Computing linear approximations to nonlinear neuronal response

Melinda E. Koelling* and Duane Q. Nykamp†

July 24, 2008

Abstract

We present an approach to obtain nonlinear information about neuronal response by computing multiple linear approximations. By calculating local linear approximations centered around particular stimuli, one can obtain insight into stimulus features that drive the response of highly nonlinear neurons, such as neurons highly selective to a small set of stimuli. We implement this approach based on stimulus-spike correlation (i.e., reverse correlation or spike-triggered average) methods. We illustrate the benefits of these linear approximations with a simplified two-dimensional model and a model of an auditory neuron that is highly selective to particular features of a song.

1 Introduction

Many sensory neurons respond to stimuli in a highly nonlinear fashion. For example, neurons have been reported to be highly selective to particular classes of stimuli such as faces [4] or a bird's own song [14]. It is a challenge to understanding the origin of such properties because it is difficult to develop analysis techniques that give insight into strongly nonlinear response.

Many commonly used analysis techniques assume neuronal response is fundamentally linear. Such methods estimate the direction in stimulus space (the linear kernel) that leads to the greatest modulation in the neuron's response. For example, one can calculate the correlation between the stimulus and the neuron's spikes (often referred to as reverse correlation or the spike-triggered average) [3, 17, 6, 10], calculate the stimulus direction that maximizes mutual information between the neuron's response and projection of the stimulus onto the direction [29], or fit the response to generalized linear models [21]. A linear model predicts that a neuron's response is modulated by the projection of the stimulus in only one

*Department of Mathematics, Western Michigan University, Kalamazoo, MI 49008-5248 (melinda.koelling@wmich.edu)

†School of Mathematics, University of Minnesota, Minneapolis, MN 55455 (nykamp@math.umn.edu).

direction in stimulus space. Since any modification of the stimulus that leaves that projection unchanged will not alter the response of a linear model, linear models cannot capture the high degree of selectivity needed for faces or a bird's own song.

The manner in which one typically applies such linear techniques implicitly assumes a reference point near the origin in stimulus space. The analyses obtain linear kernels that point along directions emanating from the origin due to that fact that they employ stimulus ensembles that are centered near the origin. Typically, the stimulus ensembles are symmetric about the origin, or symmetric among the directions contained in the positive orthant (due to a constraint that all stimulus components must be positive). The result is a linear approximation of the neural response that is centered around the origin.

For a nonlinear neuronal response, the linear approximation will depend strongly on the location in stimulus space around which one builds the stimulus ensemble. A neuron could have multiple operating regimes where its behavior changes dramatically depending on the location in stimulus space. If so, analyses based around different locations in stimulus space would obtain different stimulus directions (i.e. linear kernels), reflecting the different stimulus features that most strongly modulate the neuron's response. Each linear approximation would be an equally valid description of the neuron's response, capturing the response of the neuron to stimuli near its reference point. The key point is that by computing *linear* approximations around different points, one could obtain further insight into the *nonlinear* behavior of the response.

Experiments have shown that linear kernels can change by considering different operating points. In songbirds, there are systematic differences between linear kernels computed from natural versus synthetic stimulus ensembles [30]. Further, these changes in linear kernels can provide information about nonlinearities. In cat inferior colliculus, the dependence of estimated linear kernels on the operating point revealed the nonlinearity in the processing of auditory signals [12].

In this paper, we outline a general framework for understanding how linear kernels depend on operating point and present a method to compute linear approximations of neuronal response that are not centered around the origin. We describe how to use these approximations to capture nonlinear effects. In section 2, we review the basic concept of a linear approximation from multivariable calculus. In section 3, we describe an algorithm for calculating such a linear approximation from measurements of neuronal spiking response to appropriately chosen stimuli. We demonstrate the results with a simple two-dimensional example in section 4 and with a more realistic high-dimensional example in section 5. We discuss the results in section 6.

2 The linear approximation

Imagine that one was probing the response of a neuron to a stimulus consisting of two adjacent bars. Let x and y represent the luminances of the bars relative to a background level in some arbitrary units. Assume for simplicity that the response of the function depends only on x and y, and let $f(x, y)$ represent the spiking rate of the neuron in response to the stimulus x and y.

Since f is a function of just two variables, we can easily plot it. Suppose that f was a

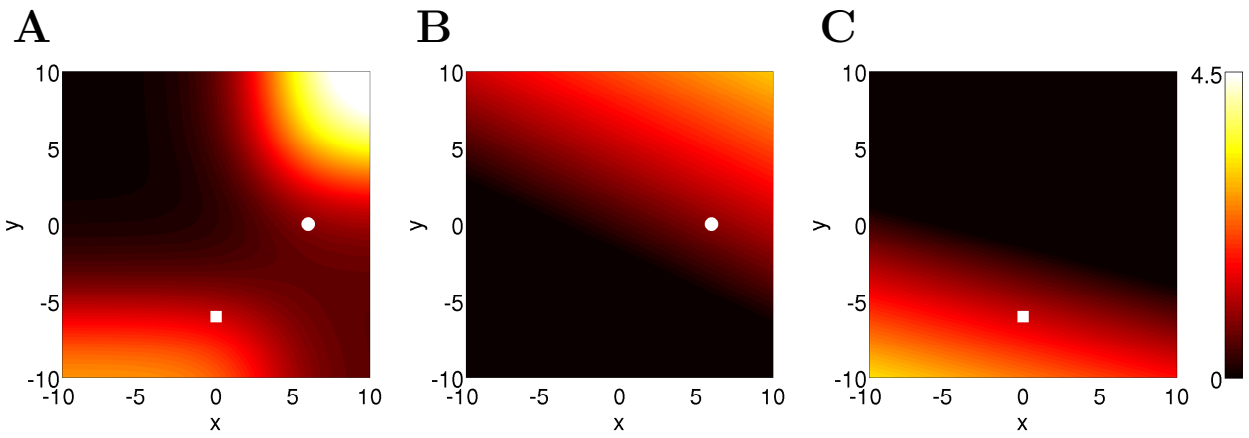

Figure 1: Plots of an example two-dimensional function $f(x, y)$ and its linear approximations. **A**. The function $f(x, y)$ indicates the spiking rate of an imaginary neuron in response to two bars of luminance x and y, respectively. Color scale is shown at right. Units are arbitrary. White circle and square indicates points around which the linear approximations were computed. **B**. Linear approximation of $f(x, y)$ computed at the point $(x, y) = (6, 0)$, shown as white circle. **C**. Linear approximation computed at the point $(x, y) = (0, -6)$, shown as a white square. Linear approximations closely match behavior of $f(x, y)$ around the points where they were computed. Negative values of linear approximations were truncated to zero.

highly nonlinear function of x and y, such as the function illustrated by a pseudocolor plot in figure 1A. This function indicates that the neuron will have the strongest response when both bars are bright (i.e., if x and y are both large). On the other hand, if the second bar is dark (i.e., large negative y), then the neuron will also response relatively strongly as long as the first bar is not bright (i.e., x is zero or negative).

The global structure of $f(x, y)$ cannot be captured by any linear approximation. The plot of any linear function of two variables is a plane, which clearly cannot have two peaks. Nonetheless, linear approximations are very good at capturing local structure of (smooth) functions. One can compute a linear approximation around an operating point (x_0, y_0) that matches $f(x, y)$ in the neighborhood of (x_0, y_0).

In multivariable calculus, one learns that the linear approximation of a multivariable function $F(\mathbf{x})$ of vector $\mathbf{x} \in \mathbf{R}^n$ calculated at a point $\mathbf{a} \in \mathbf{R}^n$ is

$$L_\nabla(\mathbf{x}) = F(\mathbf{a}) + \nabla F(\mathbf{a}) \cdot (\mathbf{x} - \mathbf{a}). \tag{1}$$

The gradient $\nabla F(\mathbf{x})$ is the vector of all the partial derivatives of F at $\mathbf{x}$. L_∇ is a very good approximation of F near $\mathbf{a}$, but it may be a poor approximation for more distant points. In practice, one may replace the gradient vector $\nabla F(\mathbf{a})$ in (1) with another vector $\mathbf{h}(\mathbf{a})$ obtaining a linear approximation

$$L(\mathbf{x}) = F(\mathbf{a}) + \mathbf{h}(\mathbf{a}) \cdot (\mathbf{x} - \mathbf{a}). \tag{2}$$

If $\mathbf{h}(\mathbf{a}) \neq \nabla F(\mathbf{a})$, then L will be worse than L_∇ at approximating values of $F(\mathbf{x})$ near $\mathbf{a}$. However, one may choose an $\mathbf{h}(\mathbf{a})$ so that L better approximates $F(\mathbf{x})$ at points further away from $\mathbf{a}$.

A key point is the linear approximation L could depend strongly on the point $\mathbf{a}$ around which it was calculated. In the two-dimensional example of figure 1, one could calculate linear approximation around different operating points to capture different features of the $f(x, y)$. For example, a linear approximation around $(x, y) = (6, 0)$ could show how $f(x, y)$ increases with increasing y when x is large (figure 1B). On the other hand, a linear approximation around $(x, y) = (0, -6)$ could indicate that decreasing y will increase the function when operating around a point where y is negative (figure 1C).

In this paper, we show how one can easily calculate such linear approximations of neuron's response to a stimulus. By calculating linear approximations around differing points, one can obtain insight into the nature of the neuron's response to a stimulus. In the following section, we outline how standard stimulus-spike correlation analyses can be used to compute such linear approximations.

3 Calculating linear approximations of neuronal response to a stimulus

As a starting point for our analysis, we will approximate the spiking probability of a neuron as a function purely of a stimulus. In particular, since we ignore how a neuron's spiking probability can be modulated by its own spiking history, we approximate the firing times of a neuron as a inhomogeneous Poisson process. (Such an approximation is implicit when employing stimulus-spike correlation analyses.)

Unlike typical approaches that employ stimulus-spike correlation, we do not postulate a fundamentally linear response to the stimulus or sensitivity to just one direction of stimulus space. Instead, we will allow a neuron's spiking probability to be an arbitrary function of the stimulus. Let the vector $\mathbf{X}$ represent the recent stimulus, i.e. the stimulus at as many previous time steps as influence the neuron's spiking probability. We fix some time bin relative to the stimulus, and let the random variable R represent the number of spikes the neuron fires in that time bin.[1]

We model the probability distribution of R as a Poisson distribution whose mean is $F(\mathbf{X})$, some unknown function of the stimulus. In other words, we assume the probability distribution of R conditioned on the stimulus $\mathbf{X}$ is

$$\Pr(R \,|\, \mathbf{X}) = \Gamma(R, F(\mathbf{X})), \tag{3}$$

where $\Gamma(n, \lambda) = \frac{1}{n!}\lambda^n e^{-\lambda}$ is the Poisson distribution with mean λ.

Note that we do not assume a particular functional form of $F(\mathbf{X})$. The response function $F(\mathbf{X})$ could be some highly nonlinear function of the stimulus $\mathbf{X}$. We only require that $F(\mathbf{X})$ is a differentiable function of $\mathbf{X}$.

The function $F(\mathbf{X})$ captures how the recent stimulus determines the spiking probability. If $\mathbf{X}$ is high-dimensional, we have no hope of sampling $F(\mathbf{X})$ over all possible stimuli and estimating it directly. To make progress, one typically restricts $F(\mathbf{X})$ to a small class of models and attempts to determine the parameters of the model.

[1] Clearly the distribution of R depends on the time bin chosen.

Rather than postulating a form for $F(\mathbf{X})$ that is valid for all $\mathbf{X}$, we instead focus on describing the behavior of $F(\mathbf{X})$ when the stimulus is similar to some reference stimulus $\hat{\mathbf{x}}$. Since we restrict ourselves to finding such a local approximation of $F(\mathbf{X})$ that is valid only for $\mathbf{X}$ close to $\hat{\mathbf{x}}$, we can employ a linear approximation of $F(\mathbf{X})$ in order to study its properties.

We use stimulus-spike correlation to estimate the linear approximation of $F(\mathbf{X})$. To probe the response of the neuron for $\mathbf{X}$ near $\hat{\mathbf{x}}$, we create many realizations of a stimulus $\mathbf{X}$ that are equal to the reference stimulus $\hat{\mathbf{x}}$ plus some noise $\mathbf{Z}$. If the magnitude of the noise $\mathbf{Z}$ is sufficiently small, then the average response of the neuron will be well represented by the linear approximation (1) evaluated around $\mathbf{a} = \hat{\mathbf{x}}$,

$$F(\mathbf{X}) = F(\hat{\mathbf{x}} + \mathbf{Z}) \approx F(\hat{\mathbf{x}}) + \nabla F(\hat{\mathbf{x}}) \cdot \mathbf{Z}. \tag{4}$$

Since $F(\mathbf{X})$ is an unknown function, we don't know the scale at which the local linear approximation (4) is valid. Moreover, there are practical limits on how small one can make magnitude of the noise $\mathbf{Z}$ and still reliably detect how the noise is modulating the response of the neuron. Hence, we don't view the local linear approximation (4) as the linear approximation that will best capture the response of the neuron to $\mathbf{X}$. Instead, we allow our approximation to vary from the linear approximation of multivariable calculus and view the response of the neuron as being approximated by the more general linear equation

$$F(\mathbf{X}) = F(\hat{\mathbf{x}} + \mathbf{Z}) \approx \mu_R(\hat{\mathbf{x}}) + \bar{\mathbf{h}}(\hat{\mathbf{x}}) \cdot \mathbf{Z}, \tag{5}$$

where $\bar{\mathbf{h}}(\hat{\mathbf{x}})$ is a linear kernel and $\mu_R(\hat{\mathbf{x}})$ is the average value of R in response to $\mathbf{X}$. Note that both $\bar{\mathbf{h}}(\hat{\mathbf{x}})$ and $\mu_R(\hat{\mathbf{x}})$ will depend on the statistics of the noise $\mathbf{Z}$ (as well as the mean stimulus $\hat{\mathbf{x}}$), though the notation does not make this fact clear.

We can use standard stimulus-spike correlation techniques to estimate the linear kernel [30]. For completeness, we sketch the well-known result that one can estimate the linear kernel by correlating the neuron's response R with the noise $\mathbf{Z}$ (assumed to be mean zero, as one can incorporate the mean into $\hat{\mathbf{x}}$). Taking the expected value of (3) conditioned on the stimulus $\mathbf{X}$, we see that $E(R \,|\, \mathbf{X}) = F(\mathbf{X})$. Combining this with (5), we obtain that

$$E(R \,|\, \mathbf{X}) - \mu_R(\hat{\mathbf{x}}) \approx \bar{\mathbf{h}}(\hat{\mathbf{x}}) \cdot \mathbf{Z}.$$

Multiplying by the noise $\mathbf{Z}$ and taking the expected over all values of the noise, we calculate that

$$E(\mathbf{Z}(R - \mu_R(\hat{\mathbf{x}}))) \approx E(\mathbf{Z}(\bar{\mathbf{h}}(\hat{\mathbf{x}}) \cdot \mathbf{Z})) = C_{\mathbf{Z}} \bar{\mathbf{h}}(\hat{\mathbf{x}}), \tag{6}$$

where $C_{\mathbf{Z}}$ is the (known) covariance matrix of the noise $\mathbf{Z}$.

Therefore, to estimate the linear kernel, we average $\mathbf{Z}(R - \mu_R(\hat{\mathbf{x}}))$ over the experiment, then divide by the covariance matrix $C_{\mathbf{Z}}$. We obtain the estimate of the linear kernel,

$$\mathbf{h}(\hat{\mathbf{x}}) = C_{\mathbf{Z}}^{-1} \langle \mathbf{Z}(R - \mu_R(\hat{\mathbf{x}})) \rangle = C_{\mathbf{Z}}^{-1} \langle (\mathbf{X} - \hat{\mathbf{x}})(R - \mu_R(\hat{\mathbf{x}})) \rangle, \tag{7}$$

where $\langle \cdot \rangle$ indicates the average over the experiment and $\mu_R(\hat{\mathbf{x}}) = \langle R \rangle$. We have dropped the bar off of $\mathbf{h}$ to indicate that it is the estimate of $\bar{\mathbf{h}}$ obtained from an experiment.

Note that $\mathbf{h}(\hat{\mathbf{x}})$ is almost identical to the weighted spike-triggered average estimate of a linear kernel, which is[2] $C_{\mathbf{Z}}^{-1}\langle \mathbf{Z}|R=1\rangle = C_{\mathbf{Z}}^{-1}\langle \mathbf{Z}R\rangle/\langle R\rangle$. Ignoring the proportionality constant $1/\langle R\rangle$, the only difference is that we have subtracted off the mean response $\mu_R(\hat{\mathbf{x}})$. Moreover, since $E(\mathbf{Z}(R-c)) = E(\mathbf{Z}R)$ for any real number c (as the noise $\mathbf{Z}$ is mean zero), the expected value of $\mathbf{h}(\hat{\mathbf{x}})$ is identical to that of the weighted spike-triggered average. So the natural question is why we have left $\mu_R(\hat{\mathbf{x}})$ in our formula (7).

The reason we use (7) as written is because we are interested in exploring the behavior of neurons around stimuli $\hat{\mathbf{x}}$ that may already elicit is significant response from the neurons. The actual mean response $\mu_R(\hat{\mathbf{x}})$ may be a large number. Although setting $\mu_R(\hat{\mathbf{x}})$ to zero in (7) does not alter the expected value of our estimator $\mathbf{h}(\hat{\mathbf{x}})$, it will change the variance. In fact, we show that using $\mu_R(\hat{\mathbf{x}})$ in (7) minimizes the variance of our estimate $\mathbf{h}(\hat{\mathbf{x}})$, provided that we can neglect any skew in the noise $\mathbf{Z}$ and neglect any deviation of the neuronal response from a linear approximation of the form (5).

To demonstrate that using $\mu_R(\hat{\mathbf{x}})$ minimizes the variance of $\mathbf{h}(\hat{\mathbf{x}})$, we first minimize the variance of the estimator $\langle \mathbf{Z}(R-c)\rangle$ over real numbers c. Let n denote the number of samples in the experiment; let $\mathbf{Z}_i$ and R_i denote the values of noise and neuron response, respectively, in sample i. Then, we can write

$$\langle \mathbf{Z}(R-c)\rangle = \frac{1}{n}\sum_{i=1}^{n} \mathbf{Z}_i(R_i - c).$$

Clearly the expected value of this estimator is $E(\langle \mathbf{Z}(R-c)\rangle) = E(\mathbf{Z}R)$, independent of c. To calculate the covariance matrix of this estimator, we compute

$$\begin{aligned} E(\langle \mathbf{Z}(R-c)\rangle\langle \mathbf{Z}^T(R-c)\rangle) &= E\left(\frac{1}{n^2}\sum_{i,j=1}^{n} \mathbf{Z}_i\mathbf{Z}_j^T(R_i-c)(R_j-c)\right) \\ &= \frac{1}{n}E\left(\mathbf{Z}\mathbf{Z}^T(R-c)^2\right) + \frac{n-1}{n}E(\mathbf{Z}R)E(\mathbf{Z}^TR), \end{aligned}$$

where for the last step, we separated out the n terms where $i=j$ from the $n(n-1)$ terms where $i \neq j$. Since we assume samples are independent, we can factor out the expected values of the latter terms. Subtracting off $E(\mathbf{Z}R)E(\mathbf{Z}^TR)$, we obtain that the the covariance matrix of our estimator is

$$\frac{1}{n}\left[E\left(\mathbf{Z}\mathbf{Z}^T(R-c)^2\right) - E(\mathbf{Z}R)E(\mathbf{Z}^TR)\right].$$

We look for critical points by differentiating with respect to c and setting the result to zero. We find that c must satisfy $E(\mathbf{Z}\mathbf{Z}^T 2(R-c)) = 0$, or

$$cC_{\mathbf{Z}} = E(\mathbf{Z}\mathbf{Z}^TR). \tag{8}$$

This is an overdetermined system for c, and one would not expect the same value of c to be a critical point for each component of the covariance matrix. However, if we assume that the mean of R given the noise $\mathbf{Z}$ is given by (5) and we neglect any skew of $\mathbf{Z}$, then

$$E(\mathbf{Z}\mathbf{Z}^TR) \approx E(\mathbf{Z}\mathbf{Z}^T(\mu_R(\hat{\mathbf{x}}) + \bar{\mathbf{h}}(\hat{\mathbf{x}})\cdot\mathbf{Z})) \approx C_{\mathbf{Z}}\mu_R(\hat{\mathbf{x}}).$$

[2]We assume one takes the average of just the mean zero noise $\mathbf{Z}$ triggered on the event of a spike and that at most one spike occurs in the bin.

Therefore, setting $c = \mu_R(\hat{\mathbf{x}})$ satisfies (8), showing this value of c is a critical point for all components of the covariance matrix of $\langle \mathbf{Z}(R-c)\rangle$.

Finally, we observe that each component of the estimator $C_{\mathbf{Z}}^{-1}\langle \mathbf{Z}(R-c)\rangle$ is simply a linear combination of the components of $\langle \mathbf{Z}(R-c)\rangle$. This implies each component in the covariance matrix of the former is a linear combination of components of the covariance matrix of the latter. Consequently, $c = \mu_R(\hat{\mathbf{x}})$ is also a critical point for all components of the covariance matrix of $C_{\mathbf{Z}}^{-1}\langle \mathbf{Z}(R-c)\rangle$. Since the variance terms (the diagonal components) are positive and quadratic in c, we conclude that the choice $c = \mu_R(\hat{\mathbf{x}})$ in our definition of $\mathbf{h}(\hat{\mathbf{x}})$ minimizes the variance of the estimator.

We emphasize that in this context, we cannot talk about *the* linear kernel that captures the response of a neuron. Since the calculated linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ depends on choice of reference stimulus $\hat{\mathbf{x}}$, we obtain a whole family of kernels that, combined with the linear approximation (5), capture the neural response around each operating point $\hat{\mathbf{x}}$. The linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ represents which stimulus features modulate the neuron's response when the stimulus is close to $\hat{\mathbf{x}}$. Typically, we are concerned only with the direction of $\mathbf{h}(\hat{\mathbf{x}})$ (not its length) because the direction is enough to specify these stimulus features. So, unless specified otherwise, we will normalize $\mathbf{h}(\hat{\mathbf{x}})$ to a unit vector.

4 Demonstration with two-dimensional example

We demonstrate the calculation of linear approximations with two examples. The first example is a two-dimensional model that is simplistic but more easily visualized. The second example is a 50-dimensional model of a highly selective neuron that is more realistic but less easily visualized.

The two-dimensional example is the visual neuron from section 2 with spiking probability $f(x, y)$ that is a function only of the luminances x and y of two bars. The neuron responds most strongly if both x and y are large, and the neuron also responds moderately strongly if y takes on a large negative value while x is zero or negative. Explicitly,

$$f(x,y) = c_1 g(a_1(x-b_1)) + c_2 g(a_2(y-b_2))g(a_3(x-b_3)) + c_4 g(a_4(y-b_4))g(a_5(x-b_5)) \quad (9)$$

where $g(x) = 1/(1+e^{-x})$ is a sigmoidal function. We set the parameters[3] so that the first term created an overall bias for larger values of x, the second term created a large peak of $f(x, y)$ for large x and y, and the third term created a smaller peak for negative values of x and y. A plot of $f(x, y)$ was shown in figure 1A. Our goal with this example is to compare $f(x, y)$ to what we can recover about $f(x, y)$ by linear approximation.

4.1 Different linearizations reveal different features

In this section, we will illustrate how the linear approximation depend on reference point and noise statistics. In these computations, we will use a huge number of realizations ($n =$ 100,000) to ensure the results did not depend on the amount of data available. We will discuss the effects of lowering n in section 4.2.

[3]The parameters were $a_1 = a_2 = a_3 = 0.5$, $a_4 = a_5 = -0.5$, $b_1 = b_5 = 3$, $b_2 = b_3 = 4$, $b_4 = -6$, $c_1 = 0.5$, $c_2 = 5$, and $c_4 = 3$.

First, we explore how the response properties of this neuron would be detected by a linear approximation centered at the origin, i.e. the way in which one usually employs the spike-triggered average. Since $\hat{\mathbf{x}} = (0, 0)$, we stimulate the neuron with Gaussian white noise $(X, Y) = \mathbf{Z}$, where each component has standard deviation σ. For each of n realizations of the noise, we calculate the response R according to (3) with $F = f(X, Y)$. We then calculate the linear kernel $\mathbf{h}(0, 0)$ according to (7).[4] With $\sigma = 1$, the linear kernel is $\mathbf{h}(0, 0) \approx (.9, -.4)$. Since the magnitude of the noise was small compared with the structure of $f(x, y)$, $\mathbf{h}(0, 0)$ points approximately in the same direction as the gradient $\nabla f(0, 0)$. With $\sigma = 10$, the kernel points is $\mathbf{h}(0, 0) \approx (0.998, 0.06)$. The larger noise smears the function f so both peaks in f influence the linear approximation. The upward pull of the tall narrow peak in the upper right effectively balances the downward pull of the low broad peak in the lower left, and the kernel points almost entirely in the positive x direction. Both linear kernels calculated at around the origin are shown by the arrows coming from the diamond (⧫) in figure 2A. Both calculations indicate the neuron may prefer positive x. Neither calculation reveals that the neuron would respond most strongly to the combination of large x and y values. It appears that the neuron prefers negative y values ($\sigma = 1$ case) or that the value of y does not influence the neuronal response ($\sigma = 10$ case).

Next, we compute a linear approximation around a different stimulus $\hat{\mathbf{x}}$ to examine how the stimuli around that point influence the neuronal response. To do so, we add noise on top of the base stimulus $\hat{\mathbf{x}}$, stimulating the neuron with $(X, Y) = \hat{\mathbf{x}} + \mathbf{Z}$. As before, for each of n realizations of the noise, we calculate the response R according to (3) with $F = f(X, Y)$, and compute the linear kernel using (7).

For example, imagine that we knew (either from the above linear kernels at $(0, 0)$ or from other experiments) that a positive luminance x of the first bar causes the neuron to respond strongly. We might take $\hat{\mathbf{x}} = (6, 0)$. With weak noise of $\sigma = 1$, the kernel estimate is $\mathbf{h}(6, 0) \approx (.5, .9)$, and with strong noise of $\sigma = 10$, the kernel estimate is $\mathbf{h}(6, 0) \approx (.6, .8)$. Both linear kernels are shown by arrows coming from the circle (•) in figure 2A. In this case, the direction the linear kernel changes little with the magnitude of the noise used. Both estimates of $\mathbf{h}(6, 0)$ point toward the peak of large x and y values and indicate that, if we start with a large value of the first bar's luminance x, then we can increase the neuron's response even further by also increasing the luminance y of the second bar. Hence the linear approximation around $\hat{\mathbf{x}} = (6, 0)$ revealed features of the neuron's response properties that were invisible with the typical spike-triggered average that is centered at the origin.

If we start with a negative value of y, we obtain different information. For $\hat{\mathbf{x}} = (0, -6)$, the linear kernel computed with either weak ($\sigma = 1$) or strong ($\sigma = 10$) noise was $\mathbf{h}(0, -6) \approx (-.2, -.97)$. Both kernels are shown as arrows coming from a square (■) in figure 2A. These linear approximations point in the direction of the broad peak for negative y and moderately negative x.

The choice of noise can dramatically affect the direction of the kernel. As shown by the arrows coming from the star (★) in figure 2A, the estimates of the kernel $\mathbf{h}(4, -5)$ based on weak ($\sigma = 1$) and strong ($\sigma = 10$) noise point in nearly opposite directions. The point $\hat{\mathbf{x}} = (4, -5)$ is at the base of the lower peak in $f(x, y)$, and the gradient $\nabla f(4, -5)$ points up the peak, i.e., downward and to the left. The linear kernel computed with weak noise is

[4]Since the noise was white, the covariance matrix was $C_{\mathbf{Z}} = \sigma^2 I$, where I is the 2×2 identity matrix.

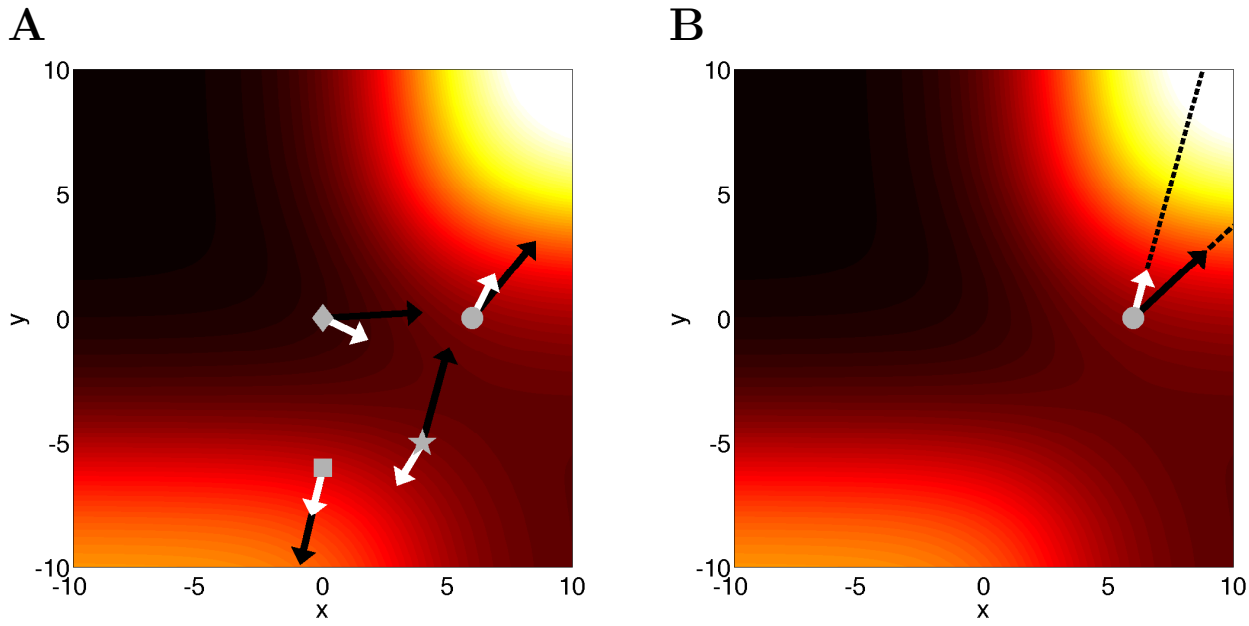

Figure 2: Estimates of linear kernel vary with location and noise magnitude. **A**. Linear kernel estimates for weak noise $\sigma = 1$ (short white arrow) and strong noise $\sigma = 10$ (long black arrow) for linear approximations at $(0, 0)$ (♦), at $(6, 0)$ (•), at $(-6, 0)$ (■), and at $(4, -5)$ (★) as described in section 4.1. Both linear kernels calculated at origin indicate that firing rate increases with x but miss the positive y component of large peak. Both estimates of $\mathbf{h}(6, 0)$ point toward the global maximum. Estimates of $\mathbf{h}(-6, 0)$ indicate the location of the smaller peak. The estimate of $\mathbf{h}(4, -5)$ depends on size of noise. The linear approximation based on $\mathbf{h}(6, 0)$ and $\sigma = 1$ was shown in Figure 1B. The linear approximation based on $\mathbf{h}(0, -6)$ and $\sigma = 1$ was shown in Figure 1C. With the exception of the plots in Figure 1B & C, we view linear kernels as unit vectors. Different arrow lengths were used here just for display purposes. **B**. Examples of linear kernels computed at $(6, 0)$ from 500 noise realizations. Arrows as in panel A. Dashed lines illustrate the lines $X = (6, 0) + s\mathbf{h}(6, 0)$ along which we sampled f to explore behavior in the direction of kernels, as described in section 4.3.

based on local sampling of $f(x, y)$, and we obtain a kernel estimate $\mathbf{h}(4, -5) \approx (-.5, -.9)$ that closely resembles the gradient. On the other hand, the strong noise effectively smooths $f(x, y)$ by sampling the function over a large region. As a result, the upper right peak in $f(x, y)$ exerts a large influence, and the resulting kernel estimate $\mathbf{h}(4, -5) \approx (.3, .96)$ points toward that peak. The linear approximation formed using (5) with either of these kernels can be considered a legitimate approximation of the function $f(x, y)$ since the magnitude of the noise determines the length scale of the linear approximation.

4.2 Reduction in variance of estimator

The above estimates of $\mathbf{h}(\hat{\mathbf{x}})$ are based on a huge number of realizations ($n = 100,000$) so that variance in the estimate would not play a big role in the computation. In this section, we consider a smaller number of realizations ($n = 500$) so that the kernel estimates $\mathbf{h}(\hat{\mathbf{x}})$ would have a larger variance. Our goal is to determine how much our choice of μ_R in formula (7) reduced the variance compared to estimates of the kernels with μ_R replaced with zero. To understand the practical significance of the variance calculations, note that if a method reduces the variance by 50%, that means one could reduce the number of realizations by 50% to achieve an estimate with the original variance.

For each of the linear kernels mentioned in section 4.1, we recomputed the kernel with 500 realizations of the noise and calculated the variance in the estimate by repeating this stimulation 400 times. For comparison, we repeated each calculation after replacing the μ_R in (7) by zero. For all the kernel estimates with $\hat{\mathbf{x}} \neq (0, 0)$, we found that using μ_R reduced the variance of the estimates by 35%–50% (reduced the standard deviation by 20%–30%) compared to the kernel estimate with μ_R replaced with zero. Even when computing the typical linearization centered around the origin $\hat{\mathbf{x}} = (0, 0)$, using (7) reduced the variance in the estimate of $\mathbf{h}(0, 0)$ by at least 20% (reduced the standard deviation by at least 10%) compared to the kernel estimate with μ_R replaced with zero.

We conclude that, at least for this example, we achieved an improved estimate of $\mathbf{h}(\hat{\mathbf{x}})$ by subtracting off μ_R in (7). The amount of improvement depends on the magnitude of the average firing rate μ_R. For example, if we doubled the $f(x, y)$ in (9) so that the average firing rate μ_R doubled, the reduction in variance was larger than above values. On the other hand, if we halved $f(x, y)$, the reduction in variance was smaller than the above values. Nonetheless, in every example we tested, using μ_R in (7) reduced the variance in our estimate of $\mathbf{h}(\hat{\mathbf{x}})$, consistent with our analysis.

4.3 Displaying preferred stimulus features

The linear kernel at a point $\hat{\mathbf{x}}$ points in the direction of stimulus space along which the response of the neuron increases most rapidly. As a result, the function $F(\mathbf{X})$ should increase as $\mathbf{X}$ moves along the line $\mathbf{X} = \hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$ for $s > 0$ as long as $\mathbf{X}$ is close enough to $\hat{\mathbf{x}}$ that $F(\mathbf{X}) - \mu_R(\hat{\mathbf{x}})$ is proportional to $\mathbf{h}(\hat{\mathbf{x}}) \cdot (\mathbf{X} - \hat{\mathbf{x}})$. Hence, one way to summarize our linear approximation is to plot $\mathbf{X} = \hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ for a value of α where $F(\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}}))$ is large.

We determine an appropriate value of the parameter α as follows. After calculating $\mathbf{h}(\hat{\mathbf{x}})$ using noise with magnitude σ, we repeatedly present the stimuli $\mathbf{x}_s \equiv \hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$ for $s = \sigma, 2\sigma, \ldots, 20\sigma$. We seek α in steps of size σ because σ represents the length scale over

which $F(\mathbf{X})$ was averaged to determine $\mathbf{h}(\hat{\mathbf{x}})$. After presenting each stimulus 100 times, we estimate $F(\mathbf{x}_s)$ as the average number of spikes elicited by that stimulus and estimate the standard error δ_s as the standard deviation of the mean. We let $s_{\max}$ be the value of s that maximizes $F(\mathbf{x}_s)$. Then we let α be the smallest value of s where $F(\mathbf{x}_s) \geq F(\mathbf{x}_{s_{\max}}) - 2\delta_{s_{\max}}$. In this way, $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ represents a stimulus pointed out by $\mathbf{h}(\hat{\mathbf{x}})$ that drives the neuron strongly.

We demonstrate this procedure with our example two-dimensional function $f(x, y)$ and the linearization around $\hat{\mathbf{x}} = (6, 0)$. We estimated the kernel $\mathbf{h}(6, 0)$ with $N = 500$ realizations of $\sigma = 1$ and $\sigma = 10$ noise. The kernels are shown in figure 2B. The lines $\mathbf{X} = \hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$ are also displayed on the graph. The result of sampling the response of the neuron along each line is shown in figure 3A,C. In both cases, the response of the neuron increases in the direction of the linear kernel, indicating that the kernel did capture stimulus features to which the neuron is sensitive. Our choice of α corresponds to the first data point (shown by a triangle) that is above the line $F = F(\mathbf{x}_{s_{\max}}) - 2\delta_{s_{\max}}$, shown by the gray line.

We summarize these results in figure 3B,D. Each panel displays the vector (x, y) represented by two rectangles whose shading corresponds to the value of a component of the vector. The representation captures the meaning of x and y as the luminance of two bars. More importantly, such a representation can generalize to higher dimensions, unlike the representations in figure 2. In figure 3B,D, the first two panels represent the original stimulus $\hat{\mathbf{x}} = (6, 0)$ and the linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ estimated from the analysis, respectively. The third panel represents the combination of stimulus and linear kernel $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ that drives the neuron strongly.

For comparison, last panels of figure 3B,D display linear kernels computed at the origin. As when the kernel was computed at $(6, 0)$, these were also computed with 500 realizations of the noise for $\sigma = 1, 10$. For both values of σ, the second component of the kernel is nearly zero, corresponding to arrows pointing rightward in figure 2. In contrast, in the combinations $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$, both components are positive and indicate that the neuron responds strongly when both bars have a high luminance. These combination plots correspond to points in figure 2B that lie along the lines emanating from the linear kernel directions.

5 Demonstration with a highly selective neuron

For our second example, we demonstrate the results of calculating a linear approximation of a model neuron that is highly selective to particular stimulus features. The model is a caricature of the selectivity observed in many nuclei of songbirds, motivated by the observation that many of these neurons respond selectively to auditory playback of a bird's own song (BOS) [14, 5, 16, 13, 15, 9, 18]. BOS neurons respond preferentially to playback of the BOS compared to songs that are similar, such as the BOS played in reverse or the songs of other birds of the same species. These neurons can be selective to syllables in the song or even the temporal order of the syllables within the song.

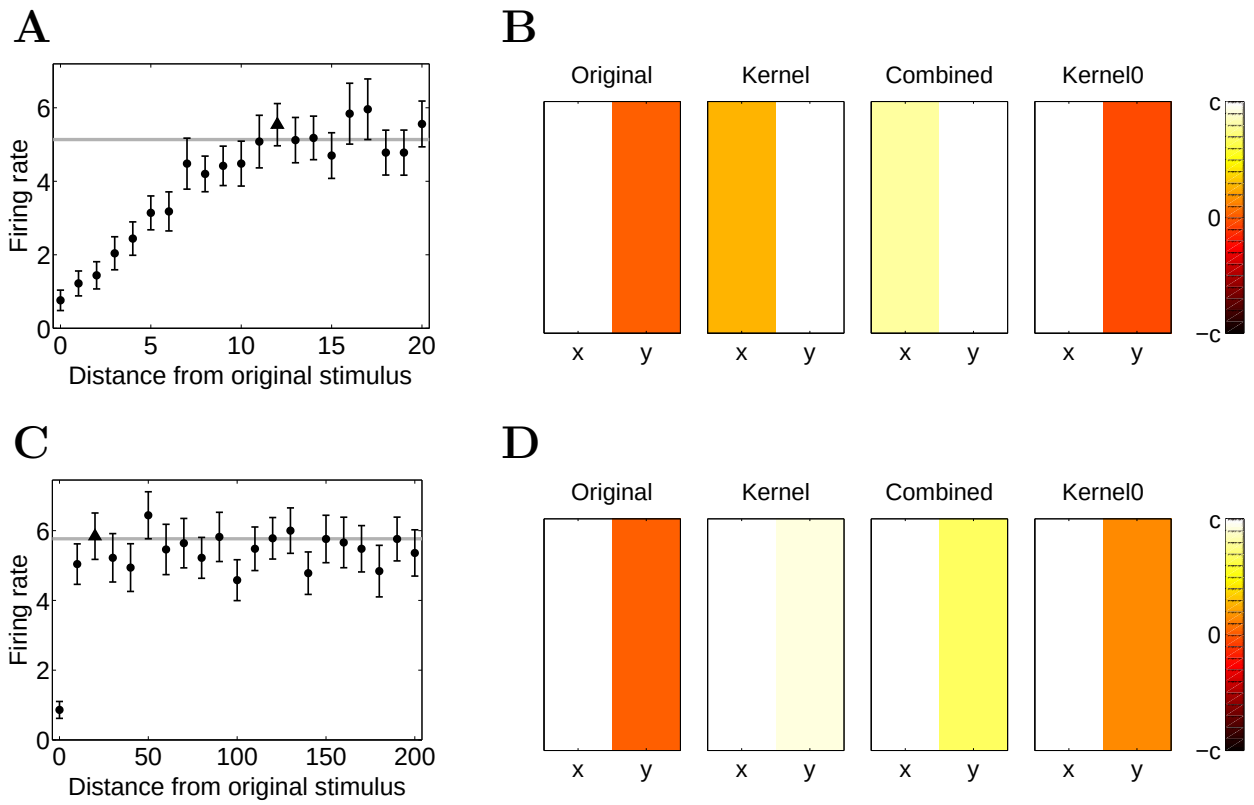

Figure 3: Results from calculating the linear approximation around $\hat{\mathbf{x}} = (6, 0)$ using 500 realizations of noise with $\sigma = 1$ (panels A and B) and $\sigma = 10$ (panels C and D). **A**. The firing rate of the neuron increases as we stimulate with $\hat{\mathbf{x}}$ plus a multiple of the linear kernel $\mathbf{h}(\hat{\mathbf{x}})$. Each point is calculated from 100 repetitions of $\hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$. Error bars indicate two standard errors. Gray line represents firing rate corresponding to the maximum observed firing rate minus two standard errors. The first point above the gray line is represented by a triangle and corresponds to the α used in $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ in panel B. **B**. Pseudocolor plots of the stimulus and linear kernels. The first plot is $\hat{\mathbf{x}} = (6, 0)$. The second plot is $\mathbf{h}(\hat{\mathbf{x}}) \approx (0.3, 0.96)$, which is also displayed by the short white arrow in figure 2B. The third plot is the combination $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}}) \approx (9, 12)$ with $\alpha = 12$. For comparison, the last plot shows the linear kernel calculated at the origin, $\mathbf{h}(0, 0) = (0.999, -0.04)$. In each plot, the scale (shown at right) was adjusted so that c was the maximum of the absolute value of the components. **C**. The firing rate also increased along the direction of the kernel computed with $\sigma = 10$. Plot is identical to panel A. Because the noise was larger in magnitude, we tested the firing rate at larger steps, increasing the scale of the distance from the original stimulus $\hat{\mathbf{x}} = (6, 0)$. **D**. The same pseudocolor plots of the stimulus and linear kernels as in panel B. In the case where $\sigma = 10$, the kernel was $\mathbf{h}(\hat{\mathbf{x}}) \approx (0.7, 0.7)$ and the combination plot was $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}}) \approx (21, 14)$ with $\alpha = 20$. The linear kernel calculated at the origin was $\mathbf{h}(0, 0) \approx (0.99, 0.14)$.

5.1 Description of the model neurons

For our model neuron, the stimulus is a simplified syllable of a "song" containing ten frequencies at each of five time bins. In this way, $\mathbf{X}$ contained 50 dimensions, with X_j^i representing the power at frequency j and time bin i, for $i = 1, 2, \ldots, 5$ and $j = 1, 2, \ldots, 10$. For simplicity, we ignored any phase information of the song, and we only included a small number of times and frequencies.

To implement such selectivity in a simplified model of the form (3), we employed a divisive normalization model [8, 28]. We modeled the expected number of spikes (in some time bin) elicited by the stimulus $\mathbf{X}$ by the function

$$F(\mathbf{X}) = 0.1 + \frac{g(\mathbf{h}_1 \cdot \mathbf{X} - 10) + 2g(\mathbf{h}_2 \cdot \mathbf{X} - 2)g(2(\mathbf{h}_1 \cdot \mathbf{X} - 10))}{0.1 + 100 \sum_{j=3}^{27} g(\mathbf{h}_j \cdot \mathbf{X} - 10)} \tag{10}$$

where $g(x) = 1/(1 + e^{-x})$ is a sigmoidal function. The kernels $\mathbf{h}_j$ have the same size and shape as the stimulus $\mathbf{X}$. Although we picture the stimulus $\mathbf{X}$ and the kernels $\mathbf{h}_j$ as matrices in time and frequency, the dot products $\mathbf{h}_j \cdot \mathbf{X}$ are defined by regarding them as vectors with 50 components.

Since the model (10) is a function of the 50-dimensional stimulus vector, we cannot plot $F(\mathbf{X})$ as we did in the previous example. Instead, we describe the model by showing a representative sample of its 27 different linear kernels in figure 4. Each kernel is zero except for a small set of elements that are set to the same positive value. The value of the nonzero elements is determined by the condition that all kernels be normalized to unit vectors.

Projection of the stimulus onto the two kernels $\mathbf{h}_1$ and $\mathbf{h}_2$ can increase the neuron's response. If the projection of $\mathbf{X}$ onto the primary kernel $\mathbf{h}_1$ is large, the stimulus can elicit a strong response of the neuron. The primary kernel is shown in the upper left corner of figure 4. All components of $\mathbf{h}_1$ are zero except for a sequence of frequencies that increases with time. The dot product $\mathbf{h}_1 \cdot \mathbf{X}$ will be large if $\mathbf{X}$ contains power in that sequence of frequencies. In this way, the first term in the numerator of (10) is sensitive to an upward sweep of frequencies in the song $\mathbf{X}$, reminiscent of frequency modulations observed in birds' songs.

A large projection onto the secondary kernel $\mathbf{h}_2$, in contrast, cannot drive the neuron by itself. Even if the projection of $\mathbf{X}$ onto $\mathbf{h}_2$ is large, the stimulus will not elicit a response from the neuron unless the stimulus simultaneously has a large projection onto the primary kernel $\mathbf{h}_1$. This nonlinear interaction between the two kernels makes the secondary kernel effectively a "hidden" kernel that is unmasked only under sufficient activation of the primary kernel. As pictured in the upper right corner of figure 4, the shape of $\mathbf{h}_2$ is an upward sweep of frequencies like $\mathbf{h}_1$, but it includes lower frequencies. The optimal stimulus to the neuron is the simultaneous presence of upward sweeps in both sets of frequencies represented by $\mathbf{h}_1$ and $\mathbf{h}_2$.

The remaining 25 kernels are divisive kernels. If the stimulus $\mathbf{X}$ projects onto those kernels, it will suppress the response of the neuron. We used four groups of divisive kernels, samples of which are shown in the bottom row of figure 4. The first group of divisive kernels contains the four kernels that are sensitive upward sweeps in frequency in the frequencies ranges between $\mathbf{h}_1$ and $\mathbf{h}_2$. The second group contains the six kernels that are sensitive to downward sweeps in frequency. The third group contains the five kernels that are sensitive

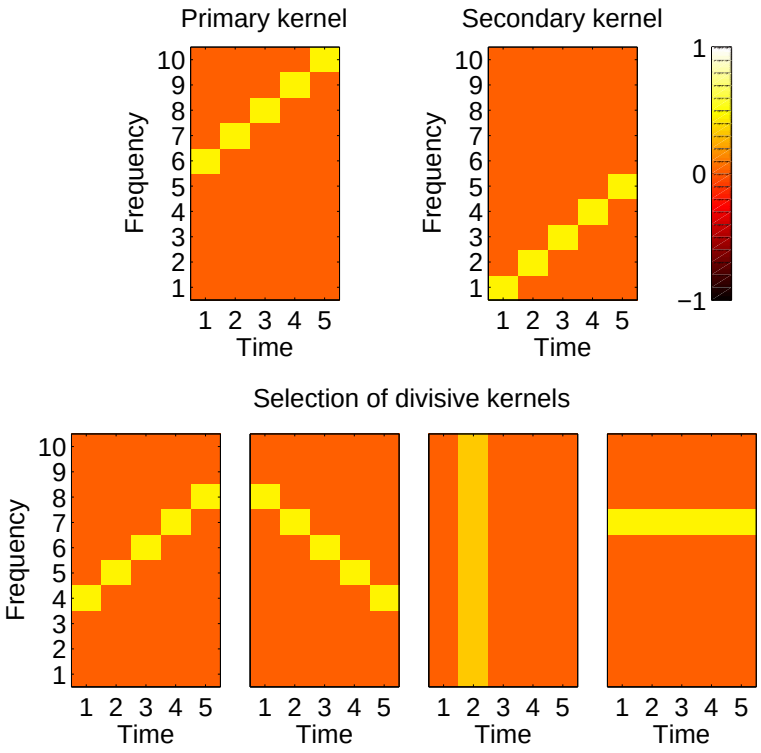

Figure 4: Pseudocolor plots of the kernels of the divisive normalization model (10). The primary kernel $\mathbf{h}_1$ represents an upward sweep of frequencies in the upper frequency band. The secondary kernel $\mathbf{h}_2$ also represents an upward sweep of frequencies, but one that occurs in the lower frequencies. The bottom row contains representative kernels from the four groups of divisive kernels. The first group corresponds to upward sweeps (at frequencies between $\mathbf{h}_1$ and $\mathbf{h}_2$). The second group corresponds to downward sweeps. The third and fourth groups correspond to simultaneous presentation of all frequencies at one time and to presentation of a single constant frequency at all times, respectively.

to the presence of all frequencies in one time step. The fourth group contains the ten kernels that are sensitive to a single constant frequency being present in all time steps.

The latter three groups of divisive kernels overlap with the primary and secondary kernels. The presence in $\mathbf{X}$ of power at a single frequency and time will always contribute to the projection of $\mathbf{X}$ along at least two divisive kernels. The divisive kernels activate at the same rate as the primary kernel and faster than the "unmasking" of secondary kernel by the primary kernel. Hence, the presence of power at a single frequency and time cannot significantly increase the firing rate of the neuron.

The model allows only one way to increase the firing rate significantly above the background firing: the stimulus must include combinations of frequencies that match the primary kernel $\mathbf{h}_1$ and avoid combinations of frequencies that match any one divisive kernel. Once that condition is satisfied, the firing rate can be increased even further by including combinations of frequencies that match the secondary kernel $\mathbf{h}_2$. In this way, model (10) represents a neuron that is highly selective to these upward sweeps in frequency.

5.2 A sparse noise stimulus

Because the model neuron is highly selective, many combinations of frequencies in the stimulus will suppress the neuron's response. If we employed a dense noise stimulus, such as the Gaussian noise of the previous section, the noise will tend to simultaneously activate many of the kernels. It would be highly unlikely for a realization of such dense noise to selectively activate just a small number of kernels, as would be needed to increase the response of the model neuron. If such dense noise were strong enough to significantly modulate the neuron's response, it would be highly likely to suppress the response and prevent our analysis from determining any of the neuron's response properties.

To increase our chances of obtaining useful information from probing the neuron with noise, we need a form of noise whose statistics better matched the types of stimuli that would drive a highly selective neuron. We used a sparse noise that had power at no more than two frequencies in each time bin. In this way, a single realization of the noise would be relatively likely to strongly activate a small number of kernels.

We created each realization of the sparse noise $\mathbf{Z}$ by randomly selecting two frequencies at each time bin, and then randomly selecting the noise at those frequencies to be either positive or negative. Since the stimulus is intended to represent the power of the song at each frequency, each component of the stimulus must be positive. As described in the appendix, if the original stimulus $\hat{\mathbf{x}}$ is small, then we adjust the noise to prevent the stimulus $\hat{\mathbf{x}} + \mathbf{Z}$ from becoming negative. This adjustment results in a positive value for the mean noise. Because equation (7) assumes mean zero noise, we subtract the mean value from the noise and add the mean value to the original stimulus (i.e., we replace $\mathbf{Z}$ with $\mathbf{Z} - E(\mathbf{Z})$ and replace $\hat{\mathbf{x}}$ with $\hat{\mathbf{x}} + E(\mathbf{Z})$). In following examples, we attempt to linearize around stimuli $\hat{\mathbf{x}}$ that contain many zero components. Because the noise for those components must have a positive mean, we effectively linearize around stimuli with all positive components.

Because we allow nonzero noise in only two frequencies per time bin, the components of the noise corresponding to the same time bin are not independent. When the negative and positive values of the noise were symmetric around zero, these components are still uncorrelated. However, the process of adjusting the noise to keep $\hat{\mathbf{x}} + \mathbf{Z}$ nonnegative breaks

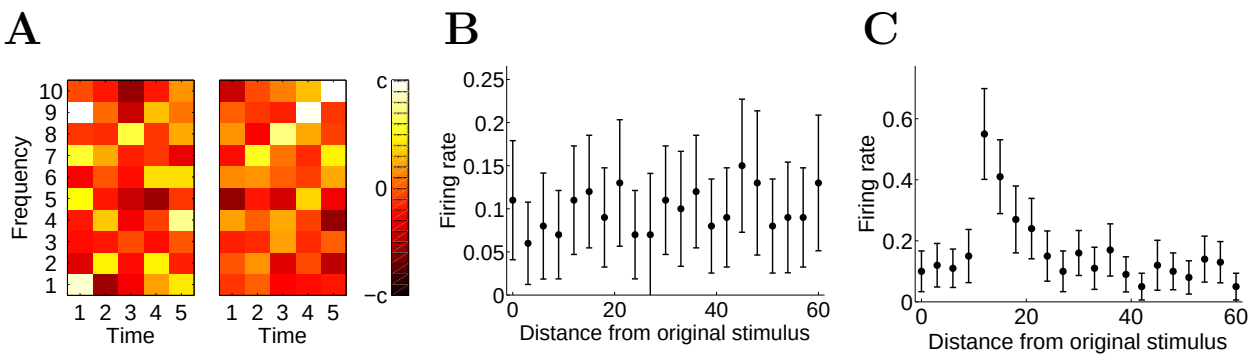

Figure 5: Results from the standard spike-triggered average of highly selective neuron (10). **A**. Linear kernels computed from 10,000 (left) and 200,000 (right) realizations of $\sigma = 3$ sparse noise alone. With 10,000 realizations, the kernel shows little structure. With 200,000 realizations, the structure of the primary kernel (figure 4, upper left) is visible. Scale magnitude c chosen as the maximum absolute value of kernel elements. Even though the kernels were computed with a nonnegative stimulus, negative kernel values are possible because each component of the original stimulus $\hat{\mathbf{x}}$ is effectively the positive mean value of the noise. **B**. Demonstration that the kernel computed from 10,000 noise realization does not capture properties of the neuron's response. The average firing rate was estimated by 100 presentations of the original positive stimulus plus a multiple of the left kernel from panel A. (Negative values were thresholded to zero.) The firing rate does not change with increasing multiples of the estimated kernel. Error bars are two standard errors. **C**. Demonstration that the kernel computed from 200,000 noise realization does capture properties of the neuron's response. Plot is the same as in panel B, except that the right kernel from panel A was used. When the stimulus was 10–15 times the kernel (plus the original positive stimulus), the neuron fired significantly higher than with the original stimulus alone.

that symmetry and can introduce correlations among these noise components. We calculate the covariance matrix $C_{\mathbf{Z}}$ in the appendix.

5.3 Results of analysis

5.3.1 Sparse noise alone

We probed the neuron's response with the sparse noise alone. Because the noise had to be nonnegative, the mean of the noise was positive. Hence, computing the kernel from (7) is effectively computing the linear approximation around a vector $\hat{\mathbf{x}}$ whose components are all positive, equal to the mean of the noise. Nonetheless, computing the kernel in this way is equivalent to the standard way to employ the spike-triggered average.

For a wide range of noise magnitudes ranging from $\sigma = 0.1$ to 100, we generated 10,000 realizations of the noise and computed the linear approximation via (7). The kernel $\mathbf{h}(\hat{\mathbf{x}})$ appeared unstructured (e.g., figure 5A, left). In these cases, the neuron's firing rate was nearly identical to the background firing rate of 0.1, so the estimates of the kernels were created from spikes that were essentially independent of the stimulus. In order to test if the estimated kernel pointed in a direction of stimulus space that modulated the neuron's

response, we sampled the neuron's response to stimuli in the direction of the kernel. These stimuli did not lead to an increased firing rate (e.g., figure 5B). We failed to obtain any indication of the neuron's stimulus preferences from this linear analysis. The suppressive effect of the divisive subfields masked any effect of the other subfields.

To test if vast (and experimentally unrealistic) amounts of data could be used to detect the primary or secondary kernels, we increased the number of noise realization to 200,000. For $\sigma = 3$, we could detect the primary subfield with this linearization centered near the origin (figure 5A, right). We confirmed that stimuli in the direction of the kernel did modulate the response of the neuron (figure 5C). However, even with such a large number of noise realizations, we see no evidence of the secondary kernel. (Even if we further increased the number of noise realizations to 1,000,000, we still could see no evidence of the secondary kernel.)

5.3.2 Linearization around a stimulus to which the neuron responds robustly

Suppose we knew the neuron responded to an upper sweep of frequencies as represented by the primary kernel. This might occur if the upper sweep of frequencies was present in the bird's own song, and that the neuron responded robustly to the presentation of that part of the song. From such experiments, this upper sweep may appear to be the optimal stimulus for the neuron.

To explore the behavior of the neuron to stimuli similar to this upper sweep, we compute the linear approximation around a stimulus $\hat{\mathbf{x}}$ that is nonzero only in the sequence of frequencies in the upper sweep (see figure 6B, left). Each nonzero entry has the same value, x_0. We generated 2,000 realizations of the sparse noise $\mathbf{Z}$ and added it to $\hat{\mathbf{x}}$. As before, the zero entries of $\hat{\mathbf{x}}$ effectively become positive due to adding the mean value of the noise.

The results with noise magnitude $\sigma = 1$ and original stimulus amplitude $x_0 = 6$ are shown in figure 6A,B. The linear kernel (figure 6B, second left) clearly shows the upward frequency sweep in the lower frequency band that is present in the secondary kernel $\mathbf{h}_2$ (figure 4, upper right). By computing our linear approximation around the baseline stimulus $\hat{\mathbf{x}}$, we have unmasked the influence of the secondary kernel $\mathbf{h}_2$ that was invisible when stimulating with the sparse noise alone. The linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ points in the direction of $\mathbf{h}_2$ because, starting at $\hat{\mathbf{x}}$, moving in the $\mathbf{h}_2$ direction increases the neuron's firing rate the most. This example demonstrates how a suitable application of a linear analysis can reveal nonlinear features in the response of a neuron.

As with the two dimensional example in section 4.3, we seek to summarize the results with a combination $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ that maximizes the response of the neuron. Figure 6A shows the results from sampling the response of the neuron in the direction of $\mathbf{h}(\hat{\mathbf{x}})$. We observe that the firing rate of the neuron along the line $\mathbf{x}_s = \hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$ saturates around $s \in (5, 10)$ and then decreases for larger s (the decrease is due to suppression of the firing rate by the divisive kernels). The combination $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ for $\alpha = 6$ is shown in figure 6B, second right. This plot summarizes that the neuron responds most strongly to the combination of frequency upsweeps in both low and high frequency bands.

The linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ contains many values that are negative, but the negative values of $\mathbf{h}(\hat{\mathbf{x}})$ are not an indication the they were based on negative values of power at those frequencies. All components of the effective stimulus become positive when adding the mean

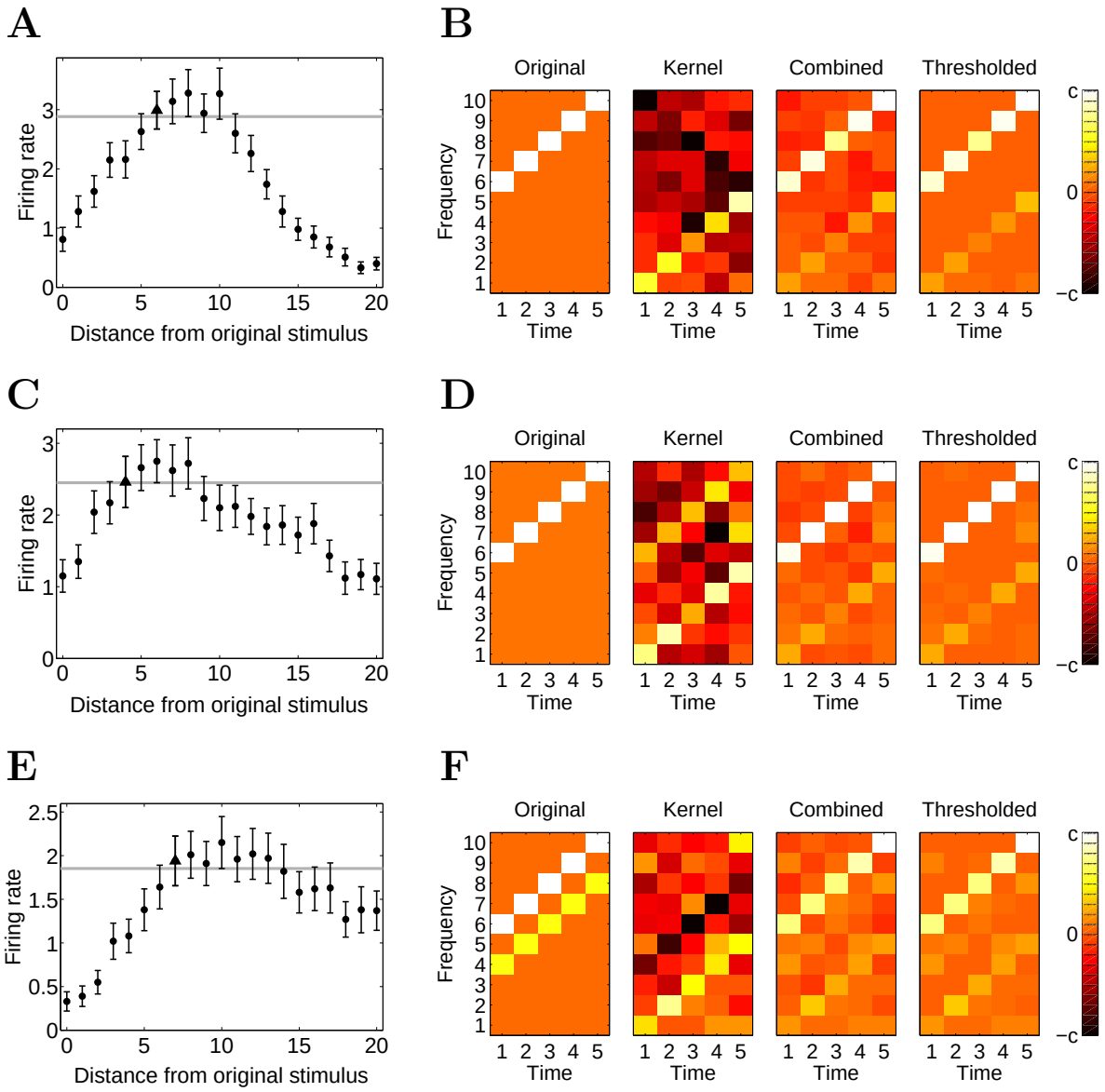

Figure 6: Calculating the linear kernel around different baseline songs $\hat{\mathbf{x}}$. In each case, the firing rate increases as multiples of the linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ are added to the original stimulus $\hat{\mathbf{x}}$, indicating that the kernel captured relevant stimulus features (panels A, C, and E). Pseudocolor plots of original stimulus $\hat{\mathbf{x}}$, linear kernels $\mathbf{h}(\hat{\mathbf{x}})$, and combinations $\hat{\mathbf{x}}+\alpha\mathbf{h}(\hat{\mathbf{x}})$ are shown in panels B, D, and F. Panel as in figure 3, except that right most plot redisplays the combination plot $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ with negative values thresholded to zero. **A,B**. Results centered around an upward sweep song $\hat{\mathbf{x}}$ of magnitude $x_0 = 6$. The original stimulus matches the primary kernel (figure 4, top left). The linear kernel captures the secondary kernel (figure 4, top right). The combination plot is $\hat{\mathbf{x}}+\alpha\mathbf{h}(\hat{\mathbf{x}})$, with $\alpha = 6$ determined from panel *A*. **C,D**. Results centered around an upward sweep song $\hat{\mathbf{x}}$ of magnitude $x_0 = 5$. Both primary and secondary kernels appear in the linear kernel. Even so, the combination plot ($\alpha = 4$) is similar to the $x_0 = 6$ case in panel B. **E,F**. Results centered around a song that combines the upward sweep of panel B with an upward sweep at intermediate frequencies. The linear kernel both captures the secondary kernel and indicates the suppressive nature of the added upward sweep. The combination plot ($\alpha = 7$) eliminates most of the added upward sweep.

value of the noise to $\hat{\mathbf{x}}$. Since most components of the stimulus have only a suppressive effect on the neuron's firing probability, the linear kernel points back to the zero values in those components. However, when calculating values in the direction of the kernel, we stimulate the neuron with stimulus $\hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$. If s is sufficiently large, this stimulus can take on negative values. When running the simulations to calculate figure 6A, we threshold each component of the stimulus $\mathbf{X}$ to nonnegative values. For the same reason, the combination plot (figure 6B, second right) does contain some values that are negative. To reflect the actual stimulus used, we display $\hat{\mathbf{x}} + \alpha\mathbf{h}(\hat{\mathbf{x}})$ for $\alpha = 6$ with all components thresholded to nonnegative values in figure 6B, right. Since many components of the combination were negative, the thresholded picture has a very clean appearance. The thresholding masks the large variation in the estimated values of the linear kernel.

5.3.3 Reduction of variance

As we did with the two-dimensional example, we investigated how much our choice of μ_R in (7) reduced the variance in the estimate of $\mathbf{h}(\hat{\mathbf{x}})$. By using the same algorithm to estimate the variance in the estimators, we discovered that using μ_R decreased the variance in the components of $\mathbf{h}(\hat{\mathbf{x}})$ by 35% on average (decreased the standard deviation by 20% on average) compared to estimating the kernel with μ_R replaced by zero.

5.3.4 Effects of noise magnitude on estimated kernel

We also investigated in how the estimate of the linear kernel varied with noise magnitude σ. Because of the neuron was highly selective to particular stimulus features, large values of the noise magnitude would tend to suppress the neuron's firing; the noise would drive the input too far from the neuron's preferred stimuli. If we doubled the noise magnitude to $\sigma = 2$, we still achieved comparable results. However, increasing the noise magnitude to $\sigma = 5$ greatly suppressed the neuron's firing, and we were unable to detect any structure in the estimated linear kernel. Dropping σ below 0.5 required additional noise realizations to achieve good estimates of the linear kernel, as the weaker noise created less modulation in the neuron's firing rate.

5.3.5 Effects of choice of stimulus $\hat{\mathbf{x}}$ on estimated kernel

We chose the value of the original stimulus amplitude $x_0 = 6$ because such a stimulus elicited a strong response from our model neuron. The determination of the linear kernel $\mathbf{h}(\hat{\mathbf{x}})$, of course, depends on the selection of $\hat{\mathbf{x}}$. In section 5.3.1, we demonstrated that choosing $x_0 = 0$ results in a failure to detect response properties of the neuron with reasonable numbers of noise realizations. Lowering x_0 to 5 altered the response to the original stimulus only slightly. However, the change made a qualitative difference in the form of the estimated linear kernel, as shown in figure 6C,D. In this case, the estimated linear kernel $\mathbf{h}(\hat{\mathbf{x}})$ contained a combination of the primary and secondary kernels of the model neuron. The kernel demonstrated that, starting with $x_0 = 5$, the response of the neuron would increase most rapidly if one simultaneously increased the components of the stimulus corresponding to both upward frequency sweeps. On the other hand, the combination of the linear kernel with the original song (figure 6D, right two panels) was essentially identical to that of the $x_0 = 6$ case. Both

linear approximations indicated that the neuron responded most strongly to a combination of the two frequency sweeps.

Increasing the original song magnitude x_0 did not result in qualitative changes to the estimated linear kernel. This increase did reduce the neuron's firing rate so that more noise realizations were required to achieve good estimates of the linear kernel. For example, we could still recover the secondary kernel starting from $x_0 = 10$ if we presented 6,000 realizations of the noise (not shown).

If the original song $\hat{\mathbf{x}}$ included elements that suppressed the neuron response, the linear kernel could capture this suppression. To demonstrate, we included in the original song a frequency upsweep corresponding to the intermediate frequencies of the first divisive kernel shown in figure 4. We added this upsweep with amplitude 3 to the upsweep of amplitude 6 that corresponded to the primary kernel. Figure 6E,F shows the results from presenting this stimulus $\hat{\mathbf{x}}$ along with 6,000 realizations of the noise. (Tripling the number of realizations compensated for the spike rate dropping due to the suppression.) The linear kernel indicates that the upsweep at intermediate frequencies was indeed suppressive, as the components at those frequencies are negative. At the same time, the linear kernel captures the fact that projections of the stimulus onto the secondary do increase the firing rate (figure 6F, second panel). The combination plots (figure 6F, right two panels) nearly eliminate the upward sweep corresponding to the divisive kernel and are similar to those from the other linear approximations.

6 Discussion

6.1 Application to BOS neurons

The preceding analysis was motivated by our desire to understand how the response of a highly nonlinear neuron is modulated by the stimulus. As one example, we want to understand a neuron selective to a bird's own song (BOS) [14, 5, 9, 18]. The nature of the selectivity of BOS neurons is not well understood, as simple manipulations of the song tend to suppress the neuron's firing. Since the bird already gives us the BOS, we begin with a natural reference point around which to attempt to calculation a linear approximation of the neuron's response to song stimuli.

The analysis above provides a mechanism by which one might study the selectivity of BOS neurons to stimuli that are close to the BOS. Although the neurons are referred to as BOS neurons because the BOS is the stimulus that appears to drive them most strongly, there may be modifications of the BOS song that lead to an even more robust response. One can calculate linear approximations along different portions of the BOS to determine to which stimulus features the neuron is most sensitive. The calculated linear kernels would point toward stimulus features that enhance the neuron's response and away from stimulus features that suppress the response. The linear approximations may uncover preferred song features similar to the secondary kernel of section 5.3.2 shown in figure 6. One could obtain additional information about the nature of the selectivity by calculating linear approximation centered around songs near the BOS.

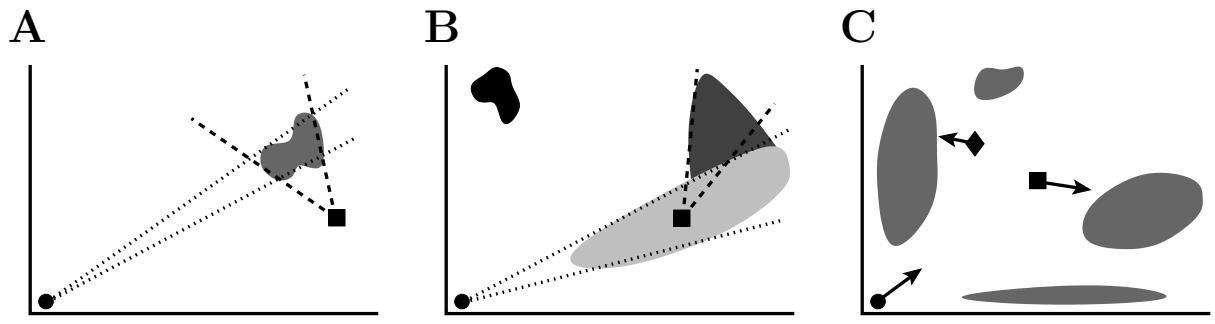

Figure 7: Schematic illustrations of the advantages of calculating multiple linear approximations of a neuron's response to a stimulus. **A**. Illustration of the benefit of calculating a linear approximation centered nearby a relevant part of the stimulus space. A highly selective neuron may respond to stimulus in a "selective region" (gray region) of stimulus space that is small compared to its distance from the origin. Starting from the origin (black circle) yields only a small range of directions (between dotted lines) that encounter the selective region, and many intermediate stimuli along those directions do not elicit a response. Starting at a nearby stimulus (black square) increases the range of directions (between dashed lines) and decreases the number of intermediate stimuli. **B**. An easily detected primary region (light gray) can help one detect an adjacent secondary region (dark gray) that drives a neuron even more strongly. If the primary region is detectable from the origin (dotted lines), one can use knowledge of the primary region to calculate a linear approximation centered on that region (black square). From the new starting point, the secondary region is detectable (dashed lines). A distant tertiary region (black shape in upper left) that drives the neuron even more strongly may still be undetectable by this procedure. **C**. The presence of multiple selective regions (gray shapes) may cause an initial linear approximation (arrow from black circle) to point in an average direction that misses all of the regions. Nonetheless, calculating a linear approximation based at a point (black square) in that direction may detect a region. Or, if other experiments indicated a point in stimulus space near one of the regions (black diamond), a linear approximation centered there may point toward that selective region.

6.2 Illustrations of benefits of multiple linear approximations

6.2.1 Highly selective neurons stymie linear approximations at origin

As illustrated by the example of section 5, it is difficult to obtain insight into a highly selective neuron's response with a linearization around the origin. (The typical way in which one employs a spike-triggered average is to compute a linearization around the origin.) When one samples stimuli symmetrically about the origin (or, for nonnegative stimuli, symmetrically within the positive orthant containing the set of directions containing all nonnegative components), it is hard to sample a sufficient number of stimuli that drive the cell in order to obtain a good estimate of a linear kernel (c.f., figure 5).

A simple sketch, shown in figure 7, can illustrate the difficulty of calculating a linear approximation to the response of highly selective neurons when one must start from the origin. Imagine that a neuron responded only to a small region of stimulus space (we'll call it the "selective region") and that the selective region was small compared to its distance

from the origin (figure 7A). In this case, only a small range of directions emanating from the origin would hit the selective region (in higher dimensions, the angle of suitable directions would correspond to high-dimensional analog of a cone). If one is searching in all directions, it will be difficult to locate the selective region. Even if one knew which direction to aim, precising characterizing the selective region starting at the origin will still be a challenge, as small differences in direction result in large differences in position around the selective region.

The problem may be much harder than pinpointing the right direction that points toward the selective region. For a highly selective cell, most stimuli in the direction of the selective region may not excite the cell. If the selective region is far from the origin relative to its size, as illustrated in figure 7A, then both the direction and the distance from the origin must be precisely specified to hit the selective region. In general, a strategy for sampling the stimulus space will sample nearby stimuli more frequently than distant stimuli.[5] Therefore, if the sampling length scale does not completely overshoot the selective region, the estimate of the linear approximation in the direction of the selective region will likely be heavily influenced by the neuron's response to stimuli between the selective region and the origin. If a neuron is highly selective, it many not respond at all to those intermediate stimuli (in fact, it's firing could even be suppressed by those stimuli). Without an increased firing rate occurring in the direction of the selective region, the linear kernel (7) will not pick out that direction.

6.2.2 Shifted linear approximations can reveal nature of selectivity

The nature of highly selective neurons may be more easily revealed if we allow the use of shifted linear approximations, i.e., those centered away from the origin. This application of shifted linear approximations is based on the idea that if we somehow knew to calculate a linear approximation around a point that was closer to the selective region, then we would be much more likely to detect the selective region. If we move the original stimulus $\hat{\mathbf{x}}$ from the point marked by the black circle near the origin to the point marked by the black square in figure 7A, the selective region both covers a much larger range of directions and becomes closer. A linear kernel calculated from the point marked by a square is much more likely to point in the direction of the selective region.

In general, determining a suitable choice for the original stimulus $\hat{\mathbf{x}}$ is not much easier than finding the selective region itself. If the neuron's response to stimuli were similar to that indicated in figure 7A, then one could exploit the ability to move the reference point of a linear approximation only if one had some other information to guide the choice of starting point (such as a stimulus that one theorized might be relevant to the type of neuron studied).

On the other hand, if a neuron's response to stimuli had structure similar to that depicted in figure 7B, then the practical use of shifted linear approximations is more evident. The light gray region in figure 7B represents a region of stimulus space that elicits a moderate response from the neuron. Although the region spans only a small range of angles from the perspective

[5] Any probability distribution over stimulus space must decay to zero once the distance from the origin is sufficiently large (otherwise the probability distribution cannot integrate to 1). Therefore, every technique to sample stimulus space must have some length scale beyond which the sample frequency will decay to zero with distance from the origin.

of the origin, the neuron will respond to most stimuli along those directions. Assume that this region is sufficiently large to be detected either by a linear approximation at the origin or other techniques. We'll refer to the light gray region as the "primary" region, as it would be the region stimulus space that would be primarily be visible to initial experiments probing the response of the neuron. Adjacent to the primary region is a darker gray region that we'll refer to as the "secondary region." Imagine that stimuli in the secondary region elicit an even stronger neuronal response than the primary region. Despite this strong response, the secondary region will be difficult to detect from the origin because the neuron does not respond to most stimuli along the direction from the origin to the secondary region. However, the primary region serves as a natural location around which to compute a linear approximation of the neuron's response. If one computed a linear approximation centered at the black square, the secondary region will be both close enough and span a large enough range of directions to be easily detected. A linear approximation computed at point marked by the black square will likely point toward the secondary region and reveal that the neuron actually prefers stimuli that are drawn from that region of stimulus space.

In our divisive normalization model (10), we designed the model to mimic the primary and secondary region of figure 7B. The primary region corresponded to stimuli that had a sufficient projection onto the primary kernel without too much projection onto the divisive kernels. The secondary region corresponded to stimuli that, in addition, had a positive projection onto the secondary kernel that outweighed any additional projection on the divisive kernels. Although the linear approximations we calculated from the origin had difficulty detecting the primary region, one could imagine detecting this region either with noise that better matches the statistics of the upper sweep "song" or through experiments probing the response of the neuron to natural "songs". Since we designed the model to be reminiscent of properties of BOS neurons, we imagine one detected the primary region through stimulating the neuron with the BOS that contained syllables in the primary region. With knowledge of the primary region, we were able to easily detect the secondary region via linearizations centered in the primary region.

If there happened to be tertiary region of stimulus space that drove the neuron even more strongly but was distant from the primary and secondary regions (e.g., the black region in upper left of figure 7B), it would remain invisible from linear approximations centered in either the primary or secondary regions. Unless a linear approximation in that region is calculated, the presence of a tertiary region will probably not be detected.

6.2.3 Multiple representations

Shifted linear approximations can capture how a neuron may be selective to different features depending on the region of stimulus space. For example, imagine that the stimulus space contains multiple distinct regions of stimuli that strongly drive a neuron, as schematized in figure 7C. If linear approximations were only calculated at the origin, the combined influence of these regions may result in the linear kernel indicating an average direction that does not actually point to any those regions (c.f., the linear kernels calculated at the origin in figure 2A). A local linear approximation centered around a point nearby one of these regions would reveal the stimulus direction to which the neuron is most sensitive in that neighborhood. If one calculated linear kernels at multiple points, the combined result could

yield a fuller picture of the nature of the neuron's response to the stimulus.

For this approach to be useful, one would need algorithms to determine around which additional points one should compute linear approximations. One possibility is that exploration of the neuron's response to rich stimuli, like the bird songs or other natural stimuli, may reveal multiple points where the neuron appears to response robustly. For example, a neuron may respond robustly to different points in a song or movie. One could compute linear approximations around each of those points to explore what ensemble of stimuli evoke strong responses from the neuron.

Another possiblity is to use previously calculated linear approximations to guide the choice for additional stimuli around which to compute linear approximations. For example, one could compute a second linear approximations along the line determined by the direction of the initial linear kernel (e.g., at the square in figure 7C). This approach was taken by Foldiak, who introduced a variant of the gradient ascent algorithm to search for stimuli that drove neurons in primary visual cortex the most strongly [7]. This algorithm was also used to search for sound spectra that maximize the firing rate of neurons in the primary auditory cortex [19].

One could also attempt to use the linear approximation to search for stimuli that maximize other functions of the spikes, such as mutual information between the stimulus and spikes. The key insight is that the linear approximation (5) in combination with the spiking model (3) can be used to estimate the local behavior of a function of the neuronal output. One could, for example, calculate in which direction the function increases most rapidly and explore the response of the neuron to stimuli in that direction. Alternatively, one could also interpolate among linear approximations based at different points in attempt to approximate the function over a larger range of stimulus values.

6.3 Comparison to other approaches

We have implemented an approach that computes linear approximations by correlating neuron's spike with the stimulus (i.e., reverse correlation or the spike-triggered average) [3, 17]. This approach has been widely used to probe the "receptive field" of sensory neurons [6, 10, 25, 23, 11, 30, 24]. Such approaches can be viewed as determining a one-dimensional subspace specified by the direction of the linear kernel (or filter) to which the neuron is most responsive [27]. By shifting perspective to looking for an affine subspace (a linear manifold or a linear subspace offset by a vector), we generalize the use of these methods to neurons whose global features cannot be captured by a low-dimensional linear subspace. We simply seek a one-dimensional affine subspace (the line $\mathbf{x}_s = \hat{\mathbf{x}} + s\mathbf{h}(\hat{\mathbf{x}})$) that captures the response properties of a neuron in a localized region of stimulus space. By restricting ourselves to a local view around an original stimulus $\hat{\mathbf{x}}$, we can analyze the response of highly nonlinear neurons and do need not postulate that the global response of a neuron is well approximated by the projection of the stimulus onto a one-dimensional subspace.

The idea of using correlation techniques to calculate a perturbation from the response to given a stimulus was introduced by Kvale and Schreiner [12]. Their work was based on using an m-sequence as the perturbation. Our work provides a general mathematical framework for using arbitrary perturbations to probe the nonlinear response properties of neurons.

The dependence of linear kernel estimates on the stimulus was a central result of Theunessen et al. [30], where they discovered systematic differences between linear kernels computed from natural versus synthetic stimulus ensembles. Christianson et al. [1] recently demonstrated how such differences in linear kernels could be caused by higher order statistics of the stimulus interacting with neuronal nonlinearities. This paper, in contrast, focuses on the effect of changing the mean stimulus by adding a constant stimulus to the noise.

An alternative method to go beyond the one-dimensional subspace spanned by the spike-triggered average is to employ the spike-triggered covariance [2, 26]. One can determine higher-dimensional subspaces spanned by significant eigenvectors of the spike-triggered covariance matrix, obtaining multiple kernels that capture the principle directions of that subspace. Since the spike-triggered covariance analysis goes beyond a single linear approximation, it represents another approach to give insight into the response properties of highly nonlinear neurons. Drawbacks of the spike-triggered covariance include the fact that it requires much more data than a spike-triggered average and that it will converge to the correct directions only if the noise is actually Gaussian [20, 27].

Other approaches use higher order statistical measures [20, 29] to estimate subspaces to which the neuron is most sensitive. One can also combine subspace estimation with models that relax the Poisson assumption in order capture history-dependent effects like refractory periods or burstiness [21, 22].

In principle, all of these approaches can be generalized to look for local affine subspaces centered around an original stimulus $\hat{\mathbf{x}}$. In this way, these analysis methods can be extended to analyze the response of highly nonlinear neurons, such as those that are highly selective to particular classes of stimuli. By taking advantage of the local linearity of arbitrary nonlinear differentiable functions, one can develop a set of tools that will reveal simplified local descriptions of a neuron's complicated response to a stimulus.

A Appendix

A.1 Calculating the sparse noise

We outline the calculation for determining a sparse noise stimulus that has at most k nonzero frequencies at each time bin. As each time bin is independent, we can, without loss of generality, consider a stimulus with only one time bin. Let N_f denotes the number of frequencies. Then, for $j = 1, 2, \ldots, N_f$, let $\hat{x}_j$ denote the jth frequency of the original stimulus $\hat{\mathbf{x}}$ and let Z_j denote the jth frequency of the noise $\mathbf{Z}$. We require that Z_j have standard deviation σ.

If the total stimulus $\hat{x}_j + Z_j$ could take on negative values, then creating such a noise is trivial. In a given realization, we randomly select k of the N_f frequencies: J_i, for $i = 1, 2, \ldots, k$. For each of those k frequencies, we let Z_{J_i} equal $\pm c$, each with probability $1/2$. In this way, Z_j has mean zero. We select the constant c so that each Z_j has standard deviation σ.

To determine c, note that, out of the $\binom{N_f}{k}$ possible ways to select k frequencies, a particular j will be included in $\binom{N_f-1}{k-1}$ of those combinations[6]. Therefore, each Z_j will be nonzero

[6]If we insist that a particular frequency is one of those chosen, then we must choose the remaining $k-1$

with probability

$$\frac{\binom{N_f-1}{k-1}}{\binom{N_f}{k}} = \frac{k}{N_f}.$$

Since Z_j will be c with probability $k/2N_f$ and $-c$ with probability $k/2N_f$, the variance of Z_j is

$$\operatorname{var} Z_j = (c)^2 \frac{k}{2N_f} + (-c)^2 \frac{k}{2N_f} = c^2 \frac{k}{N_f}.$$

For this variance to equal to σ^2, we set $c = \sigma\sqrt{N_f/k}$.

Since we require that the stimulus be nonnegative, we may need to modify this algorithm. As long as $\hat{x}_j \geq \sigma\sqrt{N_f/k}$, then $\hat{x}_j + Z_j \geq 0$ and we can choose the component of the noise Z_j using this algorithm without modification. Otherwise, we need to change the noise to keep the total stimulus nonnegative. The only change we will make is change the values that Z_j takes on when it is nonzero. In this way, we can change the values of Z_j without altering the distribution of the other Z_i for $i \neq j$.

If Z_j is chosen to be nonzero, then we will let $Z_j = a_j$ with probability 1/2 and let it be b_j with probability 1/2. The default values will be $a_j = -\sigma\sqrt{N_f/k}$ and $b_j = \sigma\sqrt{N_f/k}$, as above. However, if $\hat{x}_j < \sigma\sqrt{N_f/k}$, then we will decrease the absolute value of a_j so that $\hat{x}_j + a_j = 0$. After setting $a_j = -\hat{x}_j$, we will set the value of b_j so that Z_j still has standard deviation σ.

With these choices, the mean of Z_j is

$$E(Z_j) = a_j \frac{k}{2N_f} + b_j \frac{k}{2N_f} = \frac{(b_j - \hat{x}_j)k}{2N_f}.$$

The second moment is

$$E(Z_j^2) = a_j^2 \frac{k}{2N_f} + b_j^2 \frac{k}{2N_f} = \frac{(b_j^2 + \hat{x}_j^2)k}{2N_f},$$

so that the variance is

$$\operatorname{var} Z_j = E(Z_j^2) - E(Z_j)^2 = \frac{(b_j^2 + \hat{x}_j^2)k}{2N_f} - \frac{(b_j - \hat{x}_j)^2 k^2}{4N_f^2}$$

We set this variance equal to σ^2. After some algebra, the positive solution b_j is

$$b_j = \frac{-\hat{x}_j + \sqrt{\hat{x}_j^2 + 4\sigma^2\beta(N_f/k)^3 - \beta^2(N_f/k)^2\hat{x}_j^2}}{\beta N_f/k}, \tag{11}$$

where $\beta = 2 - k/N_f$.

In summary, in each time bin we select k frequencies of the noise to be nonzero. For each frequency j for which the noise is selected to be nonzero, we set Z_j to a_j with probability

frequencies from the other $N_f - 1$ options.

1/2 and to b_j with probability 1/2. If $\hat{x}_j \geq \sigma\sqrt{N_f/k}$, then we set $-a_j = b_j = \sigma\sqrt{N_f/k}$. Otherwise, we set $a_j = -\hat{x}_j$ and set b_j according to equation (11). In this way, Z_j has standard deviation σ and $\hat{x}_j + Z_j$ is always nonnegative. Note that if $\hat{x}_j$ is zero, then Z_j is nonzero only half of the times that the jth frequency is selected.

Equation (7) assumes that the noise is mean zero. If we adjust the values of the noise so that $a_j > -b_j$, then the noise Z_j as described here is not mean zero. Hence, in the calculation of (7), we subtract the mean from the noise $\mathbf{Z}$ and effectively add it to the original stimulus $\hat{\mathbf{x}}$.

A.2 The covariance matrix of the sparse noise

To calculate the linear kernel, we need to calculate $C_{\mathbf{Z}}$, the covariance matrix of Z_j, for (7). Since we select the noise independently for each time bins, the entries of $C_{\mathbf{Z}}$ corresponding to different time bins will be zero. Hence, it suffices to assume just one time bin as we did above.

As above, we denote the two nonzero possibilities for Z_j as a_j and b_j. We do not need to distinguish whether or not they are symmetric about zero. The mean value of Z_j is

$$E(Z_j) = \frac{(a_j + b_j)k}{2N_f}.$$

To calculate the second moment $E(Z_i Z_j)$, we need only need to enumerate the cases where both Z_i and Z_j are nonzero for $i \neq j$. The number of combinations of k frequencies that include both i and j is[7] $\binom{N_f-2}{k-2}$. Therefore, the probability that both Z_i and Z_j are nonzero is

$$\frac{\binom{N_f-2}{k-2}}{\binom{N_f}{k}} = \frac{k(k-1)}{N_f(N_f-1)}.$$

Each of the four possibilities of nonzero Z_i and Z_j occur with probability $\frac{k(k-1)}{4N_f(N_f-1)}$ and the second moment is

$$E(Z_i Z_j) = \frac{(a_i a_j + a_i b_j + b_i a_j + b_i b_j)k(k-1)}{4N_f(N_f-1)} = \frac{(a_i + b_i)(a_j + b_j)k(k-1)}{4N_f(N_f-1)}.$$

The covariance is

$$\begin{aligned}\operatorname{cov}(Z_i, Z_j) = E(Z_i Z_j) - E(Z_i)E(Z_j) &= \frac{(a_i + b_i)(a_j + b_j)k(k-1)}{4N_f(N_f-1)} - \frac{(a_i + b_i)(a_j + b_j)k^2}{4N_f^2} \\ &= -\frac{(a_i + b_i)(a_j + b_j)k(N_f - k)}{4N_f^2(N_f-1)}.\end{aligned}$$

We see that Z_i and Z_j are uncorrelated if either $a_i = -b_i$ or $a_j = -b_j$. Even though Z_i and Z_j are not independent (as they are from the same time bin), the symmetry about zero cancels any correlation. On other other, if both $\hat{x}_i$ and $\hat{x}_j$ are less than $\sigma\sqrt{N_f/k}$, then $a_i + b_i > 0$ and $a_j + b_j > 0$ as a consequence of the algorithm for choosing the a and b. The symmetry for both variables is broken, and Z_i and Z_j are negatively correlated.

[7] If we insist that the two frequencies i and j are chosen, then we must choose the remaining $k-2$ frequencies from the other $N_f - 2$ options.

Acknowledgments

We thank Teresa Nick and Steve Kerrigan for helpful discussions. This research was supported by the National Science Foundation grant DMS-0719724 (DQN).

References

[1] G. B. Christianson, M. Sahani, and J. F. Linden. The consequences of response nonlinearities for interpretation of spectrotemporal receptive fields. *J. Neurosci.*, 28:446–455, 2008.

[2] R. de Ruyter van Steveninck and W. Bialek. Real-time performance of a movement-sensitive neuron in the blowfly visual system: Coding and information transmission in short spike sequences. *Proc. Royal Soc. London B*, 234:379–414, 1988.

[3] E. DeBoer and P. Kuyper. Triggered correlation. *IEEE Trans. Biomed. Eng.*, 15:169–179, 1968.

[4] R. Desimone. Face-selective cells in the temporal cortex of monkeys. *J. Cogn. Neurosci.*, 3:1–8, 1991.

[5] A. J. Doupe and M. Konishi. Song-selective auditory circuits in the vocal control system of the zebra finch. *Proc. Natl. Acad. Sci. USA*, 88:11339–11343, 1991.

[6] J. J. Eggermont, P. I. M. Johannesma, and A. M. H. J. Aertsen. Reverse-correlation methods in auditory research. *Q. Rev. Biophysics*, 16:341–414, 1983.

[7] P. Foldiak. Stimulus optimization in primary visual cortex. *Neurocomputing*, 38-40:1217–1222, 2001.

[8] D. J. Heeger. Normalization of cell responses in cat striate cortex. *Visual Neuroscience*, 9:181–198, 1992.

[9] P. Janata and D. Margoliash. Gradual emergence of song selectivity in sensorimotor structures of the male zebra finch song system. *J. Neurosci.*, 19:5108–5118, 1999.

[10] J. P. Jones and L. A. Palmer. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *J. Neurophysiol.*, 58:1233–1258, 1987.

[11] D. J. Klein, D. A. Depireux, J. Z. Simon, and S. A. Shamma. Robust spectrotemporal reverse correlation for the auditory system: Optimizing stimulus design. *J. Comp. Neurosci.*, 9:85–111, 2000.

[12] M. Kvale and C. E. Schreiner. Perturbative m-sequences for auditory systems identification. *Acta Acustica united with Acustica*, 83:653–658, 1997.

[13] M. S. Lewicki. Intracellular characterization of song-specific neurons in the zebra finch auditory forebrain. *J. Neurosci.*, 16:5854–5863, 1996.

[14] D. Margoliash. Acoustic parameters underlying the responses of song-specific neurons in the white-crowned sparrow. *J. Neurosci.*, 3:1039–1057, 1983.

[15] D. Margoliash. Functional organization of forebrain pathways for song production and perception. *J. Neurobiol.*, 33:671–693, 1997.

[16] D. Margoliash and E. S. Fortune. Temporal and harmonic combination-sensitive neurons in the zebra finch's HVc. *J. Neurosci.*, 12:4309–4326, 1992.

[17] P. N. Marmarelis and V. Z. Marmarelis. *Analysis of physiological systems: the white noise approach*. Plenum Press, NewYork, 1978.

[18] R. Mooney. Different subthreshold mechanisms underlie song selectivity in identified hvc neurons of the zebra finch. *J. Neurosci*, 20:5420–5436, 2000.

[19] K. N. O'Connor, C. I. Petkov, and M. L. Sutter. Adaptive stimulus optimization for auditory cortical neurons. *J. Neurophysiol.*, 94:4051–4067, 2005.

[20] L. Paninski. Convergence properties of three spike-triggered analysis techniques. *Network: Comput. Neural Syst.*, 14:437–464, 2003.

[21] L. Paninski. Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Comput. Neural Syst.*, 15:243–262, 2004.

[22] L. Paninski, J. W. Pillow, and E. P. Simoncelli. Maximum likelihood estimation of a stochastic integrate-and-fire neural encoding model. *Neural Comp.*, 16:2533–2561, 2004.

[23] R. C. Reid and J. M. Alonso. Specificity of monosynaptic connections from thalamus to visual cortex. *Nature*, 378:281–284, 1995.

[24] D. L. Ringach, M. J. Hawken, and R. Shapley. Receptive field structure of neurons in monkey primary visual cortex revealed by stimulation with natural image sequences. *J. Vision*, 2:12–24, 2002.

[25] H. M. Sakai and K. Naka. Signal transmission in the catfish retina: V. sensitivity and circuit. *J. Neurophysiol.*, 58:1329–1350, 1987.

[26] O. Schwartz, E. J. Chichilnisky, and E. P. Simoncelli. Characterizing neural gain control using spike-triggered covariance. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 269–276, Cambridge, MA, 2002. MIT Press.

[27] O. Schwartz, J. W. Pillow, N. C. Rust, and E. P. Simoncelli. Spike-triggered neural characterization. *J. Vision*, 6:484–507, 2006.

[28] O. Schwartz and E. P. Simoncelli. Natural signal statistics and sensory gain control. *Nat. Neurosci.*, 4:819–825, 2001.

[29] T. Sharpee, N. C. Rust, and W. Bialek. Analyzing neural responses to natural signals: maximally informative dimensions. *Neural Comput.*, 16:223–250, 2004.

[30] F. E. Theunissen, K. Sen, and A. J. Doupe. Spectral-temporal receptive fields of nonlinear auditory neurons obtained using natural sounds. *J. Neurosci.*, 20:23152233, 2000.