M., Felleman D., 1983a, Topographic reorganization of somatosensory cortical areas 3b and 1 in adult monkeys following restricted deafferentation, *Neuroscience*, 8(1):33–55.

Merzenich M. M., Kaas J. H., Wall J. T., Sur M., Nelson R. J., Felleman D. J., 1983b, Progression of change following median nerve section in the cortical representation of the hand in areas 3b and 1 in adult owl and squirrel monkeys, *Neuroscience*, 10(3):639–665.

Mussa-Ivaldi F. A., 1988, Do neurons in the motor cortex encode movement direction? An alternative hypothesis, *Neurosci. Lett.*, 91:106–111.

Nadal J.-P., Parga N., 1994, Nonlinear neurons in the low-noise limit: A factorial code maximizes information transfer, *Network*, 5:565–581.

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

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

Salinas E., Abbott L. F., 1995, Transfer of coded information from sensory to motor networks, *J. Neuroscience*, 15(10):6461–6474.

Sanes J. N., Suner S., Lando J. F., Donoghue J. P., 1988, Rapid reorganization of adult rat motor cortex somatic representation patterns after motor nerve injury, *Proc. Natl. Acad. Sci.*, 85.

Sanger T. D., 1991a, Basis-function trees as a generalization of local variable selection methods for function approximation, In Lippmann R. P., Moody J. E., Touretzky D. S., ed.s, *Advances in Neural Information Processing Systems 3*, pages 700–706, Morgan Kaufmann, Proc. NIPS'90, Denver CO.

Sanger T. D., 1991b, A tree-structured adaptive network for function approximation in high dimensional spaces, *IEEE Trans. Neural Networks*, 2(2):285–293.

Sanger T. D., 1991c, A tree-structured algorithm for reducing computation in networks with separable basis functions, *Neural Computation*, 3(1):67–78.

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*, in press.

Schwartz A. B., Kettner R. E., Georgopoulos A. P., 1988, Primate motor cortex and free arm movements to visual targets in three-dimensional space. I. Relations between single cell discharge and direction of movement, *J. Neurosci*, 8(8):2913–2927.

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

Snippe H. P., 1996, Parameter extraction from population codes: A critical assessment, *Neural Computation*, 8:511–529.

van Gelder T., 1991, What is the D in PDP? a survey of the concept of distribution, In *Philosophy and Connectionist Theory*, pages 33–59, Lawrence Erlbaum Associates, Hillsdale, NJ.

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