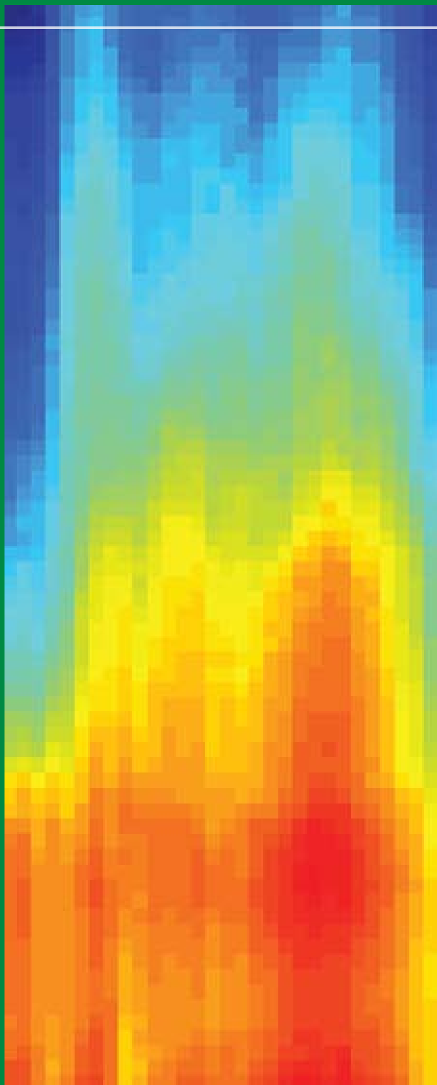
2008

Short Course III

Neural Signal Processing: Quantitative Analysis of Neural Activity

Organized by Partha Mitra, PhD



SOCIETY FOR NEUROSCIENCE

NEUROSCIENCE 2008

Short Course III

Neural Signal Processing:
Quantitative Analysis of Neural Activity

Organized by Partha Mitra, PhD

SOCIETY FOR NEUROSCIENCE

Table of Contents

Introduction

In neuroscience, the dynamical patterns of electrical activity of neurons provide a crucial bridge between cellular and psychological/behavioral levels of analysis. During the last decade or so, a significant amount of research has gone into the development of signal processing tools to quantify neuronal dynamics. These methods have been applied to a wide variety of neural signals, including EEG/MEG and single electrode recordings, as well as more contemporary multielectrode recordings or imaging techniques. These quantitative methods have now reached a certain level of maturity, indicated by common measures and signal processing algorithms used in research papers, shared analysis software (such as Chronux), and pedagogical material in courses. Apart from bridging cellular and systems levels of analysis, these methods provide a critical link between experimental data and theoretical models.

This short course will provide a survey of topics from this field, including methods for analyzing point process signals (spike trains) and continuous process signals (LFP, EEG, behavioral recordings). Nonparametric smoothing and spectral estimation techniques will be complemented by parametric stochastic process models. Pedagogical lectures in the morning will be followed by tutorial exercises participants can carry out on their own laptop computers using data sets and analysis software, which will be provided.

Course Organizer: Partha Mitra, PhD, Cold Spring Harbor Laboratory. Faculty: Hemant Bokil, PhD, Cold Spring Harbor Laboratory; Uri Eden, PhD, Boston University; Robert Kass, PhD, Department of Statistics, Carnegie Mellon University; David Kleinfeld, PhD, Department of Physics, University of California, San Diego; Bijan Pesaran, PhD, Center for Neural Science, New York University; Keith Purpura, PhD, Department of Neurology & Neuroscience, Weill Cornell Medical College; Sridevi Sarma, PhD, Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology; Andrew Sornborger, PhD, Department of Mathematics, University of Georgia; and Ofer Tchernichovski, PhD, Department of Biology, City College of New York.

Spectral Analysis for Neural Signals

Bijan Pesaran, PhD

Center for Neural Science, New York University
New York, New York

NOTES

Introduction

This chapter introduces concepts fundamental to spectral analysis and applies spectral analysis to characterize neural signals. Spectral analysis is a form of time series analysis and concerns a series of events or measurements that are ordered in time. The goal of such an analysis is to quantitatively characterize the relationships between events and measurements in a time series. This quantitative characterization is needed to derive statistical tests that determine how time series *differ* from one another and how they are *related* to one another. Time series analysis comprises two main branches: time-domain methods and frequency-domain methods. Spectral analysis is a frequency-domain method for which we will use the multitaper framework (Thomson, 1982; Percival and Walden, 1993). The treatment here also draws on other sources (Brillinger, 1978; Jarvis and Mitra, 2001; Mitra and Bokil, 2007).

In this chapter, we will focus on relationships within and between one and two time series, known as univariate and bivariate time series. Throughout this discussion, we will illustrate the concepts with experimental recordings of spiking activity and local field potential (LFP) activity. The chapter on Multivariate Neural Data Sets will extend the treatment to consider several simultaneously acquired time series that form a multivariate time series, such as in imaging experiments. The chapter on "Application of Spectral Methods to Representative Data Sets in Electrophysiology and Functional Neuroimaging" will review some of this material and present additional examples.

First we begin by motivating a particular problem in neural signal analysis that frames the examples in this chapter. Second, we introduce signal processing and the Fourier transform and discuss practical issues related to signal sampling and the problem of aliasing. Third, we present stochastic processes and their characterization through the method of moments. The moments of a stochastic process can be characterized in both the time domains and frequency domains, and we will discuss the relation between these characterizations. Subsequently, we present the problem of scientific inference, or hypothesis testing, in spectral analysis through the consideration of error bars. We finish by considering an application of spectral analysis involving regression.

Motivation

When a microelectrode is inserted into the brain, the main features that are visible in the extracellular potential it measures are the spikes and the rhythms they ride on. The extracellular potential results from currents flowing in the extracellular space, which in turn are produced by transmembrane potentials in local populations of neurons. These cellular events can be fast, around 1 ms for the action potentials that appear as spikes, and slow, up to 100 ms, for the synaptic potentials that predominantly give rise to the LFP. How spiking and LFP activity encode the sensory, motor, and cognitive processes that guide behavior, and how these signals are related, are fundamental, open questions in neuroscience (Steriade, 2001; Buzsaki, 2006). In this chapter, we will illustrate these analysis techniques using recordings of spiking and LFP activity in macaque parietal cortex during the performance of a delayed look-and-reach movement to a peripheral target (Pesaran et al., 2002). This example should not be taken to limit the scope of potential applications, and other presentations will motivate other examples.

The Basics of Signal Processing

Spiking and LFP activity are two different kinds of time series, and all neural signals fall into one of these two classes. LFP activity is a continuous process and consists of a series of continuously varying voltages in time, x_t. Spiking activity is a point process, and, assuming that all the spike events are identical, consists of a sequence of spike times. The counting process, N_t, is the total number of events that occur in the process up to a time, t. The mean rate of the process, λ, is given by the number of spike events divided by the duration of the interval. If we consider a sufficiently short time interval, δt = 1 ms, either a spike event occurs or it does not. Therefore, we can represent a point process as the time derivative of the counting process, dN_t, which gives a sequence of delta functions at the precise time of each spike, t_n. We can also represent the process as a sequence of times between spikes, $\tau_n = t_{n+1} - t_n$, which is called an interval process. These representations are equivalent but capture different aspects of the spiking process. We will focus on the counting process, dN_t. $dN_t = 1 - \lambda\delta t$ when there is a spike and $dN_t = -\lambda\delta t$ elsewhere. Note that these expressions correct for the mean firing rate of the process. As we will see, despite the differences between point and continuous processes, spectral analysis treats them in a unified way. The following sections present some basic notions that underlie statistical signal processing and time series analysis.

Fourier transforms

Time series can be represented by decomposing them into a sum of elementary signals. One domain for signals, which we call the time domain, is simply each point in time. The time series is represented by its amplitude at each time point. Another domain is the frequency domain and consists of sinusoidal

NOTES

functions, one for each frequency. The process is represented by its amplitude and phase at each frequency. The time and frequency domains are equivalent, and we can transform signals between them using the Fourier transform. Fourier transforming a signal that is in the time domain, x_t, will give the values of the signal in the frequency domain, $\tilde{x}(f)$. The tilde denotes a complex number with amplitude and phase.

$$\tilde{x}(f) = \sum_{t=1}^{N} \exp(-2\pi i f t_n)$$

Inverse Fourier transforming $\tilde{x}(f)$ transforms it to the time domain. To preserve all the features in the process, these transforms need to be carried out over an infinite time interval. However, this is never realized in practice. Performing Fourier transforms on finite duration data segments distorts features in the signal and, as we explain below, spectral estimation employs data tapers to limit these distortions.

Nyquist frequency, sampling theorem, and aliasing

Both point and continuous processes can be represented in the frequency domain. When we sample a process, by considering a sufficiently short interval in time, δt, and measuring the voltage or the presence or absence of a spike event, we are making an assumption about the highest frequency in the process. For continuous processes, the sampling theorem states that when we sample an analog signal that is band-limited, so that it contains no frequencies greater than the Nyquist rate (B Hz), we can perfectly reconstruct the original signal if we sample at a sampling rate, $F_s = \frac{1}{\delta t}$, of at least $2B$ Hz. The original signal is said to be band-limited because it contains no energy outside the frequency band given by the Nyquist rate. Similarly, once we sample a signal at a certain sampling rate, F_s, the maximum frequency we can reconstruct from the sampled signal is called the Nyquist frequency, $\frac{1}{2}F_s$.

The Nyquist frequency is a central property of all sampled, continuous processes. It is possible to sample the signal more frequently than the bandwidth of the original signal, with a sampling rate greater than twice the Nyquist rate, without any problems. This is called oversampling. However, if we sample the signal at less than twice the Nyquist rate, we cannot reconstruct the original signal without errors. Errors arise because components of the signal that exist at a higher frequency than $\frac{1}{2}F_s$ become aliased into lower frequencies by the process of sampling the signal. Importantly, once the signal has been sampled at F_s, we can no longer distinguish between continuous processes that have frequency components greater than the Nyquist frequency of $\frac{1}{2}F_s$. To avoid the problem of aliasing signals from high frequencies into lower frequencies by digital sampling, an anti-aliasing filter is often applied to continuous analog signals before sampling. Anti-aliasing filters act to low-pass the signal at a frequency less than the Nyquist frequency of the sampling.

Sampling point processes does not lead to the same problems as sampling continuous processes. The main consideration for point processes is that the sampled point process be orderly. Orderliness is achieved by choosing a sufficiently short time interval so that each sampling interval has no more than one event. Since this means that there are only two possible outcomes to consider at each time step, 0 and 1, analyzing an orderly point process is simpler than analyzing a process that has multiple possible outcomes, such as 0, 1, and 2, at each time step.

Method of moments for stochastic processes

Neural signals are variable and stochastic owing to noise and the intrinsic properties of neural firing. Stochastic processes (also called random processes) can be contrasted with deterministic processes, which are perfectly predictable. Deterministic processes evolve in exactly the same way from a particular point. In contrast, stochastic processes are described by probability distributions that govern how they evolve in time. Stochastic processes evolve to give a different outcome, even if all the samples of the process originally started from the same point. This is akin to rolling dice on each trial to determine the neural activity. Each roll of the dice is a realization from a particular probability distribution, and it is this distribution that determines the properties of the signal. When we measure neural signals to repeated trials in an experiment, we assume that the signals we record on each trial are different realizations or outcomes of the same underlying stochastic process.

Another powerful simplification is to assume that the properties of the stochastic process generating the neural signals within each trial are stationary and that their statistical properties don't change with time—even within a trial. This is clearly not strictly true for neural signals because of the nonstationarities in behavior. However, under many circumstances, a reasonable procedure is to weaken the stationarity assumption to short-time stationarity. Short-time stationarity assumes that the properties of the stochastic process have stationarity for short time intervals, say 300-400 ms, but change on longer time scales. In general, the window for spectral analysis is chosen to be as short as possible to remain consistent with the spectral structure of the data; this window is then translated in time. Fundamental to time-frequency representations is the uncertainty principle, which sets the bounds for simultaneous resolution in time and frequency. If the time-frequency plane is "tiled" so as to provide time and frequency resolutions $\Delta t = N$ by $\Delta f = W$, then $NW \geq 1$. We can then estimate the statistical properties of the stochastic process by analyzing short segments of data and, if necessary and reasonable, averaging the results across many repetitions or trials. Examples of time-frequency characterizations are given below. Note that this presentation uses "normalized" units. This means that we assume the sampling rate to be 1 and the Nyquist frequency interval to range from –½ to ½. The chapter Application of Spectral Methods to Representative Data Sets in Electrophysiology and Functional Neuroimaging presents the relationships below in units of time and frequency.

Spectral analysis depends on another assumption: that the stochastic process which generates the neural signals has a spectral representation.

$$x_t = \int_{-1/2}^{1/2} \tilde{x}(f)\exp(2\pi i f t)df$$

Remarkably, the same spectral representation can be assumed for both continuous processes (like LFP activity) and point processes (like spiking activity), so the Fourier transform of the spike train, t_n, is as follows:

$$d\tilde{N}(f) = \sum_{n=1}^{N} \exp(2\pi i f t_n)$$

The spectral representation assumes that underlying stochastic processes generating the data exist in the frequency domain, but that we observe their realizations as neural signals, in the time domain. As a result, we need to characterize the statistical properties of these signals in the frequency domain: This is the goal of spectral analysis.

The method of moments characterizes the statistical properties of a stochastic process by estimating the moments of the probability distribution. The first moment is the mean; the second moments are the variance and covariance (for more than one time series), and so on. If a stochastic process is a Gaussian process, the mean and variance or covariances completely specify it. For the spectral representation, we are interested in the second moments. The spectrum is the variance of the following process:

$$S_X(f)\delta(f-f') = E[\tilde{x}^*(f)\tilde{x}(f')]$$
$$S_{dN}(f)\delta(f-f') = E[d\tilde{N}^*(f)d\tilde{N}(f')]$$

The delta function indicates that the process is stationary in time. The asterisk denotes complex conjugation. The cross-spectrum is the covariance of two processes:

$$S_{XY}(f)\delta(f-f') = E[\tilde{x}^*(f)\tilde{y}(f')]$$

The coherence is the correlation coefficient between each process at each frequency and is simply the covariance of the processes normalized by their variances.

$$C_{XY}(f) = \frac{S_{XY}(f)}{\sqrt{S_x(f)S_y(f)}}$$

This formula represents the cross-spectrum between the two processes, divided by the square root of the spectrum of each process. We have written this expression for two continuous processes; analogous expressions can be written for pairs of point-continuous processes or point-point processes by substituting the appropriate spectral representation. Also, we should note that the assumption of stationarity applies only to the time interval during which we carry out the expectation.

Multitaper spectral estimation

The simplest estimate of the spectrum, called the *periodogram*, is proportional to the square of the data sequence, $|\tilde{x}_{tr}(f)|^2$. This spectral estimate suffers from two problems. The first is the problem of bias. This estimate does not equal the true value of the spectrum unless the data length is infinite. Bias arises because signals at different frequencies are mixed together and "blurred." This bias comes in two forms: narrow-band bias and broad-band bias. *Narrow-band bias* refers to bias in the estimate due to mixing signals at different nearby frequencies. *Broad-band bias* refers to mixing signals at different frequencies at distant frequencies. The second problem is the problem of variance. Even if the data length were infinite, the periodogram spectral estimate would simply square the data without averaging. As a result, it would never converge to the correct value and would remain inconsistent.

Recordings of neural signals are often sufficiently limited so that bias and variance can present major limitations in the analysis. Bias can be reduced, however, by multiplying the data by a data taper, w_t, before transforming to the frequency-domain, as follows:

$$\tilde{x}(f) = \sum_{t=1}^{N} w_t x_t \exp(-2\pi i f t)$$

Using data tapers reduces the influence of distant frequencies at the expense of blurring the spectrum over nearby frequencies. The result is an increase in narrow-band bias and a reduction in broad-band bias. This practice is justified under the assumption that the true spectrum is locally constant and approximately the same for nearby frequencies. Variance is usually addressed by averaging overlapping segments of the time series. Repetitions of the experiment also give rise to an ensemble over which the expectation can be taken, but this precludes the assessment of single-trial estimates.

An elegant approach toward the solution of both the above problems has been offered by the multitaper spectral estimation method, in which the data are multiplied by not one, but several, orthogonal tapers and Fourier-transformed in order to obtain the basic quantity for further spectral analysis. The simplest example of the method is given by the direct multitaper estimate, $S_{MT}(f)$, defined as the average of individual tapered spectral estimates,

$$S_{MT}(f) = \frac{1}{K} \sum_{k=1}^{K} |\tilde{x}_k(f)|^2$$

$$\tilde{x}_k(f) = \sum_{t=1}^{N} w_t(k) x_t \exp(-2\pi i f t)$$

The $w_t(k)$ ($k = 1, 2, \ldots, K$) constitute K orthogonal taper functions with appropriate properties. A particular choice for these taper functions, with optimal spectral concentration properties, is given by the discrete prolate spheroidal sequences, which we will call "Slepian functions" (Slepian and Pollack, 1961). Let $w_t(k, W, N)$ be the kth Slepian function of length N and frequency bandwidth parameter W. The Slepians would then form an orthogonal basis set for sequences of length, N, and be characterized by a bandwidth parameter W. The important feature of these sequences is that, for a given bandwidth parameter W and taper length N, $K = 2NW - 1$ sequences, out of a total of N, each having their energy effectively concentrated within a range $[-W, W]$ of frequency space.

Consider a sequence w_t of length N whose Fourier transform is given by the formula

$U(f) = \sum_{t=1}^{N} w_t \exp(-2\pi i f t)$. Then we can consider the problem of finding sequences w_t so that the spectral amplitude $U(f)$ is maximally concentrated in the interval $[-W, W]$. Maximizing this concentration parameter, subject to constraints, yields a matrix eigenvalue equation for $w_t(k, W, N)$. The eigenvectors of this equation are the Slepians. The remarkable fact is that the first $2NW$ eigenvalues $\lambda_k(N,W)$ (sorted in descending order) are each approximately equal to 1, while the remainder approximate zero. The Slepians can be shifted in concentration from $[-W, W]$ centered around zero frequency to any nonzero center frequency interval $[f_0 - W, f_0 + W]$ by simply multiplying by the appropriate phase factor $\exp(2\pi i f_0 t)$—an operation known as *demodulation*.

The usual strategy is to select the desired analysis half-bandwidth W to be a small multiple of the Raleigh frequency $1/N$, and then to take the leading $2NW - 1$ Slepian functions as data tapers in the multitaper analysis. The remaining functions have progressively worsening spectral concentration properties. For illustration, in the left column of Figure 1, we show the first four Slepian functions for $W = 5/N$. In the right column, we show the time series example from the earlier subsection multiplied by each of the successive data tapers. In the left column of Figure 2, we show the spectra of the data tapers themselves, displaying the spectral concentration property. The vertical marker denotes the bandwidth parameter W. Figure 2 also shows the magnitude-squared Fourier transforms of the tapered time series presented in Figure 1. The arithmetic average of these spectra for $k = 1, 2, \ldots, 9$ (note that only 4 of 9 are shown in Figs. 1 and 2) gives a direct multitaper estimate of the underlying process.

Figure 3A shows the periodogram estimate of the spectrum based on a single trial of LFP activity during the delayed look-and-reach task. The variability in the estimate is significant. Figure 3B presents the multitaper estimate of the spectrum on the same data with $W = 10$ Hz, averaged across 9 tapers. This

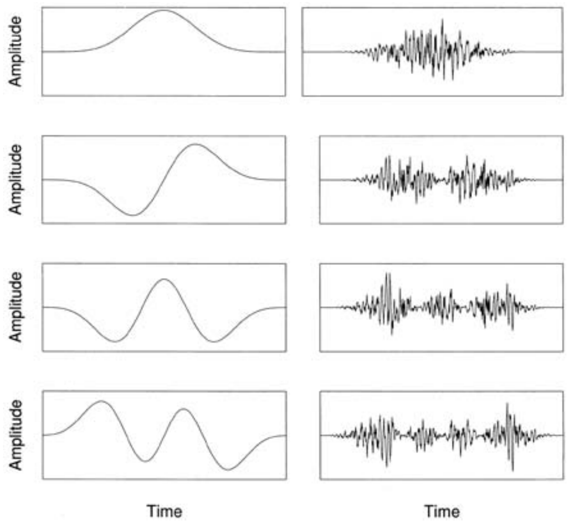


Figure 1. Slepian functions in the time domain. Left panels: First four Slepian functions for *NW* = 5. Right panels: Data sequence multiplied by each Slepian data taper on left.

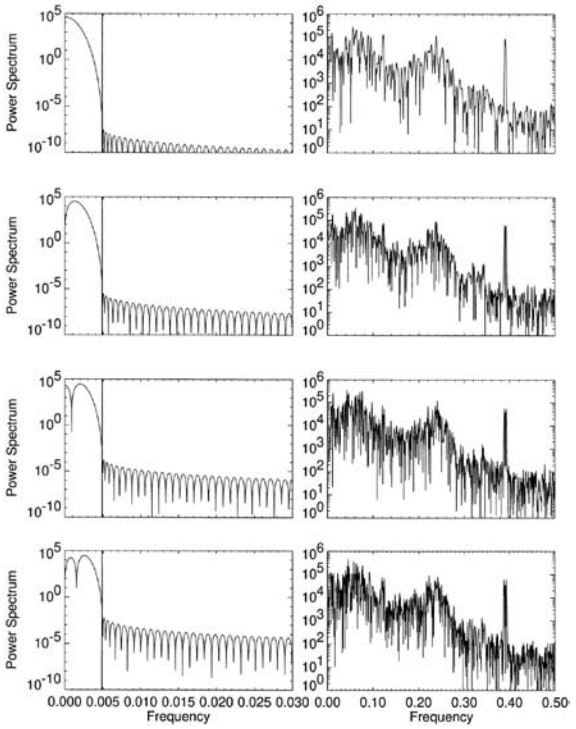


Figure 2. Slepian functions in the frequency domain. Left panel: spectra of Slepian functions from left panels of Figure 1. Right panel: spectra of data from right panels of Figure 1.

estimate is much smoother and reveals the presence of two broad peaks in the spectrum, at 20 Hz and 60 Hz. Figure 3C shows the multitaper spectrum estimate on the same data with *W* = 20 Hz. This estimate is even smoother than the 10 Hz, which reflects the increased number of tapers available to

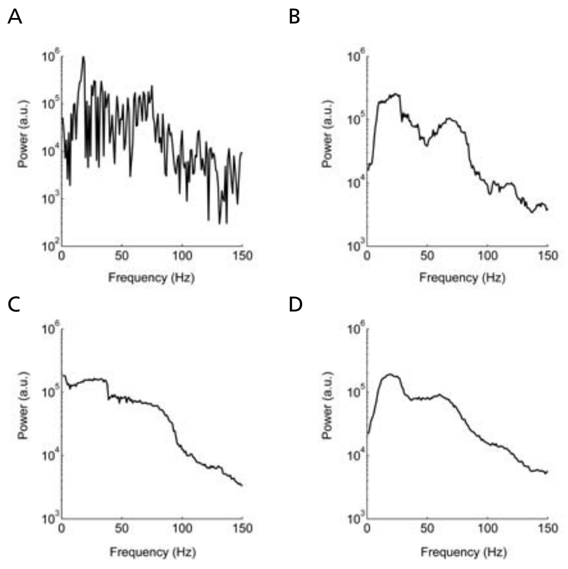


Figure 3. Spectrum of LFP activity in macaque lateral intraparietal area (LIP) during delay period before a saccade-and-reach to preferred direction. *A*, Single trial, 500 ms periodogram spectrum estimate. *B*, Single trial, 500 ms, 10 Hz multitaper spectrum estimate. *C*, Single trial, 500 ms, 20 Hz multitaper spectrum estimate. *D*, Nine-trial average, 500 ms, 10 Hz multitaper spectrum estimate. a.u. = arbitrary units.

average across (19 tapers instead of 9). However, the assumption that the spectrum is constant with the 20 Hz bandwidth is clearly wrong and leads to noticeable distortion in the spectrum, in the form of a narrow-band bias. Figure 3*D* shows the multitaper estimate with bandwidth *W* = 10 Hz averaged across 9 trials. Compared with Figure 3*B*, this estimate is noticeably smoother and contains the same frequency resolution. This series illustrates the advantages of multitaper estimates and how they can be used to improve spectral resolution.

Bandwidth selection

The choice of the time window length *N* and the bandwidth parameter *W* is critical for applications. No simple procedure can be given for these choices, which in fact depend on the data set at hand, and are best made iteratively using visual inspection and some degree of trial and error. 2*NW* gives the number of Raleigh frequencies over which the spectral estimate is effectively smoothed, so that the variance in the estimate is typically reduced by 2*NW*. Thus, the choice of *W* is a choice of how much to smooth. In qualitative terms, the bandwidth parameter should be chosen to reduce variance while not overly distorting the spectrum by increasing narrow-band bias. This can be done formally by trading off an appropriate weighted sum of the estimated variance and bias. However, as a rule, we find fixing the time bandwidth product *NW* at a small number (typically

NOTES

3 or 4), and then varying the window length in time until sufficient spectral resolution is obtained, to be a reasonable strategy. It presupposes that the data are examined in the time-frequency plane so that N may be significantly smaller than the total data length.

Figure 4 illustrates these issues using two spectrogram estimates of the example LFP activity averaged across 9 trials. Each trial lasts approximately 3 s and consists of a 1 s baseline period, followed by a 1–1.5 s delay period, during which a movement is being planned. The look-reach movement is then executed. Each spectrogram is shown with time on the horizontal axis, frequency on the vertical axis, and power color-coded on a log base-10 scale. Figure 4*A* shows the spectrogram estimated using a 0.5 s duration analysis window and a 10 Hz bandwidth. The time-frequency tile this represents is shown in the white rectangle. This estimate clearly shows the sustained activity following the presentation of the spatial cue at 0 s that extends through the movement's execution. Figure 4*B* shows a spectrogram of the same data estimated using a 0.2 s duration analysis window and a 25 Hz bandwidth. The time-frequency tile for this estimate has the same area as Figure 4*A*, so each estimate has the same number of degrees of freedom. However, there is great variation in the time-frequency resolution trade-off between these estimates: Figure 4*B* better captures the transients in the signal, at the loss of significant frequency resolution that distorts the final estimate. Ultimately, the best choice of time-frequency resolution will depend on the frequency band of interest, the temporal dynamics in the signal, and the number of trials available for increasing the degrees of freedom of a given estimate.

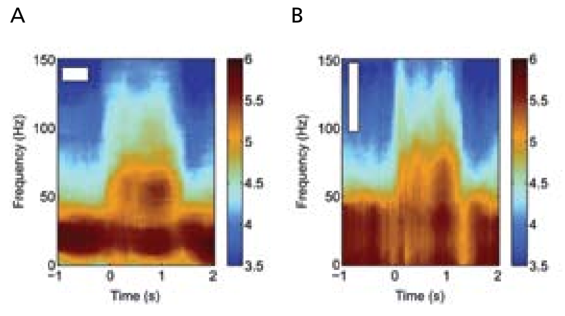


Figure 4. Spectrogram of LFP activity in macaque LIP averaged across 9 trials of a delayed saccade-and-reach task. Each trial is aligned to cue presentation, which occurs at 0 s. Saccade and reach are made at around 1.2 s. ***A***, Multitaper estimate with duration of 500 ms and bandwidth of 10 Hz. ***B***, Multitaper estimate with duration of 200 ms and bandwidth 25 Hz. White rectangle shows then time-frequency resolution of each spectrogram. The color bar shows the spectral power on a log scale in arbitrary units.

Calculating error bars

The multitaper method confers one important advantage: It offers a natural way of estimating error bars corresponding to most quantities obtained in time series analysis, even if one is dealing with an individual instance within a time series. Error bars can be constructed using a number of procedures, but broadly speaking, there are two types. The fundamental notion common to both types of error bars is the local frequency ensemble. That is, if the spectrum of the process is locally flat over a bandwidth $2W$, then the tapered Fourier transforms $\tilde{x}_k(f)$ constitute a statistical ensemble for the Fourier transform of the process at the frequency, f_o. This locally flat assumption and the orthogonality of the data tapers mean that the $\tilde{x}_k(f)$ are uncorrelated random variables having the same variance. This provides one way of thinking about the direct multitaper estimate presented in the previous sections: The estimate consists of an average over the local frequency ensemble.

The first type of error bar is the asymptotic error bar. For large N, $\tilde{x}_k(f)$ may be assumed to be asymptotically, normally distributed under some general circumstances. As a result, the estimate of the spectrum is asymptotically distributed according to a χ^2_{dof} distribution scaled by $\frac{S(f)}{dof}$. The number of degrees of freedom (dof) is given by the total number of data tapers averaged to estimate the spectrum. This would equal the number of trials multiplied by the number of tapers.

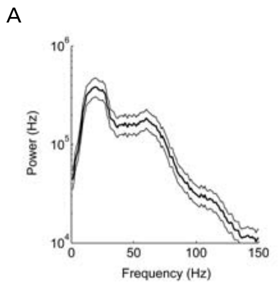
For the second type of error bar, we can use the local frequency ensemble to estimate jackknife error bars for the spectra and all other spectral quantities (Thomson and Chave, 1991; Wasserman, 2007). The idea of the jackknife is to create different estimates by, in turn, leaving out a data taper. This creates a set of spectral estimates that forms an empirical distribution. A variety of error bars can be constructed based on such a distribution. If we use a variance-stabilizing transformation, the empirical distribution can be well approximated using a Gaussian distribution. We can then calculate error bars according to the normal interval by estimating the variance of the distribution and determining critical values that set the error bars. Inverting the variance-stabilizing transformation gives us the error bars for the original spectral estimate. This is a standard tool in statistics and provides a more conservative error bar than the asymptotic error bar. Note that the degree to which the two error bars agree constitutes a test of how well the empirical distribution follows the asymptotic distribution. The variance-stabilizing transformation for the spectrum is the logarithm. The variance-stabilizing transformation for the coherence, the magnitude of the coherency, is the arc-tanh.

As an example, Figure 5*A* shows asymptotic and Figure 5*B* empirical jackknife estimates of the spectral estimate illustrated in Figure 3*D*. These are 95% confidence intervals and are largely the same between the two estimates. This similarity indicates that, for these data, the sampling distribution of the spectral estimate follows the asymptotic distribution across trials and data tapers. If we were to reduce the estimate's number of degrees of freedom by reducing the number of trials or data tapers, we might expect to see more deviations between the two estimates, with the empirical error bars being larger than the asymptotic error bars.

Correlation functions

Neural signals are often characterized in terms of correlation functions. Correlation functions are equivalent to computing spectral quantities but with important statistical differences. For stationary processes, local error bars can be imposed for spectral estimates in the frequency domain. This is not true for correlation functions, even assuming stationarity, because error bars for temporal correlation functions are nonlocal. Nonlocality in the error bars means that uncertainty about the correlation function at one lag is influenced by the value of the correlation function across other lags. The precise nature of the nonlocality relies on the temporal dependence within the underlying process. Consequently, correlation function error bars must be constructed by assuming there are no dependencies between different time bins. This is a far more restrictive assumption than the one holding that neighboring frequencies are locally flat and rarely achieved in practice. Other problems associated with the use of correlation functions are that if the data contain oscillatory components, they are compactly represented in frequency space and lead to nonlocal effects in the correlation function. Similar arguments apply to the computation of correlation functions for point and continuous processes. One exception is for spiking examples in which there are sharp features in the time-domain correlation functions, e.g., owing to monosynaptic connections.

Figure 6 illustrates the difference between using spectral estimates and correlation functions. Figure 6A shows the spectrum of spiking activity recorded in macaque parietal cortex during a delay period before a coordinated look-and-reach. The duration of the spectral estimate is 500 ms, the bandwidth is 30 Hz, and the activity is averaged over nine trials. Thin lines show the empirical 95% confidence intervals. Figure 6B shows the auto-correlation function for the same data, revealing some structure around short lags and inhibition at longer lags. There is a hint of some ripples, but the variability in the estimate is too large to see them clearly. This is not too surprising, because the correlation function estimate is analogous to the periodogram spectral estimate, which also suffers from excess statistical variability. In contrast, the spectrum estimate clearly reveals the presence of significant spectral suppression and a broad spectral peak at 80 Hz. The dotted line shows the expected spectrum from a Poisson process having the same rate.



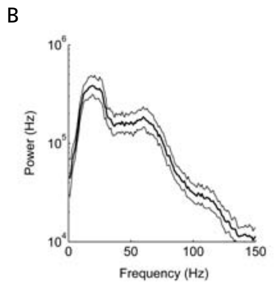
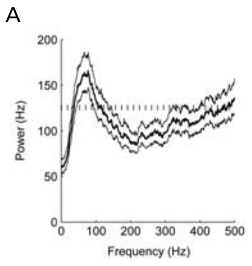


Figure 5. 95% confidence error bars for LFP spectrum shown in Figure 3*D*. ***A***, Asymptotic error bars assuming chi-squared distribution. ***B***, Empirical error bars using leave-one-out jackknife procedure.



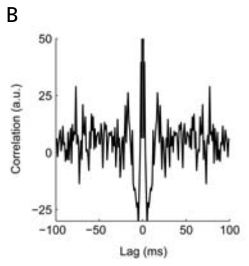


Figure 6. Spike spectrum and correlation function of spiking activity in macaque LIP during delay period before a saccade-and-reach to the preferred direction. ***A***, Multitaper spectrum estimate duration of 500 ms; bandwidth 15 Hz averaged across 9 trials. Thin lines show 95% confidence empirical error bars using leave-one-out procedure. Dotted horizontal line shows firing rate. ***B***, Correlation function estimate from which shift predictor has been subtracted. a.u. = arbitrary units.

Coherence

The idea of a local frequency ensemble motivates multitaper estimates of the coherence between two-point or continuous processes. Given two time series and the corresponding multiple tapered Fourier transforms $\tilde{x}_k(f)$, $\tilde{y}_k(f)$, the following direct estimates can be defined for the coherence function:

$$C_{XY}(f) = \frac{\frac{1}{K}\sum_k \tilde{x}_k^*(f)\tilde{y}_k(f)}{\sqrt{S_x(f)S_y(f)}}$$

This definition allows us to estimate the coherence from a single trial. Estimating the coherence presents many of the same issues as estimating the spectrum, except that more degrees of freedom are needed to ensure a reasonable estimate. In common with spectrum estimates, the duration and bandwidth of the estimator need to be chosen to allow sufficient degrees of freedom in the estimator. Increasing the number of trials will increase the effective resolution of the estimate.

Figure 7 shows the coherence and correlations between two simultaneously recorded spike trains from macaque parietal cortex averaged over nine trials. Figure 7A shows the coherence estimated with 16 Hz bandwidth. The horizontal dotted line represents expected coherence for this estimator when there is no coherence between the spike trains. The coherence significantly exceeds this threshold, as shown by the 95% confidence intervals, in a broad frequency band. Figure 7*B* illustrates the coherence estimated with a 30 Hz bandwidth. The variability in the estimate is reduced, as is the noise floor of the estimator, as shown by the lower horizontal dotted line. Figure 7C shows the cross-correlation function for these data. Here, too, there is structure in the estimate, but the degree of variability lowers the power of the analysis.

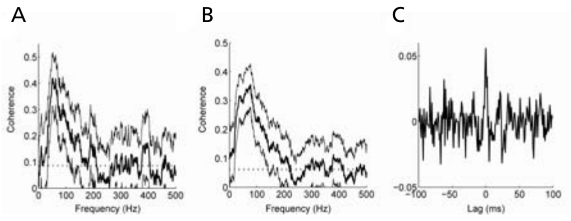


Figure 7. Spike coherence and cross-correlation function for spiking activity of two simultaneously recorded neurons in macaque LIP during delay period before a saccade-and-reach to the preferred direction for both cells. ***A***, Multitaper coherence estimate duration of 500 ms; bandwidth 16 Hz averaged across 9 trials. Thin lines show 95% confidence empirical error bars using leave-one-out procedure. Dotted horizontal line shows expected coherence under the null hypothesis that coherence is zero. ***B***, Multitaper coherence estimate duration of 500 ms; bandwidth 30 Hz averaged across 9 trials. Conventions as for *A*. ***C***, Cross-correlation function estimate from which shift predictor has been subtracted.

Regression using spectral feature vectors

Detection of period signals is an important problem that occurs frequently in the analysis of neural data. Such signals can arise as a result of periodic stimulation and can manifest as 50/60 Hz line noise. We pursue the effects of periodic stimulation in the multivariate case in the next chapter Multivariate Neural Data Sets: Image Time Series, Allen Brain Atlas. As discussed therein, certain experiments that have no innate periodicity may also be cast into a form that makes them amenable to analysis as periodic stimuli. We now discuss how such components may be detected and modeled in the univariate time series by performing a regression on the spectral coefficients.

Periodic components are visible in preliminary estimates as sharp peaks in the spectrum, which, for multitaper estimation with Slepians, appear with flat tops owing to narrow-band bias. Consider one such sinusoid embedded in colored noise:

$$x(t) = A\cos(2\pi f t + \varphi) + \eta(t)$$

It is customary to apply a least-squares procedure to obtain A and ϕ, by minimizing the sum of squares $\sum_t |x(t) - A\cos(2\pi f_0 t + \phi)|^2$. However, this is a nonlinear procedure that must be performed numerically; moreover, it effectively assumes a white-noise spectrum. Thomson's F-test offers an attractive alternative within the multitaper framework by reducing the line-fitting procedure to a simple linear regression.

Starting with a data sequence containing *N* samples, multiplying both sides of the equation by a Slepian taper $w_k(t)$ with bandwidth parameter 2*W*, and Fourier transforming, one obtains this result:

$$\tilde{x}_k(f) = \mu U_k(f - f_0) + \mu^* U_k(f - f_0) + N_k(f)$$

Here $\mu = A\exp(i\phi)$ and $U_k(f)$ and $N_k(f)$ are the Fourier transforms of $w_k(f)$ and $\eta(t)$, respectively. If f_o is larger than the bandwidth *W*, then $f - f_o$ and $f + f_o$ are separated by more than 2*W*, and $U_k(f - f_o)$ and $U_k(f + f_o)$ have minimal overlap. In that case, one can set $f = f_o$ and neglect the $U_k(f + f_o)$ term to obtain the following linear regression equation at $f = f_o$:

$$\tilde{x}_k(f) = \mu U_k(0) + N_k(f_0)$$

The solution is given by

$$\hat{\mu}(f_0) = \frac{\sum_{k=1}^{K} U_k(0)\,\tilde{x}_k(f_0)}{\sum_{k=1}^{K} |U_k(0)|^2}$$

The goodness of fit of this model may be tested using an F-ratio statistic with (2, 2K − 2) degrees of freedom, which is usually plotted as a function of frequency to determine the position of the significant sinusoidal peaks in the spectrum,

$$F(f) = \frac{(K-1)\,|\hat{\mu}(f)|^2 \sum_{k=1}^{K} |U_k(0)|^2}{\sum_{k=1}^{K} |\tilde{x}_k(f) - \hat{\mu}(f)U_k(0)|^2}$$

Once an F-ratio has been calculated, peaks deemed significant by the F-test, and which exceed the significance level 1–1/N, may be removed from the original process in order to obtain a reshaped estimate of the smooth part of the spectrum:

$$S_{reshaped}(f) = \frac{1}{K}\sum_{k=1}^{K} \left|\tilde{x}_k(f) - \sum_i \mu_i U_k(f - f_i)\right|^2$$

This reshaped estimate may be augmented with the previously determined line components, to obtain a so-called mixed spectral estimate. This provides one of the more powerful applications of the multitaper methodology, since, in general, estimating such mixed spectra is difficult.

References

Brillinger DR (1978) Comparative aspects of the study of ordinary time series and of point processes. In: Developments in statistics vol. 11. Orlando, FL: Academic Press.

Buzsaki G (2006) Rhythms of the Brain. New York: Oxford UP.

Jarvis MR, Mitra PP (2001) Sampling properties of the spectrum and coherency of sequences of action potentials. Neural Comput 13:717-749.

Mitra PP, Bokil H (2007) Observed Brain Dynamics. New York: Oxford UP.

Percival DB, Walden AT (1993) Spectral analysis for physical applications. Cambridge, UK: Cambridge UP.

Pesaran B, Pezaris JS, Sahani M, Mitra PP, Andersen RA (2002) Temporal structure in neuronal activity during working memory in macaque parietal cortex. Nat Neurosci 5:805-811.

Slepian D, Pollack HO (1961) Prolate spheroidal wavefunctions: Fourier analysis and uncertainty I. Bell System Tech Journal 40:43-63.

Steriade M (2001) The intact and sliced brain. Cambridge, MA: MIT Press.

Thomson DJ (1982) Spectrum estimation and harmonic analysis. Proc IEEE 70:1055-1996.

Thomson DJ, Chave AD (1991) Jackknifed error estimates for spectra, coherences, and transfer functions. In: Advances in spectrum analysis and array processing, pp 58-113. Englewood Cliffs, NJ: Prentice Hall.

Wasserman L (2007) All of nonparametric statistics. New York: Springer-Verlag.

NOTES

Multivariate Neural Data Sets: Image Time Series, Allen Brain Atlas

Hemant Bokil, PhD

Cold Spring Harbor Laboratory
Cold Spring Harbor, New York

Introduction

Recent technological advances have led to a tremendous increase in the dimensionality of commonly acquired neural signals. For example, microelectrode measurements, electroencephalography (EEG), and magnetoencephalography (MEG) can record 100-300 channels simultaneously; images acquired in optical imaging comprise several thousand pixels; and volumetric imaging techniques such as functional magnetic resonance imaging (fMRI) provide signal measurements at tens of thousands of voxels. Although the availability of such data holds the promise of allowing researchers to look beyond the activity of individual neurons to study local and global neuronal networks, it also poses significant challenges for their analysis and interpretation.

The previous chapter in this short course, Spectral Analysis for Neural Signals, illustrated how nonparametric spectral estimation methods can be used to obtain insight into the structure of correlations in univariate and bivariate time series data. In this chapter, we extend that discussion to encompass multivariate data. First, we begin by discussing the singular value decomposition (SVD) and show how this technique may be used in both time and frequency domains in order to reduce the dimensionality of a multivariate data set and for denoising. Second, we discuss how the SVD may be combined with the Thomson F-test for lines (discussed in the previous chapter) to extract stimulus features from optical imaging experiments with repeated stimuli. Third, the oral presentation illustrates the use of clustering methods and SVD for analyzing data obtained from genomewide expression studies of the adult mouse brain. In keeping with the nature of this chapter, our discussion will be brief. More details may be found in a recent book on the analysis of neural time series data (Mitra and Bokil, 2008) and in a number of specialized textbooks on linear algebra (Strang, 1998), multivariate statistics (Anderson, 1984), and time series analysis (Percival and Walden, 1993).

Matrix Factorizations: Singular Value Decomposition

Matrix factorizations are a set of methods in linear algebra for expressing a given matrix as a product of other, simpler matrices. Examples include the LU decomposition, in which a matrix is a written as a product of a lower triangular and upper triangular part; the QR decomposition, in which a matrix is written as a product of a unitary matrix Q and an upper triangular matrix R; and the eigenvalue decomposition, in which a square matrix with all eigenvalues distinct is expressed as a product of the matrices comprising its eigenvalues and eigenvectors.

While each decomposition technique mentioned above has use in specific problems (for example, the LU decomposition arises in solving systems of linear equations), the most useful technique from a data analysis perspective is arguably the SVD. The SVD of a matrix M is given by the following equation:

$$M = U\Lambda V^{\dagger}, \tag{1}$$

where U and V are unitary matrices viz. $UU^{\dagger} = VV^{\dagger} = I$ and Λ is a diagonal matrix of real numbers. When M is a $p \times q$ matrix (with $p \geqslant q$), U is a $p \times q$ matrix, and $\Lambda = diag\,(\lambda_1, \lambda_2, \ldots, \lambda_q)$ and V are $q \times q$ matrices. Letting u_i and v_i denote the i^{th} columns of U and V, respectively, it can be shown that $Mu_i = \lambda_i v_i$ and $Mv_i = \lambda_i u_i$. λ_i are known as singular values, and u_i and v_i are known as the left and right singular vectors, respectively.

The following properties of the SVD are worthy of note:

(a) When M is hermitian (i.e., $M = M^{\dagger}$), the eigenvalues of M are its singular values. In general, this is not true: Singular values are always real and positive (for matrices with real entries, the singular vectors are also real), whereas eigenvalues are, in general, complex. Furthermore, eigenvalues can be defined only for square matrices, whereas the SVD exists for any matrix.

(b) The SVD is closely related to principal components analysis. Interpreting the rows of M as p samples of q variables, and assuming that the mean of each column is zero makes $M^{\dagger}M/p$ the sample covariance matrix. Using Equation 1, $M^{\dagger}M$ is given by the formula below:

$$M^{\dagger}M = V\Lambda^2 V^{-1} \tag{2}$$

and we have,

$$M^{\dagger}MV = \Lambda^2 V \tag{3}$$

Therefore, the singular vectors v_i are the eigenvectors of the sample covariance matrix, i.e., the principal components, and λ_i^2/p are the variances along the principal directions.

(c) For a $p \times q$ matrix whose entries are independent, identically distributed Gaussian random variables with standard deviation σ, the density of singular values is given by the following:

$$\rho(\lambda) = \frac{1}{\pi\sigma\lambda^2}\sqrt{\left(\lambda_+^2 - \lambda^2\right)\left(\lambda^2 - \lambda_-^2\right)} \text{ for } \lambda_- \leq \lambda \leq \lambda_+ \tag{4}$$
$$= 0 \text{ otherwise,}$$

where $\lambda_{\pm} = \sqrt{2}\sigma\sqrt{(p+q)/2 \pm \sqrt{pq}}$

NOTES

NOTES

(Denby and Mallows, 1991; Sengupta and Mitra, 1999). As illustrated below, this formula provides a principled method for separating the signal and noise subspaces. A general formula can also be derived that includes correlations in the noise (Sengupta and Mitra, 1999).

SVD in the time domain

The simplest way to apply techniques such as the SVD to multivariate time-series data is to cast such data into a matrix $M(t,x)$, where t denotes the acquisition time and x denotes the spatial variable being measured. Thus, we write:

$$M(t,x) = \sum_{\alpha} \lambda_\alpha u_\alpha(t) v_\alpha(x) \qquad (5)$$

The left singular vectors are functions of time (and are referred to as *temporal modes*), and the right singular vectors are functions of the spatial variables (and are referred to as *spatial modes*). Although casting the data into this matrix form ignores the spatial relationship between the spatial variables, x, it can still provide useful insight into the structure of the data, as illustrated below.

Figure 1 shows the sorted singular values for a sample fMRI data set. Five hundred and fifty echo planar images were acquired for 110 s at 5 Hz during binocular stimulation by a pair of flickering red LED patterns (Mitra et al., 1997). Each image was masked from the set of original 64 × 64 pixel images, and the resulting image is composed of 1877 pixels and 550 samples. The SVD was performed on this 1877 × 550 matrix.

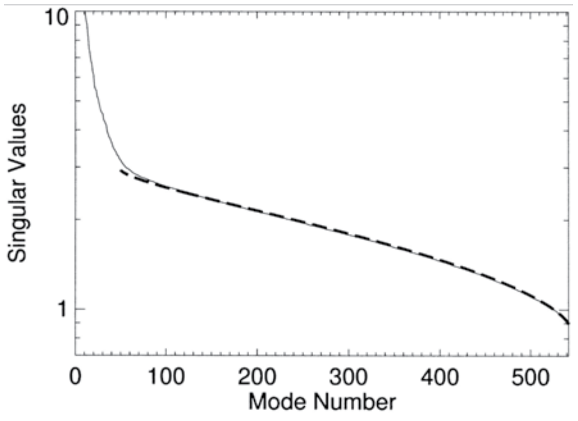


Figure 1. Example of sorted singular values determined by the space-time SVD.The tail of the singular value spectrum (solid line) is fit by the theoretical density of singular values for a pure noise matrix (dashed line). The range of this plot has been truncated to highlight the tail of the spectrum. Reprinted from Mitra and Pesaran, 1999, their Figure 8, with permission of the Biophysical Society.

Note the shape of the singular value plot, which displays an initial rapid drop followed by a slower decay. This separation between a fast and slow decay is quite common, the slowly varying part corresponding principally to noise. In fact, for this data set, the uncorrelated noise formula of Equation 4 provides a good fit for all but the 50 highest singular values, and an approximate signal subspace may be defined by the corresponding singular vectors. But even when an explicit fit is unavailable, the sum in Equation 5 may be truncated to keep only those singular values that are higher than the "knee" of the curve in Figure 1. Finally, truncation may be carried out so as to capture a certain fraction of the total variance (see property (b), above). Each of these methods serves to denoise the data and to reduce its dimensionality.

Once a reduction has been carried out (as per the discussion in the previous paragraph), leading temporal and spatial modes may be studied in order to obtain understanding of the data. Figure 2 shows the average spectrum computed from the 10 leading temporal modes for an fMRI data set collected using rapid, single-shot gradient-echo echo-planar imaging (EPI) at 7T, with a repetition time (TR) of 450 ms. In this experiment, visual stimuli were presented in 16.2-s blocks with 22-s control periods. Two different visual stimuli were used, and these were presented in alternating blocks (e.g., ABABAB). This resulted in two stimulus paradigm frequencies: 0.0262 Hz (corresponding to the AB frequency) and 0.0131 Hz (corresponding to the frequency of A

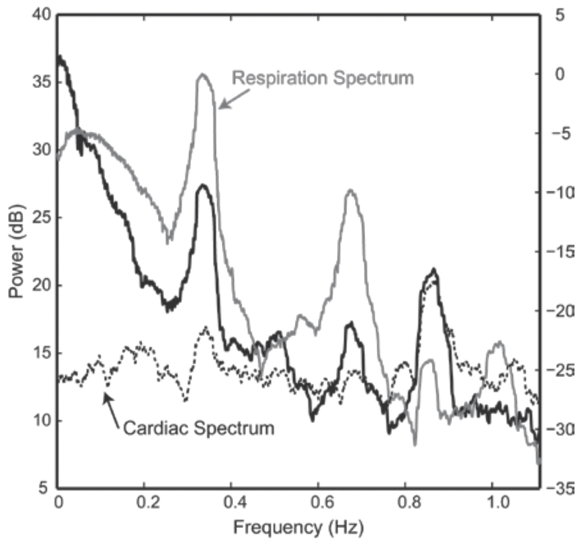


Figure 2. Average spectrum computed from the 10 largest principal components (thick line) from an fMRI data set, plotted with the estimated spectra of the monitored respiratory signal (thin line) and cardiac signal (dashed line). Courtesy of Partha Mitra and Wayne King.

NOTES

and B alone). Respiratory and cardiac signals were collected simultaneously with magnetic resonance (MR) acquisition, and the estimated spectra of those time series are also shown in the figure. By comparing the average principal component spectrum with the respiration and cardiac spectra, one can visually determine that the spectral peaks at ~0.33 Hz and ~0.66 Hz are very likely the result of respiratory fluctuations. Thus, in this case, the leading components of the time domain SVD contain physiological responses that are unrelated to the stimulus response of interest. However, at frequencies near the stimulus frequency (near DC), less of a contribution appears to be made by the physiological signals, and the power in the principal components is likely the result of a stimulus response. Thus, noise affects various frequencies differently, suggesting the need for a frequency-localized analysis, discussed next.

Frequency domain SVD: factorization of the cross-spectral matrix

The previous chapter in this short course discussed the concept of a cross-spectrum. For multiple channels, this generalizes to the cross-spectral matrix $S_{ij}(f)$. This is the matrix

$$S_{ij}(f) = E\,[X_i^*(f)X_j^*(f)], \qquad (6)$$

whose multitaper estimate is given by the following formula:

$$S_{ij}(f) = \frac{1}{K}\sum_{k=1}^{K}\tilde{x}_{ik}^*(f)\tilde{x}_{jk}(f). \qquad (7)$$

Here $\tilde{x}_{ik}(f)$ denotes the Fourier transform of the data with the i^{th} taper. Since this matrix is Hermitian, it has real eigenvalues, and its eigenvalue decomposition can be expressed as follows:

$$S_{ij}(f) = \sum_{\alpha=1}^{N}\sigma_\alpha(f)u_\alpha^*(i;f)u_\alpha(j;f). \qquad (8)$$

$\sigma_\alpha(f)$ are known as eigenspectra, and the $N \times N$ matrix of cross-spectra has thus been reduced to N eigenspectra. These eigenspectra can be further condensed into a global coherence by taking the fractional energy captured by the leading eigenspectrum as a fraction of the total energy at a given frequency:

$$C_G(f) = \frac{\sigma_1(f)}{\sum_{\alpha=1}^{N}\sigma_\alpha(f)} \qquad (9)$$

This is a number between 0 and 1. If $C_G(f)$ is 1, this means that the cross-spectral matrix $S_{ij}(f)$ is rank 1, at that frequency, and that all the processes are multiples of a single process, so there is perfect correlation.

The multitaper approach provides a fast algorithm for estimating this global coherence and the leading eigenspectra without having to first compute the cross-spectral matrix. One performs an SVD directly on the tapered Fourier transforms $\tilde{x}_{ik}(f)$ of the individual time series, $x_i(t)$ as follows:

$$\tilde{x}_{ik}(f) = \sum_{\alpha=1}^{K}\lambda_\alpha(f)u_\alpha^*(i;f)v_\alpha(k;f). \qquad (10)$$

Substituting Equation 10 in Equation 7, using the orthonormality of $v_\alpha(k;f)$, and comparing with Equation 8 gives us $\sigma_\alpha(f) = |\lambda_\alpha(f)|^2$. Note that since $S_{ij}(f)$ is $N \times N$, and N can become large—especially in applications with image time series—this procedure involving an $N \times K$ matrix may take the computation from being intractable to being tractable.

As an example, consider the visual stimulation experiments used in Figure 2. The global coherence in the presence or absence of visual stimulation is shown in Figure 3 (solid and dashed lines, respectively). The leading spatial eigenmodes for low center frequencies are shown in Figure 4. The stimulus response within the visual cortex during visual stimulation can be seen in the two lowest-frequency spatial modes in Figure 4.

Harmonic Analysis for Periodized Data

The previous chapter discussed the problem of detecting periodic signals buried within a noisy background using the Thomson F-test. Here we show how the same method may be used to analyze imaging experiments with repeated, not necessarily periodic, stimuli.

The basic idea is that stimuli presented repeatedly may be rearranged into a periodic pattern that makes them amenable to harmonic analysis: the so-called periodic stacking method (Sornborger et al., 2003a,b). In a typical repeated stimulation experiment, the subject is exposed M times to a set of N stimuli (often in random order). The data can be rearranged to form a periodic sequence such that the stimuli appear in order. For example, a sequence of responses to the stimulus sequence ABCCACBBCABA..., can be rearranged in the form ABCABCABCABC... Since the stimuli now appear periodically, the rearranged time series can be decomposed into a periodic response with period NT (T being the time of presentation of each stimulus) and noise. It can therefore be analyzed using the Thomson F-test. The spectrum of the rearranged data set should exhibit peaks at harmonic multiples of the fundamental frequency

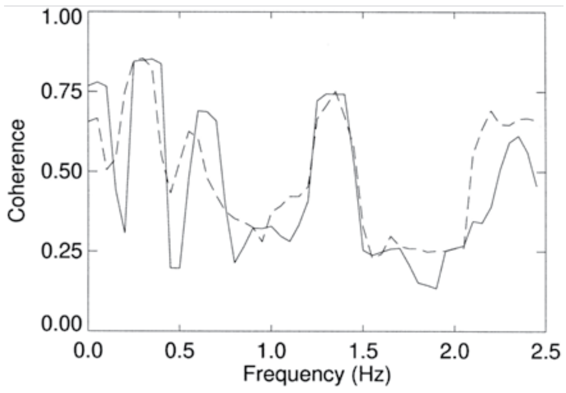


Figure 3. Global coherence spectra computed from the space-frequency SVD of the fMRI data set from the previous figure. The solid line indicates the spectrum for the data recorded in the presence of visual stimulus, the dashed line in the absence of visual stimulus. Reprinted from Mitra and Pesaran, 1999, their Figure 10, with permission of the Biophysical Society.

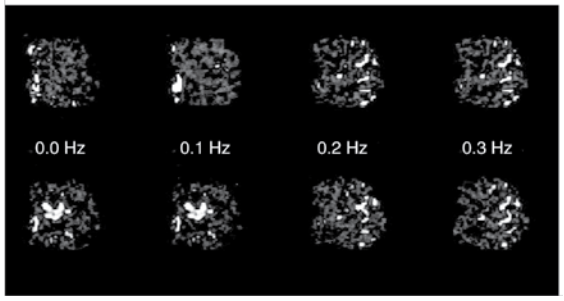


Figure 4. Amplitudes of the leading spatial eigenmodes at center frequencies from 0 to 0.3 Hz in 0.1-Hz increments, obtained from a space-frequency SVD. The components in the top row were obtained from data recorded in the absence of visual stimulus, and those in the bottom row in the presence of visual stimulus. Adapted from Mitra and Pesaran, 1999, their Fig. 26, with permission of the Biophysical Society.

$f_0 = 1/NT$. The periodic component is itself composed of a generalized response to any stimulus (called the nonspecific response) and responses that are specific to individual stimuli. The nonspecific response is rarely of interest and can be shown to correspond to the peaks in the spectrum at integer multiples of Nf_0. The peaks at the remaining harmonics constitute the specific response. Because the data are noisy and high-dimensional, it is of course preferable to apply this procedure to the temporal modes obtained from an SVD rather than to the time series of the individual pixel responses.

Figure 5 shows the spectra of the temporal modes for a sample optical imaging data set. The data set was taken from cat visual cortex, using a moving grating stimulus. Optical images measuring the cortical

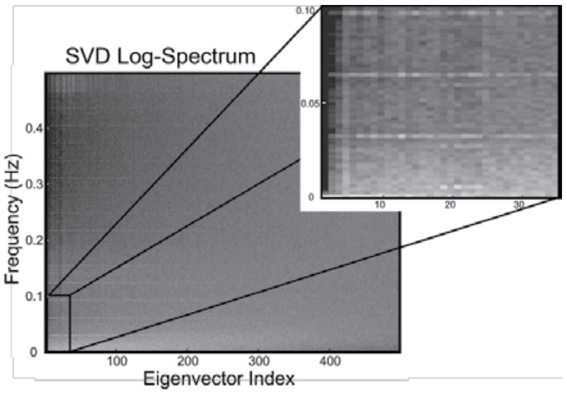


Figure 5. The log-spectrum of leading temporal modes, calculated for the optical imaging data set discussed in the text. The eigenvector index of the mode is displayed on the ***x***-axis and the frequency in hertz on the ***y***-axis. Lighter shades indicate more spectral power and darker shades less spectral power. Also shown is an enlargement of the low-frequency region. Reproduced with permission from Mitra and Bokil, 2008.

reflectance of 605 nm light were acquired at 15 Hz for a duration of $T = 34$ s and were subsequently filtered and subsampled at 1.1 Hz. Twenty measurements were taken for each of the six orientations of the grating. Note that, across many modes (up to an index of 300 or so), a sequence of harmonics has a fundamental frequency of around 0.03 Hz. Harmonics are also seen at multiples of a lower fundamental frequency of approximately 0.005 Hz in the range of mode indices from about 20 to 30 (see blow-up of the low frequency region). Since $N = 6$ and $T = 34$ s, the fundamental frequency $f_0 = 1/204 \sim 0.005$ Hz. Thus, the peak at 0.005 Hz reflects the periodicity of the N stimulus stack. Since $0.03 = 6 \times 0.005$, the peaks at multiples of 0.03 Hz correspond to the nonspecific response.

The amplitudes and the F-values of the sinusoidal components at nf_0 for $n \neq N, 2N, \ldots$ etc., can be determined and the data reconstructed from the statistically significant sinusoids. Thus, if the F-test for the singular vector $u_i(t)$ is significant at frequencies f_{ij}, the reconstructed data are given as follows:

$$I_R(t,x) = \sum_{\alpha} \lambda_\alpha v_\alpha(x) \sum_j a_{\alpha j} \cos\left(2\pi f_{\alpha j} t + \phi_{\alpha j}\right), \quad (11)$$

where a_{α_j} are the amplitudes and ϕ_{α_j} are the phases of the significant sinusoids. Figure 6 shows the results of this analysis on the example data set. To better visualize the response, the figure shows the two leading spatial modes and reconstructed temporal modes retaining the specific response. These modes contain 90% of the variance in the signal. Note the dappled pattern in the spatial modes that result from orientation columns in cat visual cortex.

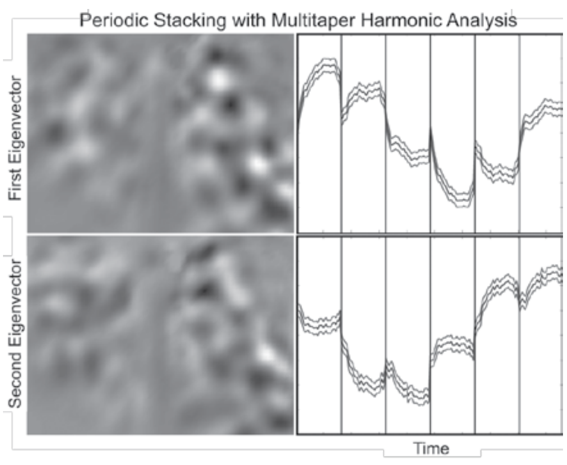


Figure 6. Two leading modes in the SVD of the optical imaging data set reconstructed from the statistically significant periodic responses. These modes contain 90% of the variance of the extracted signal. The left panels show the spatial modes. These two modes represent basis functions that make up the orientation response in cat primary visual cortex. The right panels show the corresponding time courses (black) with one-sigma global confidence bands (gray) for one stack of stimuli. The vertical lines (34 s apart) indicate the separation between responses to different orientation stimuli within a stack. Adapted with permission from Sornborger et al., 2003b.

Conclusion

We have shown that the SVD provides a powerful method for reducing dimensionality and denoising multivariate data sets, including fMRI and optical imaging data. We have also shown how the SVD in combination with the Thomson F-test may be used to extract stimulus-related responses in optical imaging experiments with repeated stimuli.

While the SVD is a powerful method, a caution is nonetheless in order regarding its use. In particular, it should be emphasized that individual eigenmodes computed from the SVD (as well as modes obtained using ICA or other matrix decomposition algorithms) do not necessarily have biological meaning. When modes can be identified with particular physiological sources, this sort of procedure is useful, especially in exploratory data analysis. Nevertheless, these are mathematical algorithms, and there is no necessity that such segregation will occur. The SVD, for example, imposes the constraint that the individual modes must be orthogonal. However, there is no reason to assume that neurobiologically relevant modes are orthogonal to one another. Similarly, it is entirely possible that the "noise tail" of the SVD actually contains physiologically interesting signals that are masked because they have small amplitudes. For these reasons, matrix decomposition algorithms have limited utility as methods for separating different noise and signal sources. In general, rather than consider individual components, it is better to consider entire subspaces consisting of groups of modes, which have a better chance of segregating noise or signal sources of interest.

References

Anderson TW (1984) An introduction to multivariate statistical analysis, 2nd edition. New York: Wiley.

Denby L, Mallows CL (1991) Singular values of large matrices subject to Gaussian perturbations. In: Computing Sciences and Statistics: Proceedings of the 23rd Symposium on the Interface, pp 54-57: Interface Foundation of North America.

Mitra PP, Bokil HS (2008) Observed Brain Dynamics. New York: Oxford UP.

Mitra PP, Pesaran B (1999) Analysis of dynamic brain imaging data. Biophys J 76:691-708.

Mitra PP, Ogawa S, Hu XP, Ugurbil K (1997) The nature of spatiotemporal changes in cerebral hemodynamics as manifested in functional magnetic resonance imaging. Magn Reson Med 37:511-518.

Percival DB, Walden AT (1993) Spectral analysis for physical applications: multitaper and conventional univariate techniques. Cambridge; New York, NY: Cambridge UP.

Sengupta A, Mitra P (1999) Distributions of singular values for some random matrices. Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics 60:3389-3392.

Sornborger A, Sailstad C, Kaplan E, Sirovich L (2003a) Spatiotemporal analysis of optical imaging data. Neuroimage 18:610-621.

Sornborger A, Sirovich L, Morley G (2003b) Extraction of periodic multivariate signals: Mapping of voltage-dependent dye fluorescence in the mouse heart. IEEE Trans Med Imaging 22:1537-1549.

Strang G (1998) Introduction to linear algebra. Wellesley, MA: Wellesley-Cambridge Press.

Application of Spectral Methods to Representative Data Sets in Electrophysiology and Functional Neuroimaging

David Kleinfeld, PhD

Department of Physics and Graduate Program in Neurosciences
University of California
La Jolla, California

Background

Experimental neuroscience involves the use of many different tools from physics, genetics, and other fields. Proper data analysis is an integral part of this set of tools. Used correctly, analysis will help to define the magnitude and significance of a given neural effect; used creatively, analysis can help reveal new phenomena.

In this chapter, we consider five example cases that introduce the utility and implementation of spectral methods:

(1) Deduction of variation in the power of a high-frequency cortical oscillation from a human electrocorticogram (ECoG). This will illustrate frequency-domain concepts such as the spectrogram.
(2) Deduction of synaptic connectivity between neurons in the leech swim network. This will emphasize notions of spectral coherence and its associated confidence level.
(3) The discovery of neurons in rat vibrissa motor cortex that report the pitch of vibrissa movement. This will illustrate the notion of the spectral power density as the sum of δ-functions corresponding to pure tones and a slowly varying, or pink, spectrum.
(4) The denoising of imaging data in the study of calcium waves. This will introduce the concept of singular value decomposition (SVD) in the time domain and illustrate the notion of space-time correlation in multisite measurements.
(5) The delineation of wave phenomena in turtle cortex. This will illustrate the concept of SVD in the frequency domain and further illustrate the notion of space-time coherence.

Throughout this discussion the emphasis will be on explaining the analysis and not on the scientific questions per se.

Advantages of Working in the Frequency Domain

Why work in the frequency domain? One part of the answer is to delineate the number of degrees of freedom required to calculate confidence intervals. The following factors are relevant:

- Determining the number of degrees of freedom is complicated in the time domain, where all but white noise processes lead to correlation between neighboring data points.

- In contrast, counting the number of degrees of freedom is straightforward when neighboring data points are uncorrelated. This occurs in the frequency domain when the amplitude of spectral power in the data varies only slowly on the scale of the bandwidth, so that neighboring points in frequency are uncorrelated.

A second part of the answer is that some phenomena have a simpler representation in the frequency domain rather than the time domain.

This chapter builds on the discussion of the time-bandwidth product and multitaper analysis (Thomson, 1982) in Spectral Analysis for Neural Signals, presented earlier in this Short Course by Bijan Pesaran. First, we recall the time-frequency uncertainty:

$$T\Delta f = 2p$$

where T is the total length of the time series of the data; $2p$ is the number of degrees of freedom and defines the time-frequency product, with $p \geq 1$; and Δf is the resultant full bandwidth. The power is concentrated in the frequency interval Δf, optimally so for the use of family of Slepian tapers employed to estimate spectra (Thomson, 1982). The *maximum* number of tapers, denoted K, that supports this concentration, and which is employed throughout our presentation, is as follows:

$$K = 2p - 1.$$

Rosetta Stone

The variables used in the Pesaran chapter herein and past reviews (Mitra and Pesaran, 1998; Mitra et al., 1999) expressed in discrete, normalized variables by writing (1) $T = N\Delta t$, where N is the number of data points in the time series and $\Delta t = 1/(2f_{Nyquist})$ is the sample time; and (2) $2W = \Delta t\, \Delta f$, where W is the half-bandwidth. Thus, $NW = p$. In normalized variables, time spans the interval $[1, N]$ rather than $[\Delta t, T]$; frequency spans the interval $[-½, ½]$ rather than $[-f_{Nyquist}, f_{Nyquist}]$; and the integral

$\frac{1}{\sqrt{T}}\int_0^T dt$ is replaced by $\frac{1}{\sqrt{N}}\sum_0^N$ with $\Delta t = 1$. Tools for the numerical calculations used in the examples below are part of the Matlab-based Chronux package (www.chronux.org). The primary textbooks include those by Percival and Walden, 1993, and Mitra and Bokil, 2008.

Case one

We analyze a trace of human ECoG data, defined as $V(t)$, that was obtained in a study on ultra-slow brain activity (Nir et al., 2008) (Fig. 1A). The mean value is removed to form the following:

$$\delta V(t) = V(t) - \frac{1}{T}\int_0^T dt\, V(t).$$

NOTES

Our goal is to understand the spectral content of this signal—with confidence limits! The Fourier transform of this signal, with respect to the *k*th taper, is as follows:

$$\delta\tilde{V}^{(k)}(f) = \frac{1}{\sqrt{T}}\int_0^T dt\; e^{-i2\pi ft}\, w^{(k)}(t)\;\delta V(t)$$

where $w^{(k)}(t)$ is the *k*th Slepian taper, whose length is also T. We then compute the spectral power density, whose units are amplitude^2/Hz, in terms of an average over tapers:

$$S(f) \equiv \frac{1}{K}\sum_{k=1}^{K}\left|\delta\tilde{V}^{(k)}(f)\right|^2$$

where $\left|\delta\tilde{V}^{(k)}(f)\right|^2 = \delta\tilde{V}^{(k)}(f)\left[\delta\tilde{V}^{(k)}(f)\right]^*$; we further average the results over all trials, if appropriate. The above normalization satisfies Parseval's theorem, i.e.,

$$\int_0^{f_{Nyquist}} df\; S(f) = \frac{1}{T}\int_0^T dt\;\left[\delta V(t)\right]^2.$$

The spectrum in this example is featureless, having a hint of extra power between 50 Hz and 100 Hz (Fig. 1*B*). One possibility is that the spectrum is better defined on a short time scale, but drifts (Fig. 1*B*, insert). In this case, it is useful to compute the running spectrum, or spectrogram, denoted *S(f; t)*, which is a function of both frequency and time. Here we choose a narrow interval of time, compute the spectrum over that interval, and then step forward in time and recalculate the spectrum. For the example data, this process reveals an underlying modulation in the power between 40 Hz and 90 Hz (Fig. 1*C*): the so-called γ-band.

How do we characterize the γ-band's variations in power? We treat the logarithm of the power in a band as a new signal found by integrating the spectrogram over the frequency:

$$V_\gamma(t) \equiv \frac{1}{f_2 - f_1}\int_{f_1}^{f_2} df\; ln\{S(f;\,t)\}$$

This gives us a new time series (Fig. 1*D*). We take the logarithm because the spectrum is χ^2– as opposed to Gaussian-distributed; this transform stabilizes the estimate of the variance. The spectral components of the new time series are called the "second spectrum," denoted $S_\gamma(f)$ for this example:

$$S_\gamma^{(n)}(f) \equiv \frac{1}{K-1}\sum_{\substack{k=1\\k\neq n}}^{K}\left|\tilde{V}_\gamma^{(k)}(f)\right|^2 \quad \forall\; n$$

The above formula shows a number of spectral features (Fig. 1*E*) and raises two general issues.

The first issue is the calculation of confidence intervals. For variables with a Gaussian dependence on their individual spectral amplitudes, the confidence limits may be estimated in various asymptotic limits. However, the confidence intervals may also be estimated directly by a jackknife (Thomson and Chave, 1991), where we compute the standard error in terms of "delete-one" means. In this procedure, we calculate *K* different mean spectra, in which one term is left out:

$$S_\gamma^{(n)}(f) \equiv \frac{1}{K-1}\sum_{\substack{k=1\\k\neq n}}^{K}\left|\tilde{V}_\gamma^{(k)}(f)\right|^2 \quad \forall\; n$$

Estimating the standard error of $S_\gamma(f)$ requires an extra step since spectral amplitudes are defined on the interval [0, ∞) while Gaussian variables exist on (–∞, ∞). The delete-one estimates, $|C_i^{(n)}(f)|$, were replaced with the following transformed values:

$$g\{S(f)\} = ln[S(f)]$$

or

$$S(f) = e^{g\{S(f)\}}.$$

The mean of the transformed variable is as follows:

$$\mu_\gamma(f) = \frac{1}{K}\sum_{n=1}^{K} g\left\{S_\gamma^{(n)}(f)\right\}$$

and standard error of the transformed variable is as follows:

$$\sigma_\gamma(f) = \sqrt{\frac{K-1}{K}\sum_{n=1}^{K}\left|g\left\{S_\gamma^{(n)}(f)\right\} - \mu_\gamma(f)\right|^2}.$$

The 95% confidence interval for the spectral power is thus $\left[e^{\mu_{i,Mag} - 2\sigma_{i,Mag}},\; e^{\mu_{i,Mag} + 2\sigma_{i,Mag}}\right]$. The confidence

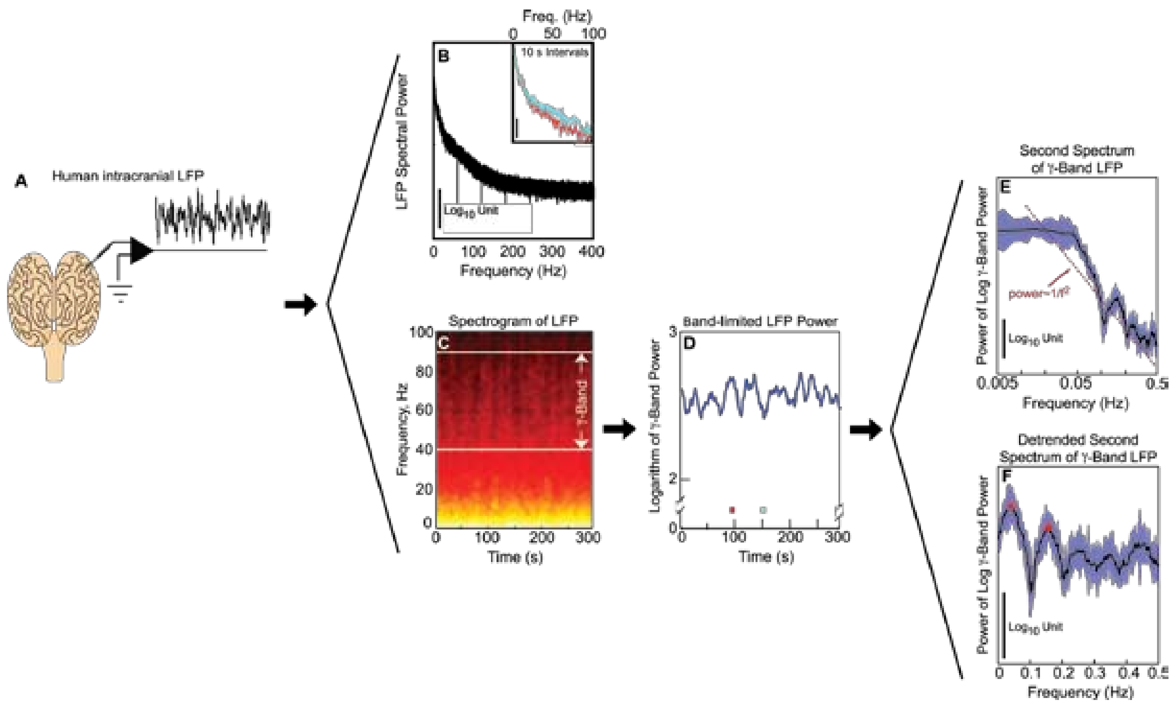


Figure 1. Analysis of the spectral properties of human local field potential (LFP) data (Drew et al., 2008, their Fig. 1, reprinted with permission from ***Nature Neuroscience***). ***A***, The LFP was obtained during stage 2 sleep; f_{Nyquist} = 500 Hz. ***B***, Spectrum (T = 300 s; K = 29) of the LFP in panel *A*; insert shows spectra (T = 10 s; K = 9) for the intervals demarcated in panel *D*. ***C***, Spectrogram (window T = 10 s, overlap = 1 s; K = 13) of the LFP with full bandwidth ΔF = 1.4 Hz; color scale maps the logarithm of power from black/red (low) to white/yellow (high). Note the systematic variation in power in the γ band. ***D***, Time series of the logarithm of the power in the γ band of the LFP, integrated from f_1 = 40 Hz to f_2 = 90 Hz; f_{Nyquist} = 0.5 Hz. ***E***, Second spectrum (T = 300 s; K = 13) using the time series in panel *D* with ΔF = 0.05 Hz; blue stripe is the 95% (2σ) confidence interval. ***F***, Second spectrum using the derivative of the time series in panel *D* as a means of removing a $1/f^2$ trend.

bands are symmetric about the mean when spectral power is plotted on a logarithmic scale (Fig. 1*E*).

A second issue is a $1/f^2$ trend in the spectrum, which obscures peaks. We remove this trend by computing the spectrum of $dV_\gamma(t)/dt$ in order to reveal peaks, particularly at f = 0.05 Hz (Fig. 1*F*). The conclusion is that power in the γ-band shows slow periodic variation, which is of interest in light of the conjecture that this variation may drive spectrally similar variations in the blood oxygenation level-dependent (BOLD) functional magnetic resonance (fMR) signal (Nir et al., 2008).

Case two

We now consider the use of optical imaging to determine potential pairwise interactions between neurons (Cacciatore et al., 1999). We focus on imaging data taken from the ventral surface of a leech ganglion and seek to identify cells in the ganglion that receive monosynaptic input from neuron Tr2 in the head (Fig. 2*A*). This cell functions as a toggle for regulating the swim rhythm in these animals. Rather than serially impaling each of the roughly 400 cells in the ganglion and looking for postsynaptic currents induced by driving Tr2, a parallel strategy was adopted by Taylor and colleagues (2003). The cells in the ganglion were stained with a voltage-sensitive dye (Fig. 2*B*), which transforms changes in membrane potential into changes in the intensity of fluorescent light. The emitted light from all cells was then detected with a CCD camera (Fig. 2*B*) from which time series for the change in fluorescence were calculated for each neuron in the field. Presynaptic cell Tr2 was stimulated with a periodic signal, at frequency f_{Drive}, with the assumption that candidate postsynaptic followers of Tr2 would fire with the same periodicity (Fig. 2*C*). The phase of the coherence relative to the drive depends on several factors: the sign of the synapse, propagation delays, and filtering by postsynaptic processes.

The coherence between the response of each cell and the drive (a complex function denoted $C_i(f)$) was calculated over the time period of the stimulus. We denoted the time series of the optical signals as $V_i(t)$

NOTES

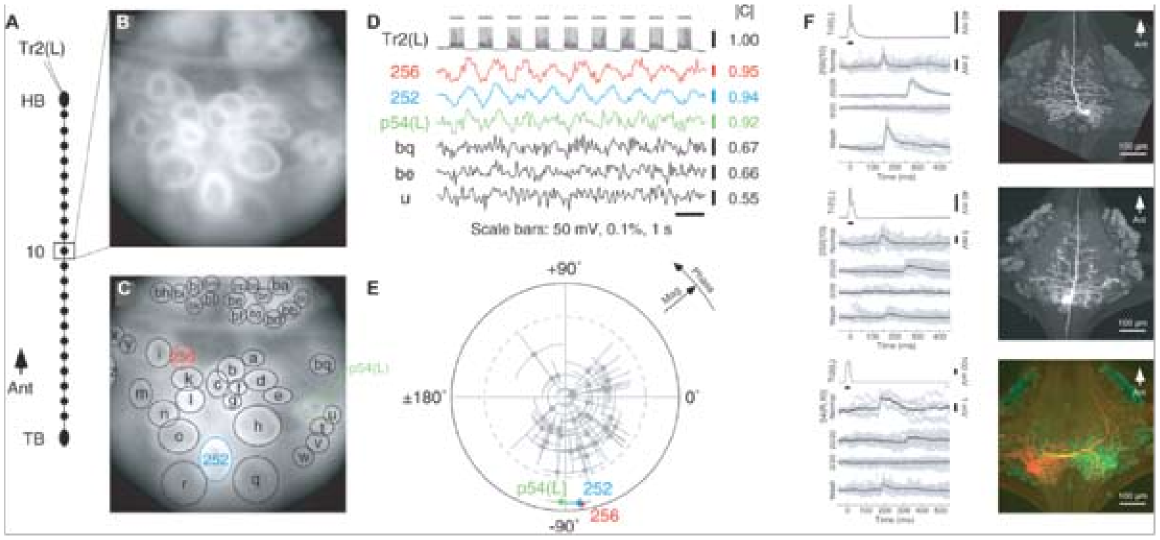


Figure 2. Analysis of voltage-sensitive dye imaging experiments to find followers of Tr2 (Taylor et al., 2003, their Fig. 1 and Fig. 3, reprinted with permission from the *Journal of Neuroscience*). ***A***, Cartoon of the leech nerve cord; input to Tr2 forms the drive ***U(t)***. ***B***, Fluorescence image of ganglion 10 stained with dye. ***C***, Ellipses drawn to encompass individual cells and define regions whose pixel outputs were averaged to form the ***V_i(t). D***, Simultaneous electrical recording of Tr2, i.e., ***U(t),*** and optical recordings from 6 of the cells shown in panel ***C***, i.e., ***V_1(t)*** through ***V_6(t)***, along with $|C_i(f_{Drive})|$ (T = 9 s; K = 11) ***E***, Polar plot of $C_i(f_{Drive})$ between each optical recording and the cell Tr2 electrical recording for all 43 cells in panel ***C***. The dashed line indicates that $\alpha = 0.001$ is the threshold for significance; error bars = one standard error. ***F***, Results of electrophysiological tests of monosynaptic drive for cells 356, 252, and p54, along with confocal images of fills of these cells. Spike-triggered averages, each from a different condition, are shown with error bands. The spike in Tr2 that triggered each sweep is shown only for the first condition; scale bar indicates current injection. The second trace (Normal) is the recording from the postsynaptic target in physiological saline. The third trace (20/20) is the recording in 20 mM Ca^{2+} / 20 mM Mg^{2+} saline to block polysynaptic transmission. The fourth trace (0/20) is in 0 mM Ca^{2+} / 20 mM Mg^{2+} saline to block all transmission. The bottom trace (Wash) is again in normal saline.

and the reference drive signal as $U(t)$. The spectral coherence was defined as follows:

$$C_i(f) = \frac{\frac{1}{K}\sum_{k=1}^{K} \tilde{V}_i^{(k)}(f)\left[\tilde{U}^{(k)}(f)\right]^*}{\sqrt{\left(\frac{1}{K}\sum_{k=1}^{K}\left|\tilde{V}_i^{(k)}(f)\right|^2\right)\left(\frac{1}{K}\sum_{k=1}^{K}\left|\tilde{U}^{(k)}(f)\right|^2\right)}}.$$

To calculate the standard errors for the coherence estimates, we again used the jackknife (Thomson and Chave, 1991) and computed delete-one averages of coherence, denoted $C_n^{(k)}(f)$, where n is the index of the deleted taper:

$$C_i^{(n)}(f) = \frac{\frac{1}{K-1}\sum_{\substack{k=1\\k\neq n}}^{K} \tilde{V}_i^{(k)}(f)\left[\tilde{U}^{(k)}(f)\right]^*}{\sqrt{\left(\frac{1}{K-1}\sum_{\substack{k=1\\k\neq n}}^{K}\left|\tilde{V}_i^{(k)}(f)\right|^2\right)\left(\frac{1}{K-1}\sum_{\substack{k=1\\k\neq n}}^{K}\left|\tilde{U}^{(k)}(f)\right|^2\right)}}.$$

Estimating the standard error of the magnitude of $C_i(f)$ requires an extra step, similar to the case for the spectral power, since $|C_i(f)|$ is defined on the interval [0, 1] while Gaussian variables exist on $(-\infty, \infty)$. The delete-one estimates, $|C_i^{(n)}(f)|$, were replaced with the following transformed values:

$$g\{|C_i|\} = ln\left(\frac{|C_i|^2}{1-|C_i|^2}\right)$$

or

$$|C_i| = \frac{1}{\sqrt{1+e^{-g\{|C_i|\}}}}.$$

The mean of the transformed variable is as follows:

$$\mu_{i;\,Mag}(f) = \frac{1}{K}\sum_{n=1}^{K} g\{C_i^{(n)}(f)\}$$

and standard error of the transformed variable is as follows:

$$\sigma_{i;\,\mathrm{Mag}}(f) = \sqrt{\frac{K-1}{K}\sum_{n=1}^{K}\left|g\left\{C_i^{(n)}(f)\right\} - \mu_{i;\,\mathrm{Mag}}(f)\right|^2}\ .$$

The 95% confidence interval for the coherence is $\left[\sqrt[-1]{1+e^{-(\mu_{i;\mathrm{Mag}}-2\sigma_{i;\mathrm{Mag}})}},\ \sqrt[-1]{1+e^{-(\mu_{i;\mathrm{Mag}}+2\sigma_{i;\mathrm{Mag}})}}\right]$. For completeness, an alternate transformation for computing the variance is $g\{|C_i|\} = \tanh^{-1}\{|C_i|\}$.

We now consider an estimate of the standard deviation of the phase of $C(f)$. Conceptually, the idea is to compute the variation in the relative directions of the delete-one unit vectors $C_i(f)/|C_i(f)|$. The standard error is computed as follows:

$$\sigma_{i;\,\mathrm{Phase}}(f) = \sqrt{2\,\frac{K-1}{K}\left(K - \left|\sum_{n=1}^{K}\frac{C_i^{(n)}(f)}{\left|C_i^{(n)}(f)\right|}\right|\right)}\ .$$

Our interest lies in the values of $C_i(f)$ for $f = f_{Drive}$ and the confidence limits for these values. We chose the bandwidth so that the estimate of $|C_i(f_{Drive})|$ is kept separate from that of the harmonic $|C_i(2f_{Drive})|$; the choice $\Delta f = 0.4\, f_{Drive}$ works well. We thus graph the magnitude and phase of $C_i(f_{Drive})$ for all neurons, along with the confidence interval, on a polar plot (Fig. 2*E*).

Finally, we consider whether the coherence of a given cell at f_{Drive} is significantly greater than zero, that is, larger than one would expect to occur by chance from a signal with no coherence, as a means of selecting candidate postsynaptic targets of Tr2. We compared the estimate for each value of $|C_i(f_{Drive})|$ to the null distribution for the magnitude of the coherence, which exceeds

$$\left|C_i\left(f_{\mathrm{Drive}}\right)\right| = \sqrt{1-\alpha^{1/(K-1)}}$$

only in α of the trials (Hannan, 1970; Jarvis and Mitra, 2001). We used $\alpha = 0.001$ in our experiments to avoid false-positives. We also calculated the multiple comparisons of α level for each trial, given by $\alpha_{multi} = 1 - (1 - \alpha)^N$, where N is the number of cells in the functional image, and verified that it did not exceed $\alpha_{multi} = 0.05$ on any trial.

The result of the above procedure was the discovery of three postsynaptic targets of cell Tr2, two of which were functionally unidentified neurons (Taylor et al., 2003) (Fig. 2*F*).

Case three

Rats can palpate objects via highly regular rhythmic whisking (Berg and Kleinfeld, 2003). Motivated by ideas from control theory, we conjectured that primary motor (M1) cortex transforms sensory input to serve as a feedback signal for the smooth motion of the vibrissae. We posed the question of whether a punctate periodic input (such as occurs when the animal rhythmically palpates an object) is transformed into a pure sinusoidal signal over the 5-20 Hz range of normal whisking frequencies.

To test this theory, we had awake rats hold their vibrissae still as periodic air puffs were delivered. Records from primary sensory cortex (S1) showed a known punctate response, while those from M1 were smooth (Figs. 3*A*,*E*). The stimulus-driven power spectrum for the S1 unit displayed multiple harmonics (Fig. 3*B*) consistent with a pulsatile response, while that for the M1 unit appeared to have power only at the fundamental frequency of the stimulus repetition rate, denoted f_1 (Fig. 3*F*). We consequently asked whether the motor response could be replaced with a pure sinusoidal process with a frequency of f_1. Thus, the time series for the spike rate must take the following form:

$$V(t) = A_1\cos\left(2\pi f_1 t + \varphi_1\right) + \eta(t)\,,$$

where $\eta(t)$ is the residual noise. We recall that $\cos(x) = ½(e^{ix} + e^{-ix})$ and define

$$B_1 \equiv \frac{A_1}{2}\,e^{i\varphi_1}$$

so that $V(t)$ can be placed in a computationally convenient form, as follows:

$$V(t) = B_1\,e^{i2\pi f_1 t} + B_1^*\,e^{-i2\pi f_1 t} + \eta(t).$$

As a means of estimating the complex amplitude B_1, we first make multiple estimates of the spectra in the vicinity of frequency f_1 so that we can regress over a number of tapers to determine B_1. The Fourier transform of $V(t)$ with respect to the kth taper is as follows:

$$\tilde{V}^{(k)}(f) = \frac{B_1}{\sqrt{T}} \int_0^T dt\, e^{-i2\pi f t} w^{(k)}(t) e^{i2\pi f_1 t} + \frac{B_1^*}{\sqrt{T}} \int_0^T dt\, e^{-i2\pi f t} w^{(k)}(t) e^{-i2\pi f_1 t} + \frac{1}{\sqrt{T}} \int_0^T dt\, e^{-i2\pi f t} w^{(k)}(t)\, \eta(t).$$

This compresses to

$$\tilde{V}^{(k)}(f) = B_1\, \tilde{w}^{(k)}(f - f_1) + B_1^*\, \tilde{w}^{(k)}(f + f_1) + \tilde{\eta}^{(k)}(f)\ ,$$

where we defined the spectrum of the kth taper as

$$\tilde{w}^{(k)}(f) = \frac{1}{\sqrt{T}} \int_0^T dt\, e^{-i2\pi f t}\, w^{(k)}(t)$$

and the kth spectral estimate of the residual as

$$\tilde{\eta}^{(k)}(f) = \frac{1}{\sqrt{T}} \int_0^T dt\, e^{-i2\pi f t}\, w^{(k)}(t)\, \eta(t).$$

The estimate of the transform at the frequency of interest, $f = f_1$, becomes

$$\tilde{V}^{(k)}(f_1) = \begin{cases} B_1 \tilde{w}^{(k)}(0) + \tilde{\eta}^{(k)}(f_1) & \text{for } k = 1, 3, 5, \ldots \\ \tilde{\eta}^{(k)}(f_1) & \text{for } k = 2, 4, 6, \ldots \end{cases}$$

where we used $\tilde{w}^{(k)}(2f_1) = 0$ from the condition $2f_1 > \Delta f$, i.e., the spectrum of a taper has no amplitude outside of its bandwidth, and note that $w^{(k)}(t)$ is an odd function for even values of k.

The above relation specifies a regression for B_1 over odd values of k, where the $\tilde{V}^{(k)}(f_1)$ are the dependent variables and the $\tilde{\eta}^{(k)}(f_1)$ are the independent variables. The least-squares estimate of B_1, denoted $\hat{B}_1$, is as follows:

$$\hat{B}_1 = \frac{\sum_{k=1,3,5,\ldots}^{K} \tilde{w}^{(k)}(0)\ \tilde{V}^{(k)}(f_1)}{\sum_{k=1,3,5,\ldots}^{K} \left[\tilde{w}^{(k)}(0)\right]^2}$$

and the associated F-statistic is derived to be the following (Thomson, 1982):

$$F_{2,\,2K-2} = \left|\hat{B}_1\right|^2 \frac{(K-1) \sum_{k=1}^{K} \left|\tilde{w}^{(k)}(0)\right|^2}{\sum_{k=1}^{K} \left| \tilde{V}^{(k)}(f_1) - \hat{V}^{(k)}(f_1)\right|^2}$$

The formula below is the estimated contribution of the periodic signal to the continuous spectrum:

$$\hat{V}^{(k)}(f_1) = \begin{cases} \hat{B}_1 \tilde{w}^{(k)}(0) & \text{for } k = 1, 3, 5, \ldots \\ 0 & \text{for } k = 2, 4, 6, \ldots \end{cases}$$

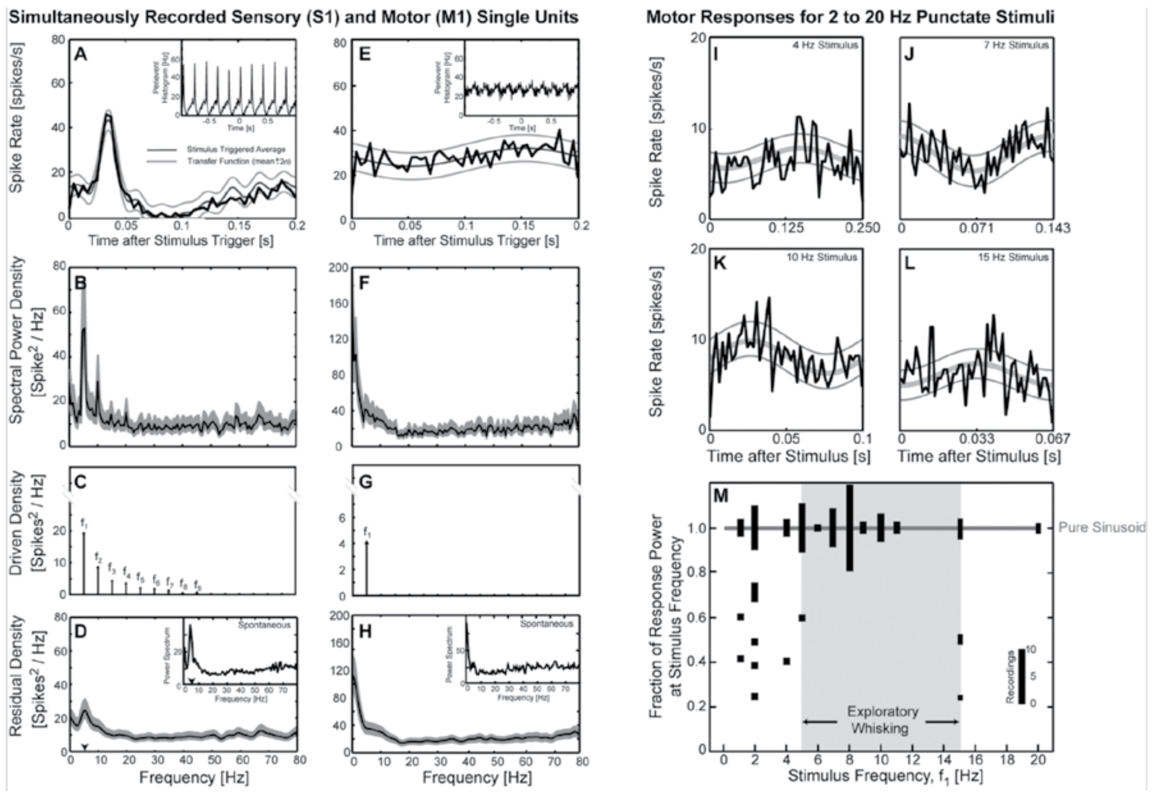


Figure 3. Simulated analysis of single unit data from S1 and M1 cortices as a single vibrissa of an awake but immobile rat (Kleinfeld et al., 2002, their Fig. 3 and Fig. 5, reprinted with permission from Neuron). ***A, E,*** Stimulus-triggered average and transfer function between the stimulus and the instantaneous spike rate of sensory and motor units (thin black line) for vibrissa stimulation at 5 Hz. The transfer function was computed from a spectral decomposition (***T*** = 100 s; ***K*** = 99) of the time series of the response. ***B***, ***F***, Spectral power of the unit spike response; full bandwidth Δ***f*** = 1.0 Hz. ***C***, ***G***, The spectral power for the stimulus-driven part of the response. The height of each arrow corresponds to the magnitude of the complex coefficient for power at the fundamental frequency of the stimulus, denoted f_1, or at the ***n***th harmonic of the stimulus, f_n. Only coefficients that surpassed the value set by an F-test were accepted and used to reconstruct the transfer functions in panels *A* and *E*, respectively. ***D***, ***H***, Power for the residual response, found by subtracting the stimulus driven components in the power (panels *C* and *G*) from the spectrum (panels *B* and *F*). For the S1 unit, note the excess power near 5 Hz (arrow in panel *D*) that is also present in the spontaneous activity. Note, too, the presence of low-frequency spiking in the residual activity for the M1 unit as well as in the spontaneous activity. ***I–L***, Summary results from a unit to show the peristimulus time histogram (black curve) and the best fit (gray curve) at four different stimulation frequencies: 4, 7, 10, and 15 Hz; at all four frequencies, the modulated spike rate captures only the fundamental frequency of the stimulus. ***M***, Summary of the relative power at f_1 for all motor units in the study. A value of 1 indicates that the unit follows only the fundamental frequency of the stimulus. The height of a bar corresponds to the number of separate units. Panels ***A, B, D, E, F, H, I, J, K***, and ***L*** plot the mean and two standard error limits.

We accept the estimator $\hat{B}_l$ if the F-statistic exceeds $1 - 1/N_t$, where N_t is the number of points in $V(t)$. If $\hat{B}_l$ is significant, we move to another frequency in order to determine the complete set of estimators, denoted $\hat{B}_m$ with $m = 1, \ldots, M$, for all M sinusoidal components. The final spectrum is expressed below:

$$\hat{S}(f) = \sum_{m=1}^{M}\left[\frac{1}{K}\sum_{k=1}^{K}\left|\tilde{V}^{(k)}(f) - \hat{B}_m\tilde{w}^{(k)}\left(f - f_m\right)\right|^2 + \left|\hat{B}_m\right|^2 \delta\left(f - f_m\right)\right].$$

This is shown for the example M1 and S1 units (Figs. 3*C*,*D*,*G*,*H*); the same expansion coefficients are used to construct transfer functions for the response (Figs. 3*A*,*E*). Note the low-frequency peak in the residual spiking activity for both units that matches the respective spontaneous activity (Figs. 3*D*,*H*). Finally, a

NOTES

measure of the purity of pitch determination is given by the ratio of the power at the fundamental to the total power:

$$\hat{C}(f_1) = \frac{\left|\hat{B}(f_1)\right|^2}{\sum_{m=1}^{M}\left|\hat{B}(mf_1)\right|^2}.$$

Of interest, the above measure was found to be $\hat{C}(f_1)=1$ only when the vibrissa stimulation was between 5 Hz and 20 Hz (Fig. 3M)—the natural whisking frequencies. Linear filtering cannot explain such an effect, and it was conjectured that an active filtering process, such as the neurological realization of a phase-locked loop, underlies this process (Kleinfeld et al., 2002).

Case four

A common issue that arises in the analysis of optical imaging data is to remove "fast" noise—that is, fluctuations in intensity that occur on a pixel-by-pixel and frame-by-frame basis. The idea is that the imaging data contain features that are highly correlated in space, such as underlying cell bodies and processes, and highly correlated in time, such as long-lasting responses. The imaging data may thus be viewed as a space-time matrix of random numbers: i.e., the fast noise, with added correlated structure.

With this model in mind, we focus on the case of Ca^{2+} waves in an organotypic culture of rat cortex, which contains both neurons and glia. All cells were loaded with a calcium indicator, and spontaneous activity in the preparation was imaged using a fast-framing (f_{sample} = 500 Hz), low-resolution (100 × 100 pixels) confocal microscope (Fig. 4A).

Imaging data takes the form of a three-dimensional array of intensities, denoted $V(x, y, t)$. We consider expressing the spatial location in terms of a pixel index so that each $(x, y) \rightarrow s$ and the data are now in the form of a space-time matrix $V(s, t)$. This matrix may be decomposed into the outer product of functions of pixel index and functions of time. Specifically,

$$V(s,t) = \sum_{n=1}^{rank\{V\}} \lambda_n \, F_n(s)\, G_n(t),$$

where the rank of $V(s, t)$ is the smaller of the pixel or time dimensions. For example data (Fig. 4A), there are N_t = 1200 frames, or time points, and N_s = 10,000 pixels, so that $rank\{V(s, t)\} = N_t$. The above decom-

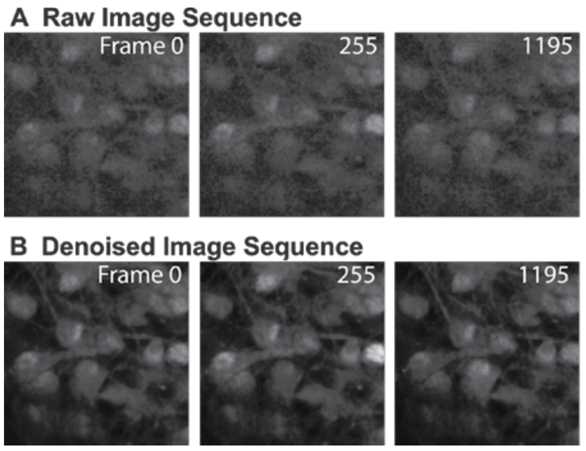


Figure 4. Denoising of spinning-disk confocal imaging data on Ca^{2+} waves in organotypic culture. ***A***, Selected frames from a 1200-frame sequence of 100 × 100 pixel data. ***B***, The same data set after reconstruction with 25 of the 1200 modes. Denoising is particularly clear when the data are viewed as video clips.

position is referred to as an SVD (Golub and Kahan, 1965). The temporal functions satisfy this eigenvalue equation:,

$$\sum_{t'=1}^{N_t} G_n(t') \left[\sum_{s=1}^{N_s} V(s,t)V(s,t')\right] = \lambda_n^2 G_n(t),$$

where the functions $F_n(s)$ and $G_n(t)$ are orthonormal, so that

$$\sum_{t=1}^{N_t} G_m(t)\, G_n(t) = \delta_{nm}$$

and

$$\sum_{s=1}^{N_s} F_m(s)\, F_n(s) = \delta_{nm}.$$

The spatial function that accompanies each temporal function is found by inverting the defining equation, so that the following holds true:

$$F_n(s) = \frac{1}{\lambda_n}\sum_{t=1}^{N_t} V(s,t)\, G_n(t).$$

When this decomposition is applied to the Ca^{2+} imaging data (Fig. 4A), we see that the eigenvalue spectrum has some very large values for the low-order modes but then rapidly falls to a smoothly decreasing function of index (Fig. 5A). (Theoretical expressions for the baseline distribution may be found in Sengupta and Mitra, 1999.) The spatial and temporal modes show defined structure for the first 15 modes; beyond these, the spatial modes appear increasingly grainy, and the temporal modes appear as fast noise (Figs. 5*B*,C).

The utility of this decomposition is that only the lower-order modes carry information. Thus, we can reconstruct the data matrix from only these modes and remove the "fast" noise, as follows:

$$V^{\text{reconstructed}}(s,t) = \sum_{n=1}^{\text{largest signficant mode}} \lambda_n F_n(s)\, G_n(t).$$

Compared with smoothing techniques, the truncated reconstruction respects all correlated features in the data and, for example, does not remove sharp edges. Reconstruction of the Ca^{2+}-wave data highlights the correlated activity by removing fast, grainy-looking variability (Fig. 4*B*).

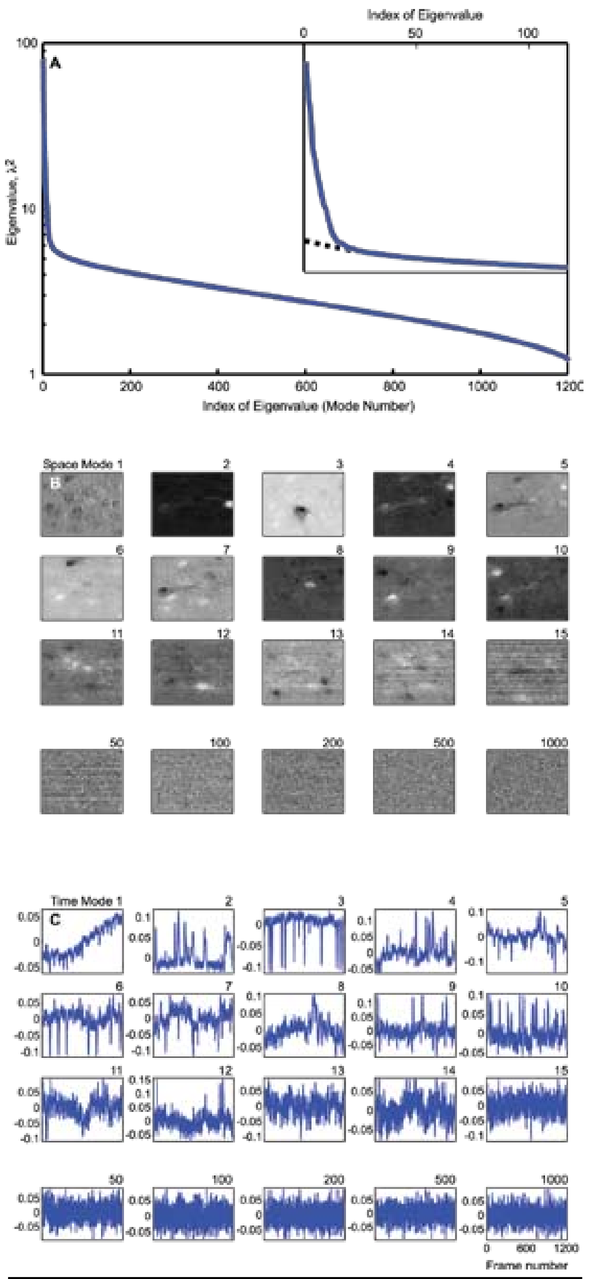


Figure 5. SVD of the imaging sequence in Figure 4. *A*, The spectrum for the square of the eigenvalues for the space and time modes. Note the excess variance in the roughly 25 dominant modes. *B*, Top 15 spatial modes, $F_n(s)$, plus high-order modes. *C*, Top temporal modes, $G_n(t)$.

Case five

The final example concerns the characterization of coherent spatiotemporal dynamics. We return to the use of voltage-sensitive dyes, this time to image the electrical dynamics of turtle visual cortex in response to a looming stimulus. Early work had shown that a looming stimulus led to the onset of ~20 Hz oscillations, the γ-band for turtle, in visual cortex. The limited range of cortical connections suggested this oscillation might be part of wave motion. Yet the raw data, even after denoising and broadband filtering, appears complex (Fig. 5*A*): Regions of net depolarization sweep across cortex, but no simple pattern emerges.

One possible explanation is that cortex supports multiple dynamic processes, each with a unique center frequency, that may be decomposed by an SVD in the frequency domain. In this method (Mann and Park, 1994), the space-time data *V(s, t)* are first projected into a local temporal frequency domain by transforming them with respect to a set of tapers:

$$\tilde{V}^{(k)}(s,f) = \frac{1}{\sqrt{T}} \int_0^T dt\, e^{i2\pi f t}\, w^{(k)}(t)\, V(s,t).$$

The index *k* defines a local frequency index in the band [*f* – Δ*f*/2, *f* + Δ*f*/2]. For a fixed frequency, f_0, an SVD is performed on this complex matrix:

$$\tilde{V}(s,k;f_0) \equiv \left[\tilde{V}^{(1)}(s,f_0), \ldots, \tilde{V}^{(K)}(s,f_0)\right].$$

This yields the following:

$$\tilde{V}(s,k;f_0) = \sum_{n=1}^{\text{rank}\{\tilde{V}\}} \lambda_n \tilde{F}_n(s)\, \tilde{G}_n(k),$$

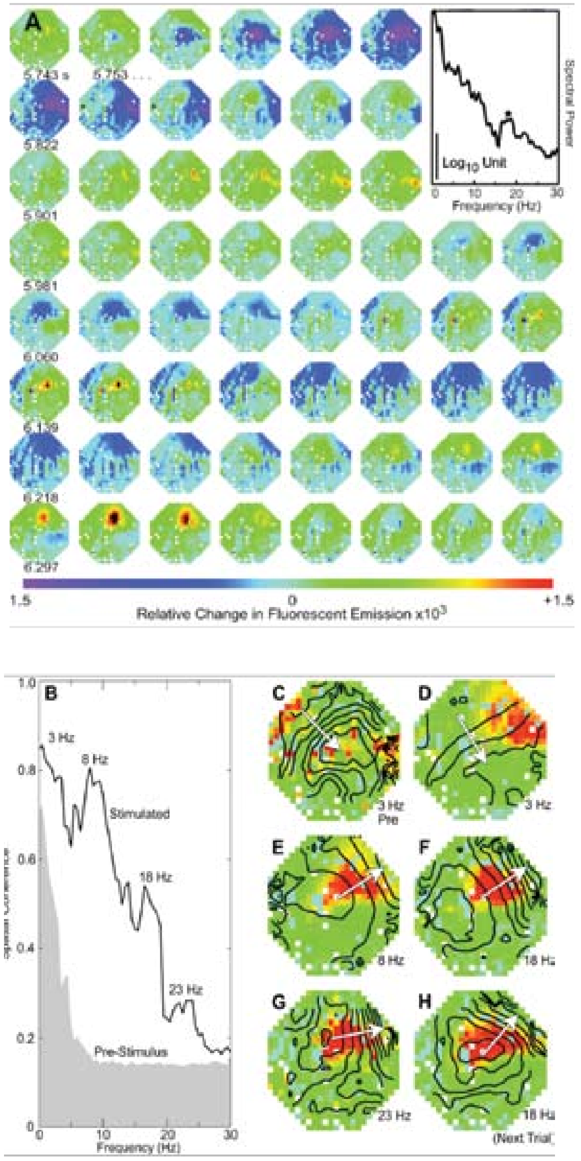


Figure 6. Analysis of single-trial, voltage-sensitive dye imaging data to delineate collective excitation in visual cortex of turtle (Prechtl et al., 1997, their Fig. 2 and Fig. 3, reprinted with permission from Proceedings of the National Academy of Sciences USA). *A*, Response while the animal was presented with a looming stimulus. The data were denoised, low-pass filtered at 60 Hz, and median filtered (400 ms width) to remove a stimulus-induced depolarization. We show every eighth frame (126 Hz); note the flow of depolarization from left to right. Insert is the spectrum for the interval 4.7 to 5.7 s *B*, Coherence, $\hat{C}(f_o)$, over intervals both before and after the onset of the stimulus (T = 3 s; K = 7) estimated at successive frequency bins; $C(f) > 0.14$ indicates significance. *C*, Spatial distribution of amplitude (red for maximum and blue for zero) and phase (π/12 radians per contour; arrow indicates dominant gradient) of the coherence at f = 3 Hz prior to stimulation. *D–G*, Amplitude and phase at f = 3, 8, 18, and 22 Hz during stimulation. *H*, Amplitude and phase at 18 Hz for the next trial (compare panels *F* and *H*).

where the rank is invariably set by *K*. A measure of coherence is given by the ratio of the power of the leading mode to the total power (Fig. 6*B*):

$$\hat{C}(f_0) = \frac{\lambda_1^2(f_0)}{\sum_{k=1}^{K} \lambda_k^2(f_0)}.$$

A completely coherent response leads to $\hat{C}(f_0) = 1$, while for a uniform random process $\hat{C}(f_0) = 1/K$. Where $\hat{C}(f_0)$ has a peak, it is useful to examine the largest spatial mode, $\tilde{F}_1(s)$. The magnitude of this complex image gives the spatial distribution of coherence at f_0, while gradients in its phase indicate the local direction of propagation.

For the example data (Fig. 6*A*), this analysis revealed linear waves as the dominant mode of electrical activity; those at 3 Hz were present with or without stimulation while those at 8 through 23 Hz were seen only with stimulation and propagate orthogonal to the wave at 3 Hz (Figs. 6*C*, *H*). It is of biological interest that the waves at 3 Hz track the direction of thalamocortical input, while those at higher frequencies track a slight bias in axonal orientation (Cosans and Ulinski, 1990) that was unappreciated in the original work (Prechtl et al., 1997).

References

Berg RW, Kleinfeld D (2003) Rhythmic whisking by rat: Retraction as well as protraction of the vibrissae is under active muscular control. J Neurophysiol 89:104-117.

Cacciatore TW, Brodfueher PD, Gonzalez JE, Jiang T, Adams SR, Tsien RY, Kristan Jr. WB, Kleinfeld D (1999) Identification of neural circuits by imaging coherent electrical activity with FRET-based dyes. Neuron 23:449-459.

Cosans CE, Ulinski PS (1990) Spatial organization of axons in turtle visual cortex: Intralamellar and interlamellar projections. J Comp Neurol 296:548-558.

Drew PJ, Duyn JH, Galanov E, Kleinfeld D (2008) Finding coherence in spontaneous oscillations. Nat Neurosci 11:991-993.

Golub GH, Kahan W (1965) Calculating singular values and pseudo-inverse of a matrix. Philadelphia: Society for Industrial and Applied Mathematics.

Hannan EJ (1970) Multiple time series. New York, NY: Wiley.

Jarvis MR, Mitra PP (2001) Sampling properties of the spectrum and coherency of sequences of action potentials. Neural Comp 13:717-749.

Kleinfeld D, Sachdev RNS, Merchant LM, Jarvis MR, Ebner FF (2002) Adaptive filtering of vibrissa input in motor cortex of rat. Neuron 34:1021-1034.

Mann ME, Park J (1994) Global-scale modes of surface temperature variability on interannual to centuries timescales. J Geophys Res 99:25819-25833.

Mitra PP, Bokil HS (2008) Observed brain dynamics. New York: Oxford UP.

Mitra PP, Pesaran B (1998) Analysis of dynamic brain imaging data. Biophys J 76:691-708.

Mitra PP, Pesaran B, Kleinfeld D (1999) Analysis of dynamic optical imaging data. In: Imaging: a laboratory manual (Yuste R, Lanni F, Konnerth A, eds). Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press.

Nir Y, Mukamel R, Dinstein I, Privman E, Harel M, Fisch L, Gelbard-Sagiv H, Kipervasser S, Andelman F, Neufeld MY, Kramer U, Arieli A, Fried I, Malach R (2008) Interhemispheric correlations of slow spontaneous neuronal fluctuations revealed in human sensory cortex. Nat Neurosci 11:1100–1108.

Percival DB, Walden AT (1993) Spectral analysis for physical applications: multitaper and conventional univariate techniques. Cambridge: Cambridge UP.

Prechtl JC, Cohen LB, Mitra PP, Pesaran B, Kleinfeld D (1997) Visual stimuli induce waves of electrical activity in turtle cortex. Proc Natl Acad Sci USA 94:7621-7626.

Sengupta AM, Mitra PP (1999) Distributions of singular values for some random matrices. Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics 60:3389-3392.

Taylor AL, Cottrell GW, Kleinfeld D, Kristan WB (2003) Imaging reveals synaptic targets of a swim-terminating neuron in the leech CNS. J Neurosci 23:11402–11410.

Thomson DJ (1982) Spectral estimation and harmonic analysis. Proc IEEE 70:1055-1096.

Thomson DJ, Chave AD (1991) Jackknifed error estimates for spectra, coherences, and transfer functions. In: Advances in spectrum analysis and array processing (Haykin S, ed), pp 58-113. Upper Saddle River, NJ: Prentice Hall.

Adaptive Spline Smoothing of Neural Data

Robert E. Kass, PhD

Department of Statistics, Center for the Neural Basis of Cognition,
Machine Learning Department, Carnegie Mellon University
Pittsburgh, Pennsylvania

The Need for Smoothing

A function *f(x)* is plotted in the upper left panel of Figure 1. A small amount of noise was added to *f(x)* for 100 values of *x*, yielding the 100 data pairs (*x*, *y*) in the upper right panel. In this case, it is easy to recover *f(x)*: not its precise formula, but a representation that provides a highly accurate value of *f(x)* for every *x* in the given range (–2, 2). When, instead, a large amount of noise was added, the 100 data pairs in the lower left panel were produced. In this case, it is impossible to recover *f(x)* accurately. The prototypical statistical problem of *smoothing* is illustrated by the data in the lower right panel, where a moderate amount of noise was added: There, the general shape of the curve *f(x)* is visible, yet there is sufficient variation so that alternative approaches to curve-fitting give noticeably different results. This is the situation I wish to address here.

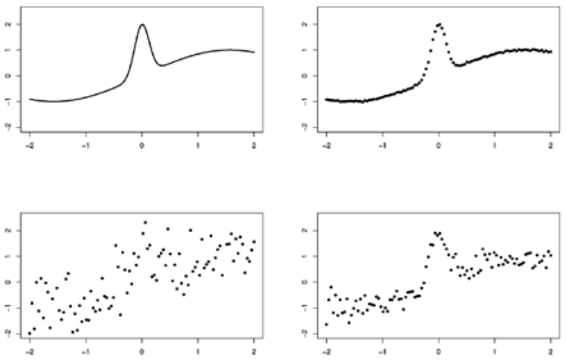


Figure 1. Data simulated from function ***f(x)*** = sin(*x*) + 2 exp(–30x^2). Top left: the function; top right: low noise; bottom left: high noise; bottom right: moderate noise.

This chapter discusses several smoothing methods but emphasizes Bayesian adaptive smoothing splines (BARS), originally proposed by DiMatteo, Genovese, and Kass (2001). A relatively low-noise application of BARS to electro-oculogram data is seen in Figure 2. A somewhat higher-noise application is found in Figure 3, which presents the problem of comparing neural firing-rate functions across different conditions. The value of smoothing for visual display is apparent in Figures 2 and 3, but a compelling statistical reason for smoothing is that it improves accuracy in estimation. Kass, Ventura, and Cai (2003) gave an illustrative example where smoothing the peristimulus time histogram (PSTH) based on 16 trials of data was numerically equivalent to using the raw PSTH based on 224 trials. In some situations, BARS can provide additional boosts in accuracy.

Functions are fundamental conceptual representations, yet they are commonly observed with moderate noise. From a theoretical point of view, though dealing with functions statistically is a somewhat different matter than dealing with numbers, the principles are pretty much the same. An important issue is that the amount of smoothness may vary across the range of the data. The function in Figure 1 varies much more rapidly within the region (–.25, 0.25) than outside it. In this sense, the function is "less smooth" within the region (–.25, 0.25) than outside it. Methods that have a fixed degree of smoothness cannot recover *f(x)* from the data in the bottom right panel. The "adaptive" part of BARS is its ability to vary the amount of smoothness appropriately, throughout the range of the data. This is a good thing when one believes strongly that a function will vary relatively slowly over portions of its domain. The ideas behind BARS are very simple, but the computational technology based on Markov chain Monte Carlo (MCMC) is very powerful. Once this is understood, it is not surprising that BARS often does a good job. On the other hand, BARS has important limitations, which I will indicate.

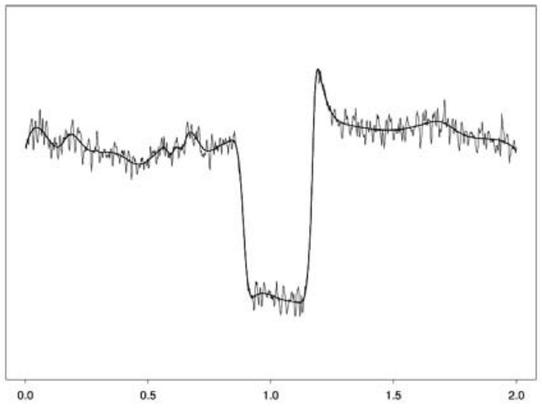


Figure 2. Electro-oculogram and BARS fit to it. Modified with permission from Wallstrom et al. (2002).

Smoothing is, in some contexts, called *filtering*. The term *denoising* is also used, but this (and to a lesser extent, filtering) usually has to do with the very low-noise case. Within the field of statistics, smoothing is also called *nonparametric regression*, which is nonparametric in the sense that the formula for *f(x)* is left unspecified.

Nonparametric Regression Models

The statistical problem of nonparametric regression is to estimate a function $y = f(x)$ from data pairs (x_j, y_j), for $j = 1, \ldots, n$, with the response values y_j assumed to be observations from the "signal plus noise" model here:

$$Y_j = f(x_j) + \varepsilon_j \,. \qquad (1)$$

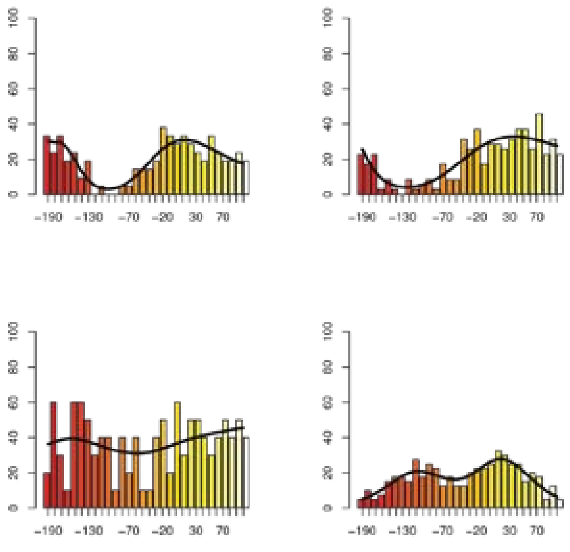


Figure 3. Firing-rate functions from two M1 neurons, under two conditions. In all four plots, BARS has been fitted to a PSTH. Top: Neuron 1 under condition 1 (left) and condition (2). Bottom: Neuron 2 under condition 1 (left) and condition (2). Neuron 2 reacts quite differently under the two conditions, but neuron 1 does not. Units are spikes per second. Modified with permission from Behseta and Kass (2005).

It is usually solved by some variation on least squares, which may be motivated by an assumption that the ε_j variables are independent and Normally distributed. Generalized nonparametric regression replaces the Normal distribution with some other distribution, and modifies the model accordingly. Of particular interest is the Poisson case:

$$\begin{aligned} Y_j &\sim P(y_j \mid \lambda_j) \\ \log \lambda_j &= f(x_j) \end{aligned} \qquad (2)$$

(Kass, Ventura, and Cai, 2003; Kass, Ventura, and Brown, 2005). Let us suppose that the unknown function involves a parameter vector θ. In place of least squares, we may then use the method of maximum likelihood, or some variant of it, based on the following likelihood function:

$$L(\theta) = \prod_{j=1}^{n} e^{-\lambda_j} \lambda_j^{y_j}$$

with θ appearing on the right-hand side implicitly through the relation $\log \lambda_j = f(x_j)$. In applications, we often take the explanatory variable x to be time t.

Mean Squared Error

A method of (generalized) nonparametric regression produces at each x an estimate $\hat{f}(x)$. A good method is one that is likely to make the difference between $\hat{f}(x)$ and $f(x)$ small for every relevant value of x. The simplest and most widely used way to measure likely difference is mean squared error (MSE):

$$MSE(x) = E\left((\hat{f}(x) - f(x))^2\right).$$

Here E stands for expectation, meaning the theoretical average (mean) over hypothetical repetitions of the data-collection process. The values of $MSE(x)$ are then integrated across all relevant values of x to produce the integrated mean squared error (IMSE or MISE). Typically, MSE decreases roughly in proportion to sample size, at least for large samples. Thus, if some method A has half the MSE of method B, method A with sample size n is likely to be about as accurate as method B would be with sample size $2n$. In (2), for large samples, the method of maximum likelihood achieves the minimum possible MSE. Bayesian methods also achieve the minimum possible MSE.

To evaluate the MSE, some additional assumptions must be made. In some simple situations, it is possible to get analytical expressions for the MSE. Otherwise, two alternative numerical methods are used: simulations based on (1) or (2) using a known function; or cross-validation. The latter produces a good estimate of MSE under (1) for large samples, i.e., large n. An extremely important general relation is as follows:

$$MSE = \text{BIAS}^2 + \text{Variance}.$$

Many estimators $\hat{f}(x)$ may be modified to produce small bias when allowed to increase variance, or small variance when allowed to increase bias. This is called the "bias-variance trade-off." In selecting a smoothing method, a small integrated MSE is desirable, but of course computational convenience (availability of software and ease of use in particular situations) is also important.

Popular Smoothing Methods

There are two general approaches to nonparametric regression. The first estimates $f(x)$ by weighting the data (x_i, y_i) according to the proximity of x_i to x in a process called *local fitting*. The methods following the second approach attempt to represent a function $f(x)$ in terms of a set of more primitive functions, such as polynomials, which are usually called basis functions. Many techniques involve *linear smoothers*, meaning that the fitted function values $\hat{f}(x_i)$ are obtained by linear operations on the data vector $y = (y_1, \ldots, y_n)^T$.

Linear smoothers are fast and easy to analyze theoretically. BARS is a nonlinear smoother that uses spline basis functions. Before discussing BARS, I will first very briefly discuss some linear smoothers.

In ordinary linear regression, the regression line is the expectation of Y as a function of$x$. Here we have: and this is extended to some newly observed value of

$$E(Y_i) = \beta_0 + \beta_1 x_i ,$$

x by writing the following:

$$E(Y \mid x) = \beta_0 + \beta_1 x . \qquad (3)$$

In words, the average value of Y at x (averaged across hypothetical replications) is linear in x. To get $E(Y \mid x)$, in principle, we take an infinite collection of replications at x and find their sample mean. In practice, with a finite sample of data, we could instead take a window of width δ and average all of the y_j values that correspond to x_j values in the interval $(x - \delta/2 , x + \delta/2)$. When we generalize (3) to

$$E(Y \mid x) = f(x) , \qquad (4)$$

we could, again in principle, also estimate $f(x)$ by averaging y_i values corresponding to x in $(x - \delta/2 , x + \delta/2)$. For very large data sets (relative to the smoothness of $f(x)$, with good estimation requiring more data when $f(x)$ is more irregular), such local averaging will succeed. An immediate concern, however, is how to choose the size of the window used for averaging. Furthermore, in estimating $f(x)$ even with moderate sized data sets, it is possible to improve on the arithmetic mean among y_i values corresponding to x_i near x. The idea of *local fitting* is to consider x_i values that are somewhat more distant from x, but to weight the various x_i values according to their proximity to x.

When a weighted average of y_j values is used, with the weights defined by a suitable function $w_j = K((x - x_j) / h)$, the result is called *kernel regression*, with $K(u)$ being the kernel. The constant h is usually called the *bandwidth* (by analogy with similar methods in spectral analysis). The most commonly used kernel is the $N(0, 1)$ pdf, in which case h effectively plays the role of a standard deviation. That is, we have $w_i \propto K_h(x - x_i)$, where $K_h(u)$ is the $N(0, h^2)$ pdf (and the proportionality constant is h). More generally, any pdf could be used as a kernel. The choice of bandwidth h in kernel regression is important: When h is small, the estimate tends to follow the data closely but is very rough, whereas when h is large, the estimate becomes smooth but may ignore places where the function seems to vary. Bandwidth selection involves a "bias versus variance" trade-off: Small h reduces bias (and increases variance) while large h reduces variance (but increases bias). Theoretical arguments can provide heuristics for selecting the bandwidth.

A second idea in local fitting is to solve a weighted least-squares problem defined at x by suitable weights $w_i = w_i(x)$. In particular, *local linear regression* at x minimizes

$$WSS(x) = \sum_{i=1}^{n} w_i (y_i - \beta_0 - \beta_1 (x - x_i))^2 \qquad (5)$$

where the weights w_i are defined in terms of a kernel. A Normal pdf may be used as the kernel, but an alternative is $K(u) = (1 - |u|^3)^3$ for $|u| < 1$ and $K(u) = 0$ otherwise. The latter form of the kernel is used as the default in some available software. Extensive study of this methodology has shown that local linear regression is effective in many situations. As with kernel regression, in local polynomial regression there remains a choice of bandwidth. (Further information about bandwidth selection, and local fitting in general, can be found in Loader, 1999, and at http://locfit.herine.net.) An important feature of local polynomial regression is that it may be extended to non-Normal families such as Binomial and Poisson. Instead of minimizing the locally weighted sum of squares in (5), a locally weighted likelihood function may be maximized.

Turning to basis functions, a particularly simple idea is to try to approximate $f(x)$ by a polynomial. This turns out to perform rather badly, however, for many common functions. A possible solution is to glue together several pieces of polynomials. If the pieces are joined in such a way that the resulting function remains smooth, then it is called a spline. In particular, a cubic spline on an interval $[a,b]$ is a function $f(x)$ made up of cubic polynomials joined at a set of knots, $\zeta_1, \zeta_2, \ldots , \zeta_p$, with $a < \zeta_1 < \zeta_2 < \ldots < \zeta_p < b$, such that $f(x)$, $f'(x)$, and $f''(x)$ are continuous (at the knots, and therefore throughout the interval $[a,b]$).

It is easy to define a cubic spline having knots at $\zeta_1, \zeta_2, \ldots , \zeta_p$. Let $(x - \zeta_j)_+$ be equal to $x - \zeta_j$ for $x \geq \zeta_j$ and 0 otherwise. Then the function

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - \zeta_1)^3_+ + \beta_5 (x - \zeta_2)^3_+ + \ldots + \beta_{p+3} (x - \zeta_p)^3_+ \qquad (6)$$

is twice continuously differentiable, and is a cubic polynomial on each segment $[\zeta_j, \zeta_{j+1}]$. Furthermore, the nonparametric regression model (1) becomes an ordinary linear regression model so that standard least-squares software may be used to obtain spline-based curve fitting. For example, suppose we have 7 data values $y_1, \ldots , y_7$ observed at 7 x values, $(x_1, \ldots , x_7) = (-3, -2\ -1, 0, 1, 2, 3)$ and we want to fit a spline with knots at $\zeta_1 = -1$ and $\zeta_2 = 1$. Then we define $y = (y_1, \dots, y_7)^T$, $x_1 = x = (-3, -2, -1, 0, 1, 2, 3)^T$, $x_2 = x^2 = (9, 4, 1, 0, 1, 4, 9)^T$, $x_3 = x^3 = (-27, -8, -1, 0, 1, 8, 27)^T$, $x_4 = (x - \zeta_1)_+ = (0, 0, 0, 1, 8, 27, 64)^T$, $x_5 = (x-\zeta_2)_+ = (0, 0, 0, 0, 0, 1, 8)^T$, and we regress y on x_1, x_2, x_3, x_4, x_5.

An important caveat in applying (6), however, is that the variables $x_1, x_2, \dots x_{p+3}$ will be highly correlated. There are two solutions to this problem. The first is to orthogonalize the x variables by sequentially replacing each variable with residuals from regression on previous variables. The second is to use a different version of splines, known as *B-splines*. These have similar properties but do not produce high correlation among the regression variables. A variant of B-splines, known as *natural splines*, assumes the second derivatives of the curve are zero at the boundaries a and b. Natural splines are often recommended.

When splines are used in regression models, they are frequently called *regression splines*. Regression splines are easy to use in (1), and also in (2) and other generalized regression models, because the problem of spline fitting becomes one of linear regression or generalized linear regression. This, however, assumes that the knot set $\zeta_1, \zeta_2, \dots, \zeta_p$ has been determined. The choice of knots can be consequential: with more knots, the spline has greater flexibility but also provides less smoothness; in addition, the placement of knots can be important. Figure 4 displays three alternative spline fits: splines with 5 and 15 knots, with locations equally spaced according to the quantiles of x (so, for example, 5 knots would be placed at the ⅙, ⅓, ½, ⅔, ⅚ quantiles), and a spline with 7 knots chosen by eye. The spline with 7 knots fits well because 5 knots are placed in the middle of the range, where the function variation is large, while only 2 are placed on the flanks where the variation is small.

One way to avoid the problem of knot selection is to use a large number of knots but to reduce, or "shrink," the values of the coefficients. One intuition here is that using a large number of knots in a regression spline would allow it to follow the function well, but would make it very wiggly; reducing the size of the coefficients will tend to smooth out the wiggles. A second intuition is obtained by replacing the least-squares problem of minimizing the sum of squares with the *penalized least-squares* problem of minimizing the penalized sum of squares:

$$PSS = \sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \int (f''(x))^2 dx$$

where λ is a constant. Here, the squared second derivative is a *roughness penalty*: Wherever $(f''(x))^2$ is large, the function is fluctuating substantially, and the integral of this quantity is a measure of the total fluctuation, or roughness. Thus, the value of the coefficient vector β^* that minimizes *PSS* will achieve some compromise between fitting the y_i values and keeping the function smooth. It turns out that the solution to the penalized least squares problem is a cubic spline with knots at every value of x_i, but with coefficients that are smaller in magnitude than those of the regression spline with knots at every x_i (which would correspond to taking $\lambda = 0$). This solution is called a *smoothing spline*. Smoothing spline technology has a strong theoretical foundation and is among the most widely used methods for nonparametric regression. There is also much well-developed software for smoothing splines. An alternative to smoothing splines is *penalized regression splines* or *P-splines* in which a modest number of knots is used. There is also substantial literature and software based on this approach.

One defect of smoothing spline technology, and of many other nonparametric methods, is that it assumes the degree of smoothness of $f(x)$ remains about the same across its domain, i.e., throughout the range

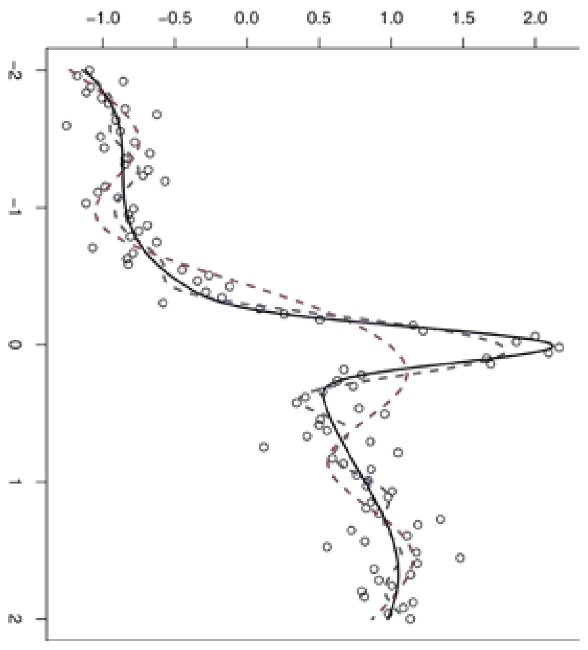


Figure 4. Three cubic spline fits to data generated from the same test function as in Figure 1. Splines with 5 and 15 knots are shown (red dashed and blue dashed lines), with knot locations selected by default in ***R***. The spline with 5 knots provides more smoothing than the spline with 15 knots and, as a result, does a poorer job of capturing the peak in the function. The spline shown in the solid line has 7 knots chosen to be $\zeta = (-1.8, -0.4, -0.2, 0, 0.2, 0.4, 1.8)$.

of x values. An alternative is to devise a method that selects good knot sets based on the data.

Bayesian Adaptive Regression Splines

The essential idea behind BARS is to assume in (1) that $f(x)$ is a cubic spline with knot set $\zeta = (\zeta_1, \ldots, \zeta_p)$, and to determine ζ by applying Bayes' Theorem via MCMC. There are many theoretical arguments supporting the Bayesian approach to problems of statistical inference (http://www.bayesian.org). From a practical point of view, Bayesian inference is based on a *posterior* probability distribution on ζ, and the uncertainty thereby represented is easily propagated to quantities of interest, such as the time at which maximal firing rate occurs; this, in turn, leads to estimates, standard errors, and confidence intervals. With MCMC as a computational engine, Bayesian inference has been enormously useful in a wide variety of applications.

The particular version of MCMC used in BARS is called *reversible-jump* MCMC, which is well suited to problems where dimensionality of the parameter space is uncertain (here, the number of spline basis vectors is determined by the number of knots, which is unknown). DiMatteo and colleagues have shown in several examples that BARS could perform dramatically better than related methods, which themselves performed better in many examples than competitors such as wavelet-based methods. BARS has been used in applications in neurophysiology, imaging, genetics, and EEG analysis. Furthermore, simulation results have indicated that, with reasonable sample sizes, posterior confidence intervals produced by BARS can have very nearly the correct frequentist coverage probabilities. In addition, as a simulation-based procedure, BARS can be less sensitive than are deterministic methods to very small perturbations of the data, such as those produced by moving data across computing platforms. We have checked our results using BARS across multiple computing platforms and have obtained highly reproducible results.

The idea of MCMC is that it generates a sequence of knot sets $\zeta^{(g)}$ according to a probabilistic scheme (a Markov chain). For a given $\zeta^{(g)}$, a "proposal" ζ^* is generated (from a proposal probability distribution). If the posterior probability density at ζ^* increases relative to its value at $\zeta^{(g)}$, then ζ^* becomes $\zeta^{(g+1)}$. Otherwise, $\zeta^{(g+1)}$ is set either to ζ^* or $\zeta^{(g)}$ with a certain probability. After the algorithm has been run for a while (after its "burn-in," so that it has "converged"), the sequence of knot sets $\zeta^{(g)}$ may be considered random draws from the posterior distribution on ζ, representing the uncertainty in ζ due to the noise in the data.

For each draw $\zeta^{(g)}$ from the posterior distribution of ζ, a draw of the spline coefficient vector $\beta_\zeta^{(g)}$ is obtained from the conditional posterior of β_ζ, conditionally on $\zeta^{(g)}$. From $\beta_\zeta^{(g)}$ we obtain fitted values $f^{(g)}(t)$ for selected t and these, in turn, may be used to produce a draw $\phi^{(g)}$ from the posterior distribution of any characteristic $\phi = \phi(f)$ (such as the value at which the maximum of $f(t)$ occurs). Thus, the key output of BARS is the set of vectors $f^{(g)} = (f^{(g)}(t_1), f^{(g)}(t_2), \ldots, f^{(g)}(t_k))$ for MCMC iterates $g = 1, \ldots, G$, each $f^{(g)}$ being a vector of fits along a suitable grid $t_1, t_2, \ldots, t_k$ that covers the interval $[a,b]$. The user may sample from the posterior distribution of any functional ϕ simply by evaluating $\phi^{(g)} = \phi(f^{(g)})$. For instance, a sample from the posterior distribution of the location of the maximum of $f(t)$ is obtained by finding the location of the maximum of $f^{(g)}$ for each g. This latter computation is performed in a postprocessing environment such as *R* or Matlab. We have implemented two versions of BARS, one using a Normal noise distribution in (1) and the other using the Poisson model (2).

A key theoretical feature of the default BARS implementation is that it effectively chooses among alternative knot sets ζ by applying the Bayes information criterion (BIC). BIC is known to be conservative (it tends to oversmooth) but consistent; that is, it chooses the correct model for large samples. A key practical feature is that BARS uses continuous distributions for knot placement together with a locality heuristic that attempts to place knots near existing knots. In addition to making the algorithm relatively efficient, one consequence is that knots can be placed essentially on top of each other, allowing BARS to fit discontinuous functions.

Comments

Two caveats should be given here. First, when analyzing spike train data, BARS is intended for use in smoothing the PSTH *after pooling across trials*. In order to use BARS for within-trial analyses, it must be modified. Second, BARS is relatively slow. For long sequences of data, it may prove too time-consuming to be effective. In such cases, I recommend the other methods discussed here, especially local linear fitting or penalized splines, though the easiest method—kernel regression (often known as Gaussian filtering)—is often just as effective.

Our experience to date with BARS has been very favorable. It offers advantages in situations like the moderate noise case described in Figure 1, where the degree of smoothness varies across the domain of the function. Software and additional references may be found at http://www.stat.cmu.edu/~kass/bars.

NOTES

References

Behseta S, Kass RE (2005) Testing equality of two functions using BARS. Stat Med 24:3523-3534. DiMatteo I, Genovese CR, Kass RE (2001) Bayesian curve fitting with free-knot splines. Biometrika 88:1055-1073.

Kass RE, Ventura V, Cai C (2003) Statistical smoothing of neuronal data. Network 14:5-15.

Kass RE, Ventura V, Brown EN (2005) Statistical issues in the analysis of neuronal data. J Neurophysiol 94:8-25.

Loader C (1999) Local regression and likelihood. New York: Springer.

Wallstrom GL, Kass RE, Miller A, Cohn JF, Fox NA (2002) Correction of ocular artifacts in the EEG using Bayesian adaptive regression splines. In: Case studies in Bayesian statistics, Vol VI (Gatsonis C, Carriquiry A, Higdon D, Kass RE, Pauler D, Verdinelli I, eds), pp. 351-365. New York: Springer.

Point Process Models for Neural Spike Trains

Uri T. Eden, PhD

Department of Mathematics and Statistics, Boston University
Boston, Massachusetts

Introduction

Spike trains are fundamental to information processing performed by the brain, and point processes form the foundation for modeling spike train data. Since the spikes from a single neuron have stereotyped waveforms, neural representations of biological and behavioral signals are based on the frequency and timing of spike events. Analyzing data of this sort presents its own unique challenges and poses its own set of questions. How can biological and behavioral signals be represented by spike sequences? What types of stochastic models are appropriate for explaining structure within these data? How can we measure how well a particular model is performing? In order to address these questions, we require a mathematical structure that can handle data of this nature.

A temporal point process is a stochastic, or random, time-series of binary events that occurs in continuous time (Daley and Vere-Jones, 2003). They are used to describe data that are localized at a finite set of time points. As opposed to continuous-valued processes, which can take on any of countless values at each point in time, a point process can take on only one of two possible values, indicating whether or not an event occurs at that time. In a sense, this makes the probability models used to describe point process data relatively easy to express mathematically. However, point process data are often inappropriately analyzed, because most standard signal-processing techniques are designed primarily for continuous-valued data. A fundamental understanding of the probability theory of point processes is vital for the proper analysis of neural spiking data (Brown et al., 2003; Brown et al., 2004; Kass et al., 2005).

A Point Process May Be Specified in Terms of Spike Times, Interspike Intervals, or Spike Counts

Let $S_1, S_2, \ldots$ be random variables describing the occurrence times or spike times of a point process. A realization of a point process is the event $S_1 = s_1, S_2 = s_2, \ldots$ for some collection of times $0 < s_1 < s_2 < \ldots$. Let X_1, $X_2, \ldots$ be a set of random variables describing the possible waiting times between occurrences, or interspike intervals. We can compute the waiting times from the spike times by taking the difference between subsequent spike times. Mathematically, $X_1 = S_1$ and $X_i = S_i - S_{i-1}$. Similarly, we can compute the spike times by taking the cumulative sum of all the waiting times. That is, $S_n = \sum_{i=1}^{n} X_i$.

Clearly, there is a one-to-one correspondence between any set of spike times and any set of interspike intervals.

Another useful way to describe a set of spiking observations is in terms of the number of spikes observed over any time interval. We define $N(t)$, the counting process, as the total number of spikes that have occurred up to and including time t (Fig. 1). The counting process gives the number of spikes observed in the interval $(0,t]$. If we let $\Delta N_{(t_1,t_2)}$ denote the total number of spikes observed in an arbitrary interval $(t_1,t_2]$, then we can compute this from the counting process, $\Delta N_{(t_1,t_2)} = N(t_2) - N(t_1)$. $\Delta N_{(t_1,t_2)}$ is sometimes called the increment of the point process between t_1 and t_2. We see that keeping track of the times at which the counting process increases is equivalent to keeping track of the spike events. Therefore, characterizing the spike events is equivalent to characterizing the counting process, and vice versa.

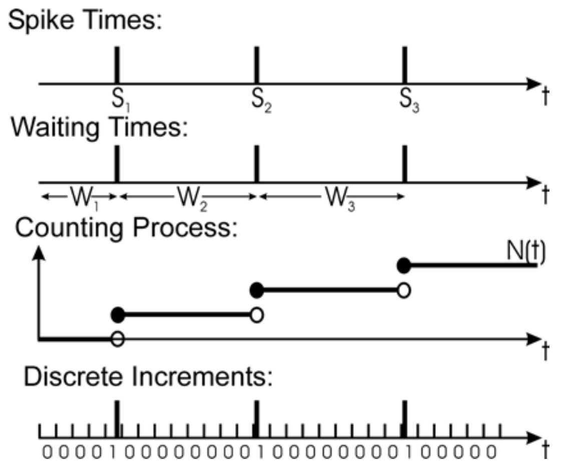


Figure 1. Multiple specifications for point process data.

The spike time, interspike interval, and counting processes are all continuous time specifications of a point process. It is often useful, both for developing intuition and for constructing data analysis methods, to consider point processes in discrete time. One approach for constructing a discrete-time representation of a point process is to take the observation interval $(0,T]$ and break it up into n small, evenly spaced bins. Let $\Delta t = T/n$, and $t_k = k \cdot \Delta t$, for $k = 1, \ldots, n$. We can now express a spike train in terms of discrete increments $\Delta N_k = N(t_k) - N(t_{k-1})$, which count the number of spikes in a single bin. If Δt is small enough so that the process cannot fire more than one spike in a single bin, then the set of increments $\{\Delta N_k\}_{k=1,\ldots,n}$ is just a sequence of zeros and ones indicating in which bins spikes are observed (Fig. 1).

NOTES

NOTES

Spike Data Often Display History-Dependent Behaviors

One of the simplest, most commonly used classes of neural spiking models is the Poisson process. Poisson processes are characterized by lack of memory, meaning that the probability distribution of spiking at any point in time is independent of all previous spiking activity. This is appealing from a mathematical point of view because it simplifies the calculation of the likelihood and other distributions used in data analysis. In some cases, especially when spikes are rare compared with the time scale of the intrinsic membrane dynamics, or the effect of history has been averaged out by combining multiple spike trains, Poisson processes can accurately describe spiking activity. For example, when the spontaneous spiking activity from multiple neurons is recorded on a single electrode and left unsorted, the past firing of one neuron may have no effect on the firing probabilities of the other neurons, so that the combined spiking activity shows little to no history dependence.

However, Poisson processes rarely furnish realistic models for spike train data. In particular, the biophysical properties of ion channels limit how fast a neuron can recover immediately following an action potential, leading to a refractory period during which the probability of firing another spike is zero immediately after and then significantly decreased further after the previous spike. This is perhaps the most basic illustration of history dependence in neural spike trains. Bursting is a more complicated, history-dependent neural behavior characterized by short sequences of spikes with small interspike intervals (ISIs). In addition, spiking activity can display oscillatory behavior. For example, neurons in the CA1 region of rodent hippocampus tend to fire at particular phases of the EEG theta rhythm (Buzsaki et al., 1983). To describe accurately the spiking structure in these neurons, the probability of a spike occurring at a given time must depend on recent past spiking activity.

The Conditional Intensity Function Specifies the Joint Probability Density of Spike Times for a General Point Process

Since most neural systems have a history-dependent structure that makes Poisson models inappropriate, it is necessary to define probability models that account for history dependence. Any point process can be completely characterized by its conditional intensity function, $\lambda(t \mid H_t)$ (Daley and Vere-Jones, 2003), defined as follows:

$$\lambda(t \mid H_t) = \lim_{\Delta t \to 0} \frac{\Pr(\Delta N_{(t,t+\Delta t]} = 1 \mid H_t)}{\Delta t}, \qquad (1)$$

where $\Pr(\Delta N_{(t,t+\Delta]} = 1 \mid H_t$ is the instantaneous conditional probability of a spike, and H_t is the history of the spiking activity up to time t. Since the probability of a spike in any interval must be nonnegative, so too must be the conditional intensity function. This conditional intensity function expresses the instantaneous firing probability and implicitly defines a complete probability model for the point process. It will therefore serve as the fundamental building block for constructing the likelihoods and probability distributions needed for point process data analysis.

By construction, the conditional intensity is a history-dependent rate function because it defines a probability per unit time. If $\Pr(\Delta N_{(t,t+\Delta]} = 1 \mid H_t$ is independent of history, then the point process reduces to a Poisson process; therefore, the conditional intensity generalizes the concept of a rate function for a Poisson process. It is important to realize that the conditional intensity can be a stochastic process itself, since it can depend on spiking history, which is stochastic. A conditional intensity function that depends on history or on any other stochastic process is often called a *conditional intensity process*, and the resulting point process is called a *doubly stochastic point process*.

Additional important intuition behind the conditional intensity function can be gained by choosing Δt to be a small time interval and re-expressing Equation 1 as follows:

$$\Pr(\Delta N_{(t,t+\Delta]} = 1 \mid H_t) \approx \lambda(t \mid H_t)\Delta t. \qquad (2)$$

Equation 2 states that the conditional intensity function multiplied by Δt gives the probability of a spike event in a small time interval Δt.

If we make Δt sufficiently small so that no more than one spike can be fired in a single bin, then the probability of a spike occurring can be analyzed as a Bernoulli process in which, for any small interval, the probability of a spike is the following:

$$\Pr(\text{spike in } [t,t+\Delta t) \mid H_t) \approx \lambda(t \mid H_t)\Delta t, \qquad (3)$$

and the probability of no spike is as follows:

$$\Pr(\text{no spike in } [t,t+\Delta t) \mid H_t) \approx 1 - \lambda(t \mid H_t)\Delta t. \quad (4)$$

This is one sense in which the conditional intensity function characterizes a spike train. We can think of any neural point process as a local Bernoulli process with a spiking probability determined by the conditional intensity function. This is one reason that point processes are conceptually simple to

NOTES

understand. At each instant, you either observe a spike or you don't. The probability of observing a spike can be a function of past spiking activity, as well as other stochastic or time-varying signals, and is characterized by the conditional intensity.

By using this Bernoulli approximation for the increments and taking the limit as $\Delta t \to 0$, it is easy to show that in an observation interval $(0,T]$, the joint probability density of observing a spike train with spikes occurring at the times $s_1,\ldots,s_{N(T)}$ is the following:

$$f_{s_1,\ldots,s_{N(T)}}(s_1,\ldots,s_{N(T)}) = \prod_{i=1}^{N(T)} (\lambda(s_i \mid H_{s_i})) \exp\{-\int_0^T \lambda(t \mid H_t)dt\}, \quad (5)$$

where $N(T)$ is the total number of spikes observed in the interval $(0,T]$, and $S_{N(T)}$ is the time of the last observed spike (Snyder and Miller, 1991).

One way to interpret Equation 5 is to look at the two terms in the product on the right side separately. The $\prod_{i=1}^{N(T)} (\lambda(s_i \mid H_{s_i}))$ term characterizes the distribution of firing at exactly the observed spike times, $s_1,\ldots,s_{N(T)}$. The $\exp\{-\int_0^T \lambda(t \mid H_t)dt\}$ term gives the probability of not firing any other spikes in the observation interval $(0,T]$. Therefore, Equation 5 describes the distribution of firing only at the observed spike times, and nowhere else. This joint probability density fully characterizes a spike train, and will be the backbone for most of our statistical analyses. For a parametric model of the conditional intensity and an observed spike train, Equation 5 represents the data likelihood as a function of the model parameters.

A Point Process Model Expresses the Conditional Intensity As a Function of Time, History, and Other Variables

Equation 1 defines the conditional intensity function in terms of the instantaneous firing probability of the point process. However, this probability distribution is typically unknown. Therefore, rather than constructing the conditional intensity based on the firing probability, we typically write down a model for the conditional intensity, which implicitly defines the probability model for any spiking data. Thus, a conditional intensity model provides a way to express the probability model for a spiking process.

The first step in writing down a conditional intensity model for a point process is determining what factors, or covariates, can influence the occurrence times of that process. We have already seen that spiking history often plays an important role in determining when the next spike will occur, and is therefore one class of covariates that should naturally be considered for neural point process models. If the spike train being modeled comes from a larger ensemble of recorded neurons that interact with each other, it may be useful to consider the firing histories of the other neurons in the model as well. For many experiments dealing with spiking data, there are other signals, or external covariates, besides history terms that affect the point process. These external covariates are often recorded simultaneously with the point process. For example, in any stimulus-response experiment, it is expected that some function of the stimulus affects the firing probability. Once we establish which covariates can influence the spike times of the point process, we next define a model for the conditional intensity as a function of those covariates. For example, if we have a point process with a firing probability that changes as a function of time, as a function of some external covariate, $x(t)$, and is history-dependent, a conditional intensity model for that process would be an expression of the form $\lambda(t \mid H_t) = g(t,x(t),H_t)$, where $g(t,x(t),H_t)$ is any non-negative function. The functional relation between a neuron's spiking activity and these biological and behavioral signals is often called the neuron's *receptive field*. In the examples below, we look at some simple models for specific neural receptive fields.

Example 1: Simple History-Dependent Spiking Model

To illustrate the effect of spiking history on current spiking probability, we define a conditional intensity model that is solely a function of recent past spiking activity as seen below:

$$\lambda_k = \exp\{\alpha_0 + \sum_{j=1}^{4} \alpha_j \Delta N_{k-j}\}. \quad (6)$$

Similar history-dependent models with higher-order temporal dependence have been studied by Ogata (1988), Brillinger (1988), and Truccolo et al. (2005). If Δt = 1 msec, then the spiking probability, $\lambda_k \Delta t$, depends on the spike occurrences in the last 4 msec. This model has four covariates, ΔN_{k-1}, ΔN_{k-2}, ΔN_{k-3}, and ΔN_{k-4}, and five parameters, α_0, α_1, α_2, α_3, and α_4. If we take $\alpha_0 = \log(10)$, $\alpha_1 = -100$, $\alpha_2 = -2$, $\alpha_3 = -0.5$, and $\alpha_4 = -0.1$ we see that these values of the coefficients allow a spike train process with an absolute and relative refractory period. If at any point in time, no spikes have occurred in the past 4 msec, then the firing intensity is $\lambda_k = \exp\{\log(10)\} = 10$ spikes per second. If a spike has occurred in the past millisecond, then the firing intensity drops to around $\lambda_k = \exp\{-97.7\}$, which is negligibly small. In other words, it is virtually impossible for this process to fire a spike within 1 msec of a previous spike. If a spike occurred 2 msec previously, and no other spike is present in the 4 msec history, then the firing intensity is $\lambda_k = \exp\{\log(10)-2\} \approx 1.35$ spikes per second. This is a substantial drop from the baseline firing rate of 10 spikes per second, but not negligibly small as it was immediately after a spike. As a spike recedes into the past, its inhibitory effect on the current spiking activity diminishes. We say that this neuron has an absolute refractory period of 1 msec, when it cannot fire, and a relative refractory period of 4 msec, when the probability of firing is decreased. Under this model, if we had one spike 2 msec in the past and another spike 4 msec in the past, then the inhibitory effects of each past spike combine and $\lambda_k = \exp\{\log(10)-2-.1\} \approx 1.22$ spikes per second—less than the intensity caused by either past spike individually. This simple example shows that the precise timing of the previous spiking activity can alter current spiking propensity, and that this can be modeled with a conditional intensity process.

Example 2. Conditional intensity model for a hippocampal place cell

Hippocampal place cells have firing patterns that relate to an animal's location within an environment (O'Keefe and Dostrovsky, 1971). Therefore, a place field model should describe the conditional intensity as a function of the animal's location at each point in time. Figure 2A shows the spiking activity of a place cell that fires maximally at a point southwest of the center of a circular environment.

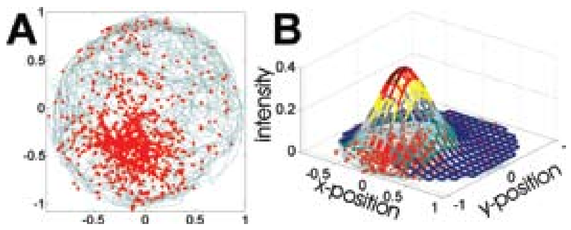


Figure 2. Spiking activity of a rat hippocampal place cell during a free-foraging task in a circular environment. *A*, Visualization of animal's path (blue) and locations of spikes (red). *B*, Gaussian place field model for this neuron with parameters fit by maximum likelihood.

Place fields of this type have been successfully modeled with conditional intensity models that have a Gaussian shape with respect to position. For example, we can construct a conditional intensity model of the following form.

$$\lambda(t) = \exp\left\{\alpha - \frac{1}{2}\begin{pmatrix} x(t)-\mu_x & y(t)-\mu_y \end{pmatrix}\begin{pmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{pmatrix}^{-1}\begin{pmatrix} x(t)-\mu_x \\ y(t)-\mu_y \end{pmatrix}\right\} \qquad (7)$$

The covariates for this model are $x(t)$ and $y(t)$, the animal's x- and y-position. The model parameters are$(\alpha, \mu_x, \mu_y, \sigma^2_x, \sigma^2_y, \sigma_{xy})$, where (μ_x, μ_y) is the center of the place field, $\exp\alpha$ is the maximum firing intensity at that point, and σ^2_x, σ^2_y, and σ_{xy} express how the intensity drops off away from the center. It is important to note that it is the shape of the place field that is Gaussian, not the distribution of the spiking activity, which is a point process.

If we observe the animal's location and the spiking activity of a hippocampal place field, we can plug the conditional intensity model in Equation 7 and the observed spikes into the joint probability in Equation 5, to obtain the data likelihood as a function of the model parameters. We can then find the parameters that maximize this likelihood function. The maximum likelihood fit for the place field shown in Figure 2 is illustrated in panel *B*.

Generalized Linear Models Are a Flexible Class of Spiking Models That Are Easy to Fit by Maximum Likelihood

Linear regression models provide a simple methodology for analyzing relationships between continuous valued data and a set of explanatory variables. Analogously, generalized linear models (GLMs) provide a simple, flexible approach to modeling relationships between spiking data and a set of covariates to which they are associated.

For neural spiking processes, a GLM can be constructed by expressing the conditional intensity as the exponential of a linear combination of general functions of the covariates whose effects we want to model:

$$\lambda(t \mid H_t) = \exp\left\{\sum_{i=1}^{n} \theta_i f_i(x_1, \ldots, x_k)\right\}. \qquad (8)$$

where $\theta = (\theta_1, \ldots, \theta_n)$ is a vector of n model parameters, $x_1, \ldots, x_k$ is a collection of covariates that are related to the spiking activity, and $f_1, \ldots, f_n$ is a collection of functions of those covariates. Notice that the linearity refers to the model parameters and not the model covariates. Therefore, GLMs can capture nonlinear relationships between the spiking activity and these covariates.

GLMs have a number of specific theoretical and computational advantages. It is easily shown that, for a GLM, the likelihood function in Equation 5 will be convex, as a function of the model parameters. This means that maximum likelihood methods will be easy to implement and guaranteed to converge to a global maximum. Additionally, the GLM framework is integrated into several standard mathematical and statistical packages. These properties make the GLM framework ideal for rapidly assessing the possible relevance of a large number of covariates on the spiking properties of neurons whose receptive fields have not previously been fully characterized (Truccolo et al., 2005).

Time Rescaling Provides Natural Approaches for Determining the Goodness-of-Fit of Neural Spiking Models

One essential component of any statistical modeling analysis is to verify that the model accurately describes the structure observed in the data. Measuring quantitatively the agreement between a proposed model for spike train data is a more challenging problem than for models of continuous-valued processes. Standard distance measures applied in continuous data analyses, such as average sum of squared errors, are not designed for point process data. One alternative solution to this problem is to apply the time-rescaling theorem (Papangelou, 1972; Ogata, 1988; Brown et al. 2002) to transform point processes data into continuous measures and then assess goodness-of-fit.

Given a point process with conditional intensity function $\lambda(t|H_t)$ and occurrence times $0 < s_1 < s_2$, …, $< s_{N(T)} \leq T$, define

$$z_1 = \int_0^{s_1} \lambda(t|H_t)dt, \text{ and}$$

$$z_j = \int_{s_{j-1}}^{s_j} \lambda(t|H_t)dt \text{ , for } j=2,\ldots,N(T). \qquad (9)$$

Then these z_j are independent, exponential random variables with rate parameter 1.

This result is called the *time rescaling theorem* because we can think of the transformation as stretching and shrinking the time axis based on the value of the conditional intensity function. If $\lambda(t|H_t)$ is constant and equal to 1 everywhere, then this is a simple Poisson process with independent, exponential ISIs, and time does not need to be rescaled. Any time when $\lambda(t|H_t)$ is less than 1, the z_j values accumulate slowly and represent a shrinking of time, so that distant spike times are brought closer together. Likewise, any time when $\lambda(t|H_t)$ is greater than 1, the z_j values accumulate more rapidly and represent a stretching of time, so that nearby spikes are drawn further apart.

Because the transformation in Equation 9 is one-to-one, any statistical assessment that measures the agreement between the z_j values and an exponential distribution directly evaluates how well the original model agrees with the spike train data. A Kolmogorov–Smirnov (KS) plot is a plot of the empirical cumulative distribution function (CDF) of the rescaled z_j's against an exponential CDF. If the conditional intensity model accurately describes the observed spiking data, then the empirical and model CDFs should roughly coincide, and the KS plot should follow a 45° line. If the conditional intensity model fails to account for some aspect of the spiking behavior, then that lack of fit will be reflected in the KS plot as a significant deviation from the 45° line. Confidence bounds for the degree of agreement between a model and the data may be constructed using the distribution of the Kolmogorov–Smirnov statistic (Johnson and Kotz, 1970).

Another approach to measuring agreement between the model and data is to construct a quantile–quantile (Q-Q) plot, which plots the quantiles of the rescaled ISIs against those of exponential distribution (Barbieri et al., 2001; Brown et al., 2002). Q-Q plots are useful for visualizing which quantiles of the rescaled data are well captured and which are poorly captured by the model. As in the case of KS plots, exact agreement occurs between the point process model and the experimental data if the points lie along a 45° line.

If the model is correct, the z_j values should be not only uniformly distributed, but also independent. Thus, even when the KS statistic is small, it is still important to show that the rescaled times do not contain significant dependence structure. One way to assess independence up to second-order temporal correlations is to compute the autocorrelation function of the transformed rescaled times. If the autocorrelation at any lag is significant, it suggests that the proposed conditional intensity model does not fully capture the structure in the data.

Example 2 Continued: KS and Q-Q plots of a hippocampal neuron

We can construct KS and Q-Q plots for the transformed place cell data using the model fit shown in Figure 2. In this case, the KS plot, shown in Figure 3A,

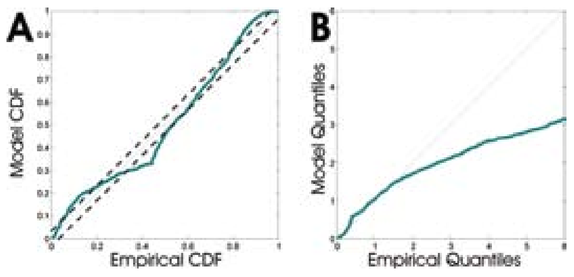


Figure 3. *A*, KS plot and *B*, Q-Q plot for distribution of rescaled intervals for the place field model shown in Figure 2.

deviates significantly from the 45° line at multiple locations. By examining the Q-Q plot in Figure 3*B*, we see that the model results in too few small rescaled ISIs, too many midrange ISIs, and too few large ISIs. This suggests that this inhomogeneous Poisson model for the spiking activity is unable to completely describe the structure in the data. It is likely that a similar model that incorporated spike history would provide a much closer fit to the data.

Conclusions

Point process methods are vital for understanding how neurons represent information about external biological and behavioral signals. In particular, point process theory provides approaches for visualizing spike train data, constructing and fitting neural receptive field models, and assessing the ability of these models to explain structure in the data.

At the heart of these methods is the conditional intensity function, which provides a unified mathematical framework for analyzing point process data. The conditional intensity implicitly defines the instantaneous probability of spiking as a function of past firing history and external covariates. A neural model can be constructed by writing down an equation for the conditional intensity in terms of these covariates and a set of model parameters. From the definition of the conditional intensity, it is easy to show that the likelihood function of any point process model of a neural spike train then has a canonical form given by Equation 5. For any observed spike train, maximum likelihood methods can then be applied to find the model parameters that best fit the data and to compute uncertainty about those parameter estimates. The likelihood framework therefore provides an efficient way to extract information from a neural spike train. Likelihood methods are some of the most widely used paradigms in statistical modeling owing to their numerous optimality properties and extensive theoretical underpinnings.

The conditional intensity function was also fundamental for constructing goodness-of-fit tests, many of which are based on the time-rescaling theorem. Assessing goodness-of-fit is a crucial, often overlooked step in neuroscience data analyses. This assessment is essential for establishing what features of data a model does and does not describe, determining whether the model is appropriate for making statistical inferences about the neural system being studied, and establishing how reliable those inferences maybe.

Although we have focused here on analyses of single neural spike train time-series, the methods can be extended to analyses of multiple, simultaneously recorded neural spike trains (Chornoboy et al., 1988; Okatan et al., 2005). These methods are becoming increasingly important since recent technology allows us to record simultaneous spike trains from large neural ensembles.

References

Barbieri R, Quirk MC, Frank LM, Wilson MA, Brown EN (2001) Construction and analysis of non-Poisson stimulus response models of neural spike train activity. J Neurosci Methods 105:25-37.

Brillinger DR (1988) Maximum likelihood analysis of spike trains of interacting nerve cells. Biol Cybern 59:189-200.

Brown EN, Barbieri R, Ventura V, Kass RE, Frank LM (2002) The time-rescaling theorem and its application to neural spike train data analysis. Neural Comp 14:325-346.

Brown EN, Barbieri R, Eden UT, Frank LM (2003) Likelihood methods for neural data analysis. In: Feng J, ed. Computational neuroscience: a comprehensive approach. Chap 9, pp 253-286. London: CRC Press.

Brown EN, Kass RE, Mitra PP (2004) Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci 7:456-461.

Buzsaki G, Eidelberg E (1983) Phase relations of hippocampal projection cells and interneurons to theta activity in the urethane anesthetized rat. Brain Res 266:334-338.

Buzsaki G. (1989) Two-stage model of memory trace formation: a role for "noisy" brain states. Neuroscience 31:551-570.

Chornoboy ES, Schramm LP, Karr AF (1988) Maximum likelihood identification of neural point process systems. Biol Cybern 59:265-275.

Daley DJ, Vere-Jones D (2003) An introduction to the theory of point processes, 2nd ed. New York, NY: Springer.

NOTES

Johnson A, Kotz S (1970) Distributions in statistics: Continuous univariate distributions. New York, NY: Wiley.

Kass RE, Ventura V, Brown EN (2005) Statistical issues in the analysis of neuronal data. J Neurophys 94:8-25.

Ogata Y (1988) Statistical models for earthquake occurrences and residual analysis for point processes. J Am Stat Assoc 83:9-27.

Okatan M, Wilson MA, Brown EN (2005) Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. Neural Comp 17:1927-1961.

O'Keefe J, Dostrovsky J (1971) The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. Brain Res 34:171-175.

Papangelou F (1972) Integrability of expected increments of point processes and a related random change of scale. Trans Amer Math Soc 165:483-506.

Snyder DL, Miller MI (1991) Random point processes in time and space. New York, NY: Springer-Verlag.

Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN (2005) A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. J Neurophysiol 93:1074-1089.

Quantifying Animal Behavior

Ofer Tchernichovski, PhD and Sigar Saar, PhD

Department of Biology, The City College of New York
New York, New York

NOTES

About Ethology

In the field of systems neuroscience, measuring behavior is often the first step to take when studying how the brain operates. One measurement approach involves using a well-established behavioral paradigm, such as training a monkey to make a saccadic eye movement toward a target, or training a rat to push a lever to obtain a food reward. In such experiments, the behavior of the trained animal enables the researchers to test hypotheses regarding the associated neuronal activity. Note that, quite often, what we call "simple behavior" is not simple at all; rather, we are imposing simplicity (by restraining the animal) on the situation or recording the outcome of complex movements.

A second approach centers on studying patterns of free, naturally occurring behaviors but without making any attempt to simplify them. The researchers then try to associate those behaviors with brain mechanisms. Examples of such behaviors are rat exploratory movements in local space and developmental song learning in birds. The measurement, analysis, and interpretation of such behaviors are challenging but can potentially lead to new hypotheses: namely, those pertaining to the discovery of mechanisms we could not have imagined before examining these patterns. Ethology is the science of studying the structure of natural behavior, that is, what happens when the animal does what it is designed to do. In *neuroethology*, we attempt to associate those behavioral patterns with underlying structures and dynamics within the brain.

Building Blocks of Behavior

For more than a century, investigators have been looking for an appropriate approach to studying behavior at the molecular level. The question remains, can we break down behaviors into units that relate animal movements to distinct neurophysiological mechanisms? For example, phototaxis (movement toward a light source) can be observed in plants and primitive animals. Such a phenomenon can be explained by a servomechanism. In much the same way that a self-guided missile automatically pursues an airplane by following the heat emitted by its engine, plant and animal organisms are equipped with sensors and differences in light intensity across sensors are used to compute an error signal, which is then used to make appropriate steering movements that minimize the error and accurately follow the target. Carefully measuring the light input and the steering movements needed might allow us to infer something about the servomechanism and to assess the delay, the gain, and the complexity of the error signal. Of course, without some knowledge of those features, it would be difficult to interpret the attendant neuronal activity.

Reflexes can also be thought as servomechanisms, except that the reflex ring (from the sensor to the spinal cord and back to the muscle) monitors and controls an internal state, such as the length of a muscle. The interest in reflexes as units of automated behavior led Sir Charles Sherrington (about one hundred years ago) to perform the first quantitative measurements of behavior. In essence, all Sherrington did was to connect a thread and a pen to the leg of a spinal frog so as to record traces of its reflexive movements on a rolling paper. But looking at those traces uncovered the foundation of modern neuroscience and won Sherrington a Nobel Prize in Physiology or Medicine in 1932. The traces showed that reflexes have a stereotyped threshold, accurate latencies, and afterdischarges. The interactions among neurons could be explained as, among other features, summation over space, summation over time, facilitation, inhibition, and chaining. Further, analysis of latencies allowed Sherrington to infer the existence of synapses (a term that he coined) between neurons.

Like a Newtonian equation (which might be practical for an engineer but inappropriate in a more general context), Sherrington's findings on reflexes have practical implications but are not appropriate building blocks for studying behavior in general (Glimcher, 2003). During the middle of the twentieth century, a new scientific field emerged: ethology. Its principal founder, Konrad Lorenz, was a zoologist who believed that animal behavior is a legitimate component of biological structure; to wit, a monkey possesses monkey genes, monkey bones, and uniquely monkey behavior. Just as investigating the structure of genes and skeletons can help us understand function in a comparable (evolutionary) context, so too can behavior be investigated as an extension of other fields of biology, e.g., comparative anatomy.

According to Lorenz, the units for quantification are the instincts: innate behaviors, with well-defined core structure, among which he hoped to find homologies across species. To a large extent, this attempt failed (Golani, 1992). Nevertheless, the framework of ethology—putting efforts into describing behavioral structures in an "appropriate manner" (as vague as this might sound)—has succeeded in several areas of neuroscience, leading to the emergence of the specialty of neuroethology. This chapter will focus on one of those neuroethology fields: developmental song learning in birds.

NOTES

Informatics and Quantifying Behavior

The dramatic increase in computational power and the decrease in data storage costs have made it easy to compile and manage large sets of behavioral data. For example, until the 1990s, all the birdsong learning literature was based on sparse sampling (say, a few minutes of singing recorded weekly). We can now record an entire developmental vocal repertoire for each animal, namely by looking at every song syllable produced by the bird during its lifetime with no sampling restrictions (approximately 1-2 million sounds per bird). Because storage costs are so low (~$0.15/Gb), it makes perfect sense to record behavior continuously, as long as the animal is in the recording environment. Of course, one must think carefully about how to organize the data, maintain the databases, and develop approaches to easily access and analyze arbitrary chunks of information. However, this is what informatics facilitates, and there are sufficient off-the-shelf methods for accomplishing this (Tchernichovski et al., 2004).

Returning to the issue of building blocks of behavior, by using appropriate databases, one can easily examine behavioral patterns across multiple time scales, attempting to recognize patterns that cannot be seen within smaller samples. Using birdsong development as a case study, I will present a short overview of such approaches, starting from extracting features, to detecting production categories at different time scales, and then following those over the course of developmental trajectories.

In contrast to obtaining behavioral data, obtaining neuronal data is difficult, expensive, and in most cases, much more limited in scope (e.g., in duration of recording, number of neurons recorded from). One bit of advice I always give my collaborators is to try collecting behavioral data over an extended period. Even if one can record electrophysiology for only a few days, it is still useful for tracking the entire song development of that bird. Later on, such behavioral records might become essential to an accurate interpretation of the neuronal data. As in diagnosis in medicine, the history of the case (here, song development) is often the most important clue to understanding what is going on right now within the birdsong structure.

Overview of Song Behavior

In general, birds sing to attract mates and defend their territories (Catchpole and Slater, 1995). Of course, the males of many other animal species put effort into attracting mates and defending their territories using other means, e.g., by marking their territory's borders. So is there a qualitative difference between birdsong and other means of defending territories and attracting mates? We do not know the answer, but most people agree that there is something very special about birdsongs: Songs are learned, and birds have local "cultures" of singing in a certain way—local dialects of a sort. Further, birdsongs are often esthetically pleasing, even musical (Rothenberg, 2005). One motivation for assessing the qualities of birdsong is in order to provide a rationale for their beauty.

On a more prosaic level, birdsongs fit well into the concept of instinct even though they carry a strong learning component. Songs can be triggered by an appropriate key stimulus (e.g., presenting a female) and yet are internally driven. For instance, isolate male zebra finches spend hours every day singing (a so-called vacuum activity, according to Lorenz). In addition, birdsongs are highly structured and stereotyped. Although songs are learned, song acquisition by itself could be considered instinctive because, in many species, the bird is "programmed" to learn its song only during a narrow, sensitive period of its development. The narrowest form of "programmed" learning is called *imprinting*, when chicks learn to recognize their mother during their first day of life and acquire some notion of their species' social and sexual identity.

In the context of system neuroscience, studying birdsong has become popular because it is a good model for developmental learning in general, and speech development in particular. Its main advantages are ease of measuring singing behavior and song development (over weeks, as opposed to years in humans), an animal subject with a compact brain, and distinct song nuclei that can be recorded while the bird is vocalizing.

Analysis of song behavior across multiple time scales

Developmental learning is difficult to study because it occurs over multiple time scales: e.g., during morning singing, then sparsely during the day, and even during sleep (Dave and Margoliash, 2000; Deregnaucourt et al., 2005). It takes a few weeks of "practice" before the bird is fully capable of imitation. How can we measure behavior so as to bridge these time spans?

The song of an adult zebra finch is composed of a few distinct syllable types repeated in fixed order. Figure 1 presents a sonogram-like display of a song. We performed multitaper spectral analysis and computed directional spectral derivatives. These provide us with a similar image to that of the spectrogram but of superior quality, because they optimize the detection of frequency traces (Tchernichovski et al., 2000). As shown in the example, it is easy to detect a stereotyped song structure with repeating syllable types (marked in colors) in each burst of song.

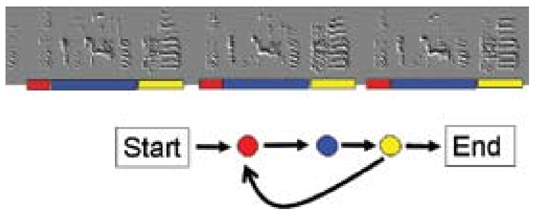


Figure 1. Spectral images derived from a zebra finch song motif. Red bars indicate syllable boundaries.

As noted earlier, this highly stereotyped birdsong was imitated from an adult tutor (in this case, from playback recordings) when the bird was young. Because we recorded the entire song-learning history of this bird, we can now examine how this song came about. One of the simplest approaches is to look at the entire song development as an image, as if viewing a sonogram.

Figure 2 presents two sound spectrograms: On the left panel, we see a male's long call, with a duration of about 100 ms. On the right panel, we see a very different spectrogram (Saar and Mitra, 2008). Using a slow-varying feature, such as the amplitude of the songs, we performed spectral analysis over long time windows (hours instead of milliseconds). We did so in order to examine song structure across lower frequencies and over the developmental time scale. These low frequencies happen to correspond to the song's rhythm, whereas in regular sonograms, frequencies reveal pitch.

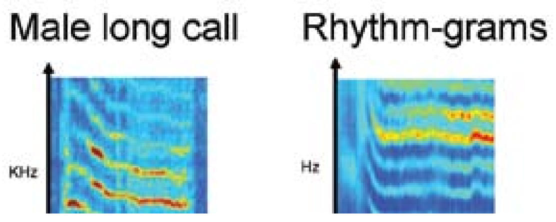


Figure 2. Left panel: sonogram of a male zebra finch call. Right panel: rhythm-gram of song development.

Just as the sonogram's left panel reveals frequency downmodulation during a call (over a course of about 50 ms), the rhythm-gram at right reveals rhythm downmodulation, which occurred approximately at days 50-55 post-hatching. We can now take a step back and look more carefully at what happened during day, when the rhythm frequency was decreasing. Here we can see how the bird has "time-warped" a syllable and increased the motif's duration by about 100 ms.

Frame-based and segment-based analysis

It is often useful to extract features from the raw behavioral signal. As shown in the previous section, we can measure rhythm using slow-varying features such as amplitude envelope. Standard methods exist for analyzing and compressing signals, e.g., spectral analysis and principal components analysis. However, whenever one has an opportunity to extract biologically relevant features from the signal, it is worth a try. Much of our progress in analyzing song development stemmed from using spectral analysis as a first step and extracting meaningful features as a second step (Tchernichovski et al., 2000). For birdsongs, we calculate features that are thought to correlate with articulation; these include pitch, frequency modulation (FM), and Wiener entropy (estimating the width of the power spectrum). These features are calculated continuously during 10-ms time windows and in 1-ms steps. Information about these features is saved to a database, and we then segment the data and calculate first- and second-order statistics of those features in each syllable, such as mean pitch and mean FM.

Analysis of song feature distribution

One advantage of having large data sets is that we can base our investigation on large-scale distribution of features. For example, Figure 3 presents developmental histograms of song features. Starting from the duration histogram (left panel), each peak represents a syllable type (syllable FM, middle panel; syllable entropy, right panel), the development of which the ridges show us.

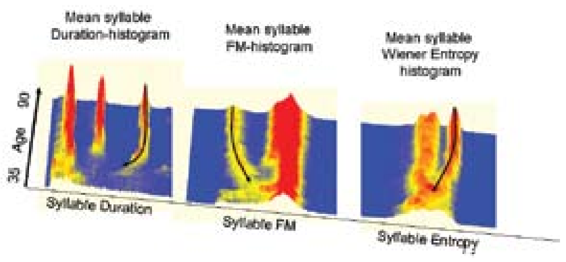


Figure 3. Developmental histograms of syllable features. Each row corresponds to a day. Histograms are based on all sounds produced in a given day. *y*-axis represents days old.

One interesting observation is that we can see no ridges during early song development (and before training the bird with song playbacks). Instead, syllable types emerge during song development, and the large amount of data makes it easy to see when and how the graded signal becomes clustered.

Looking at developmental histograms of different features (e.g., FM, Wiener entropy) also reveals different dynamics. Nonetheless, all features and all birdsongs we have analyzed so far share common characteristics: Ridges tend to move away from each other during development and sometime split (but not merge). We call this effect "differentiation of syllables."

Summary

We started our investigation by asking about units of behavior, and we ended up with descriptive models that present simple feature distributions. By looking at such distribution, we can detect significant events such as the emergence of syllable types followed by differentiation, or smooth downmodulation of rhythm frequencies when the bird "stretches out" its emerging motif. To some extent, a good and simple descriptive model can provide answers similar to those Lorenz envisioned. Such a description of animal behavior naturally lends itself to a parallel investigation at the speech-forming and neurological levels.

References

Catchpole CK, Slater PJB (1995) Bird song—biological themes and variations, Cambridge, UK: Cambridge UP.

Dave AS, Margoliash D (2000) Song replay during sleep and computational rules for sensorimotor vocal learning. Science 290:812-816.

Deregnaucourt S, Mitra PP, Feher O, Pytte C, Tchernichovski O (2005) How sleep affects the developmental learning of bird song. Nature 433:710-716.

Glimcher PW (2003) Decisions, uncertainty, and the brain. The science of neuroeconomics. Cambridge, MA: The MIT Press.

Golani I (1992) A mobility gradient in the organization of vertebrate movement: the perception of movement through symbolic language. Behav Brain Sci 15:249-308.

Rothenberg D (2005) Why birds sing: a journey through the mystery of bird song. Norwich, VT: Terra Nova Press.

Saar S, Mitra PP (2008) A technique for characterizing the development of rhythms in bird song. PLoS ONE 3:e1461.

Tchernichovski O, Nottebohm F, Ho CE, Bijan P, Mitra PP (2000) A procedure for an automated measurement of song similarity. Anim Behav 59:1167-1176.

Tchernichovski O, Lints T, Deregnaucourt S, Mitra PP (2004) Analysis of the entire song development: methods and rationale. Ann N Y Acad Sci 1016:348-363.

Optical Imaging Analysis for Neural Signal Processing: A Tutorial

Andrew T. Sornborger, PhD

Department of Mathematics and Faculty of Engineering, University of Georgia
Athens, Georgia

Tutorial: Denoising and Frequency Localization of an Imaging Time Series

In this tutorial, we will cover the basic steps to follow when using a singular value decomposition (SVD) and Thomson F-test in order to reduce the amount of noise in a neural imaging data set (Mitra and Bokil, 2008). Figures 1 through 5 depict the steps involved.

It should be remembered that, in this type of analysis, an image is considered to be primarily a vector of values (Fig. 1, top). That is, the pixels that were originally arranged in two dimensions are arranged as elements of a vector, and the two-dimensionality is, for the moment, not depicted. A movie of imaging data taken over time is therefore described as a data matrix in which each row is an image vector, and time progresses row by row from the initial image to the final image (Fig. 1, bottom).

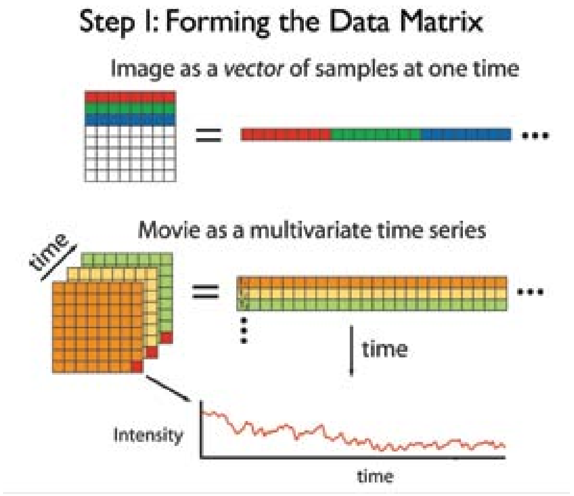

Step I: Forming the Data Matrix
Image as a vector of samples at one time
Movie as a multivariate time series
time
time
Intensity
time

Figure 1. The first step in the analysis of optical imaging data is to list the pixels in the form of a vector (top panel). Then the image vectors are arranged sequentially in time, thereby forming a data matrix.

Therefore, if you are working with imaging data saved in a MATLAB.mat file, for example, in the variable DAT, where DAT has elements DAT (i,j,k) (with i = 1...nt, j = 1...nx and k = 1...ny), then with the command

```
X = reshape(DAT, [nt nx*ny]);
```

you can make a data matrix, X, out of your imaging data. This data matrix has nt rows and $nx \times ny$ columns.

When performing an analysis for the first time, it is often useful (and prudent) to plot the data matrix. To do this, first subtract the mean of each pixel time course (each column) from each time course and make a pseudo-color plot of the data. Much of the structure of the data (including possible measurement artifacts) may be seen in such a plot. Often, more of the dynamics may be seen in this way than by viewing a movie of the raw imaging data. We can do this for our data matrix, X, using the following commands:

```
% subtract mean pixel value from each pixel
X = X – repmat(mean(X, 1), [nt 1]);
% make pseudo-color plot of matrix
pcolor(X); shading flat;
```

Raw imaging data are typically noisy. Sources of noise can be either shot noise from fluctuations in the number of photons arriving at the detector or dark noise arising from thermal fluctuations in the measurement apparatus. Patterned measurement artifacts from the imaging device and changes in the physiology of the system being imaged can often be significant sources of noise as well.

The judicious use of an SVD can help reduce such noise in imaging data sets. An SVD may be thought of in many ways, one useful conceptualization being that the SVD takes the raw data matrix (Fig. 2, top row) and decomposes it into a sum of many submatrices (Fig. 2, bottom row). Each of the submatrices is the combination of an eigenvector and a time course. In turn, each row of the submatrix is proportional to the pixel values of the eigenvector, and each column

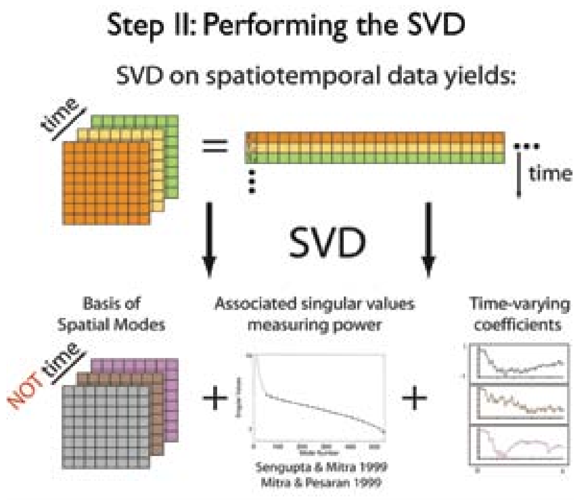

Step II: Performing the SVD
SVD on spatiotemporal data yields:
time
time
SVD
Basis of Spatial Modes
Associated singular values measuring power
Time-varying coefficients
NOT time
Sengupta & Mitra 1999
Mitra & Pesaran 1999

Figure 2. An SVD is performed on the data matrix, re-sulting in a set of eigenimages, singular values, and time-varying coefficients.

of the submatrix is proportional to the time course. The overall weight of the submatrix formed in this way is given by the singular value associated with the eigenvector and the time course.

MATLAB has a routine for performing an SVD on a data matrix. To perform an SVD on our data matrix, we type

```
[u, s, v] = svd(X, 0);
```

The matrices u, s, and v that this routine outputs are the left eigenvectors (these are time-varying coefficients), the singular values, and the right eigenvectors (often called eigenimages), respectively, of the matrix X.

If you add up all the submatrices, the result is identically equal to the original data matrix. Sometimes, however, some of the submatrices may be identified as contributing only noise. In this case, they may be deleted, and the resulting reconstruction of the original data matrix will be less noisy. One commonly used method for choosing eigenvectors and time courses that are not part of the noise is to note the location of a pronounced "knee" in a plot of the singular values. This knee is used to identify a threshold beyond which eigenvectors are discarded.

To visualize the singular values, type the command

```
% plot singular values on
semilogarithmic plot
semilogy(diag(s));
```

To reconstruct the data matrix using only the first M eigenvectors, type

```
% reconstructed data matrix using only M
eigenvectors
Xrecon = u(:,1:M) * s(1:M,1:M) * v(:,1:M)';
```

You can use several different values for M and use a pseudo-color plot to compare the reconstruction with the original data.

The denoising process is depicted in Figure 3. First, the data matrix is formed (Fig. 3, top row). Next, an SVD is performed on the data (Fig. 3, middle row). Finally, the results from the SVD are examined and noisy eigenvectors discarded (Fig. 3, middle row; note the singular values to the left of the knee inside the red circle). The data matrix is then reconstructed by reversing the SVD decomposition (adding up all the nondiscarded submatrices), resulting in a denoised data set. Since it is hoped that only random noise was discarded in this process, SVD may also be viewed as a data compression method. The retained eigenvectors and their time courses may be thought of as a low-dimensional summary of the data set.

The above method is a useful first step for denoising the neural imaging data. However, a number of possible problems can arise. One major problem is that some of the eigenvectors before the knee may be noisy. The reason for this phenomenon lies in the fact that the SVD is a blind method for analyzing data: The eigenvectors represent covarying information in the data set and nothing more, and the SVD does not contain any model of the signal. Therefore, for instance, patterned noise from the imaging apparatus (in which many pixels have covarying time courses) is likely to be depicted in the first few eigenvectors. Another problem is that small sources of covariance (possibly related to a signal of interest) may be discarded along with the eigenvectors below the knee.

One method for obtaining improved reconstructions of neural imaging data is to incorporate periodicity into the experimental design by using a periodic stimulus (Kalatsky and Stryker, 2003; Sornborger et al., 2005; Xu et al. 2008). By doing this, we are putting an extra "hook" in the data that can be used later to more accurately identify the information that is truly relevant to our experiment.

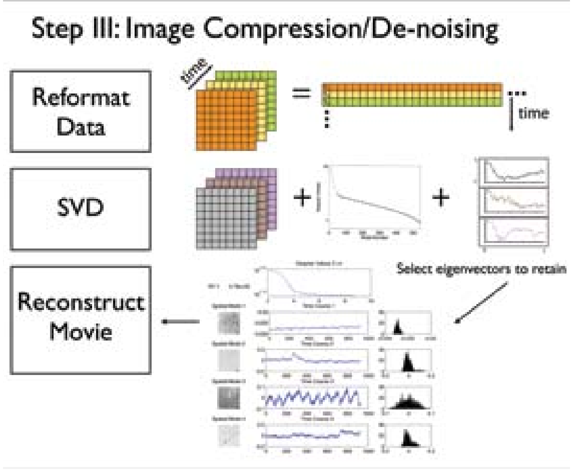

Step III: Image Compression/De-noising
Reformat Data
time
time
SVD
Reconstruct Movie
Select eigenvectors to retain

Figure 3. The data matrix may be reconstructed using only a subset of the eigenimages, their associated singular values, and time-varying coefficients. This reconstruction procedure can significantly reduce the noise in the data matrix.

In Figure 4, we show an SVD analysis of calcium imaging data taken of neural tissue that was stimulated periodically. Instead of the standard visualization in time (Fig. 4, top row, left panel), the frequency spectrum of each left eigenvector (time-varying coefficient) was estimated using Thomson's multitaper spectral estimation method (Thomson, 1982) (top row, top right panel). Using Thomson's F-test, harmonic peaks in the spectrum at the stimulation frequency were detected and then estimated and extracted from the left eigenvectors (top row, bottom right panel). This process was performed for all left eigenvectors.

To calculate a multitaper estimate of the spectrum of the first 10 left eigenvectors and to visualize the information, we use the Chronux function "mtspectrumc" and a pseudo-color plot of the data:

```
% calculate frequency spectrum for a
left eigenvector
[S, f] = mtspectrum(u(:,1:10), params);
% plot the spectrum
pcolor([1:10], f, 10*log10(S)); shading flat;
```

To perform an F-test on the first 10 left eigenvectors, we use the Chronux function f-test:

```
% perform f-test on a left eigenvector of X
at frequency
% set sampling frequency parameter
params.Fs = 10;
% perform f-test
[Fval, A, fqs, sig, sd] = ftestc(u(:,1:10),
params);
```

This Chronux code outputs *Fval*, the value of the F-statistic for each frequency and channel; *A*, the complex line amplitude for each frequency and channel; *fqs*, the frequencies that the F-test was evaluated at; *sig*, the significance level of the test; and *sd*, the standard deviation of the amplitude.

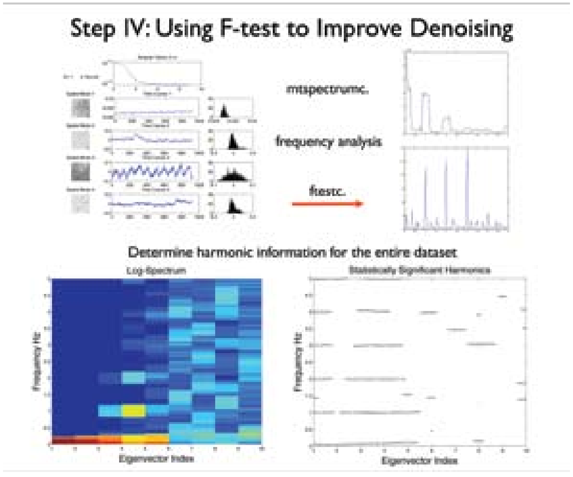


Figure 4. When a periodic stimulus/response experimental paradigm has been used, Thomson's spectral analysis may be used to visualize periodicity in the data set, and Thomson's F-test may be used to detect and estimate harmonics in the data.

A contour plot of the F-statistics across all frequencies for the first 10 left eigenvectors may be made with the following command:

```
% plot f-statistic for first 10 left
eigenvectors using contour plot
contour(Fval, [sig sig], 'k');
```

The frequencies at which significant harmonics were detected in the first left eigenvector may be output with the following command:

```
% frequencies of significant harmonics
fqs(find(Fval(:,1) > sig))
```

And the complex amplitudes of the significant harmonics are as follows:

```
% amplitudes of significant harmonics
A(find(Fval(:,1) > sig), 1)
```

Using the amplitude and frequency information from all the harmonics, we can reconstruct the statistically significant periodic part of the time series for the first left eigenvector:

```
% reconstruct periodic part
amps = A(find(Fval(:,1) > sig, 1);
fs = fqs(find(Fval(:,1) > sig));
sg = zeros(size(u(:,1)));
for I = 1:nh
 sg = sg + real(amps(i) * sin(2*pi*fs(i)*[1:nt]/
params.Fs) …
          + amps(i)' * cos(2*pi*fs(i)*[1:nt]/
params.Fs);
end;
```

In general, we would loop through all the left eigenvectors in order to obtain all the periodic components in the data. This yields the periodic components for each of the first 10 left eigenvectors, *sg*(*i*,*j*), where *i* = 1…nt and *j* = 1…10. Note that we could have investigated more than just the first 10 left eigenvectors; this is just an example. To reconstruct the periodic part of the entire data matrix, we simply put the pieces back together:

```
Xperiodrecon = sg(:,1:10) * s(1:10,1:10) * v(:,1:10)';
```

The results of the spectral and harmonic analyses of the first 10 time courses are depicted in Figure 4, bottom row, left panel. This panel shows a pseudo-color plot of the log-spectra of all time-varying coefficients plotted side by side. Note the bright yellow peaks in the third, fourth, and fifth time courses. In Figure 4 (bottom row, right panel), we plot contours of the harmonic peaks that were found across all the time-varying coefficients. Note the contours corresponding to the spectral peaks in the left panel.

NOTES

Figure 5 depicts the complete analysis process. After construction of the data matrix, we perform the SVD. Then, using the F-test, we extract statistically significant periodic responses in the data. Next, only the sinusoids with frequencies that are at multiples of the stimulus frequency are extracted. A reconstructed data set is then made of just the periodic response.

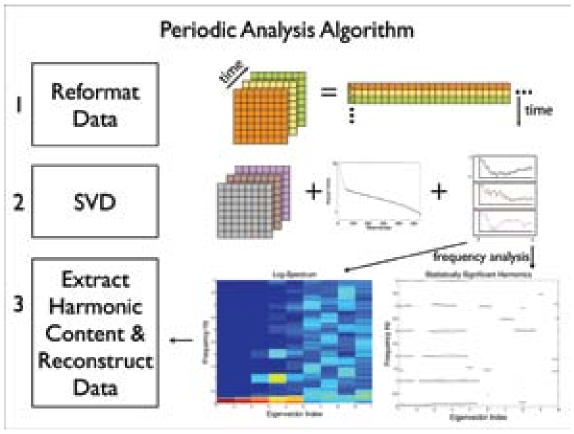

Periodic Analysis Algorithm
1
Reformat Data
time
time
2
SVD
frequency analysis
3
Extract Harmonic Content & Reconstruct Data

Figure 5. By combining statistically significant harmonics information with the associated eigenimages, a new data matrix may be reconstructed that represents only the periodic response to the stimulus.

Conclusion

Denoising data using the above methods can be extremely useful for improving the signal-to-noise ratio of neural imaging data. We have discovered that data features that would otherwise be missed can often be found using these methods. In the data set used in the above example, clear evidence was found of functional neuronal projections from one neuron to another within the neural tissue that was imaged. Without applying our methods, this functional information would not have been detectable.

References

Kalatsky VA, Stryker MP (2003) New paradigm for optical imaging: temporally encoded maps of intrinsic signal. Neuron 38:529-545.

Mitra PP, Bokil H (2008) Observed brain dynamics. Oxford, UK: Oxford UP.

Sengupta AM, Mitra PP (1999) Distributions of singular values for some random matrices. Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics 60:3389-3392.

Sornborger A, Yokoo T, Delorme A, Sailstad C, Sirovich L (2005) Extraction of the average and differential dynamical response in stimulus-locked experimental data. J Neurosci Methods 141:223-229.

Thomson, DJ (1982) Spectrum estimation and harmonic analysis. Proc IEEE 70:1055-1096.

Xu J, Sornborger AT, Lee JK, Shen P (2008) Drosophila TRPA channel modulates sugar-stimulated neural excitation, avoidance and social response. Nat Neurosci 11:676-682.

Neural Signal Processing: Tutorial 1

Keith P. Purpura, PhD[1] and Hemant Bokil, PhD[2]

[1]Department of Neurology and Neuroscience, Weill Cornell Medical College
New York, New York

[2]Cold Spring Harbor Laboratory
Cold Spring Harbor, New York

Introduction

In this chapter, we will work through a number of examples of analysis that are inspired in part by a few of the problems introduced in "Spectral Analysis for Neural Signals." Our purpose here is to introduce and demonstrate ways to apply the Chronux toolbox to these problems. The methods presented here exemplify both univariate analysis (techniques restricted to signals elaborated over a single time course) and bivariate analysis, in which the goal is to investigate relationships between two time series. Problems involving more than two time series, or a time series combined with functions of spatial coordinates, are problems for multivariate analysis. The chapters "Multivariate Neural Data Sets: Image Time Series, Allen Brain Atlas" and "Optical Imaging Analysis for Neural Signal Processing: A Tutorial" deal explicitly with these techniques and the use of the Chronux toolbox to solve these problems.

"Spectral Analysis for Neural Signals" introduces the spectral analysis of single-unit recordings (spikes) and continuous processes, for example, local field potentials (LFPs). As shown in that chapter, the multitaper approach allows the researcher to compute and render graphically several descriptions of the dynamics present in most electrophysiological data. Neural activity from the lateral intraparietal area (LIP) of the alert monkey was used to calculate LFP spectrograms and spike-LFP coherograms and, most importantly, measures of the reliability of these estimates. Although the computations required to produce these descriptions are easily attainable using today's technology, the steps required to achieve meaningful and reliable estimates of neural dynamics need to be carefully orchestrated. Chronux provides a comprehensive set of tools that organize these steps with a set of succinct and transparent MATLAB scripts. Chronux analysis software also clears up much of the confusion surrounding which of the parameters that control these calculations are crucial, and what values these parameters should take, given the nature of the data and the goals of the analysis.

Chronux is downloadable from www.chronux.org. This software package can process both univariate and multivariate time series data, and these signals can be either continuous (e.g., LFP) or point process data (e.g., spikes). Chronux can handle a number of signal modalities, including electrophysiological and optical recording data. The Chronux release includes a spike-sorting toolbox and extensive online and within-MATLAB help documentation. Chronux also includes scripts that can translate files generated by NeuroExplorer (Nex Technologies, Littleton, MA) (.NEX), and the time-stamped (.PLX) and streamed (.DDT) data records collected with Plexon (Dallas, TX) equipment.

We will proceed through components of a standard electrophysiology analysis protocol in order to illustrate some of the tools available in Chronux. Figure 1 charts the basic steps required for handling most electrophysiological data. We will assume that an editing procedure has been used to sort the data into continuous signals (LFPs or EEGs) and spikes. We will also advance to a stage that follows both the spike sorting and data conditioning steps (detrending and removing artifacts, including 60 Hz line noise). We will return to the detrending and 60 Hz line noise problems later in the chapter.

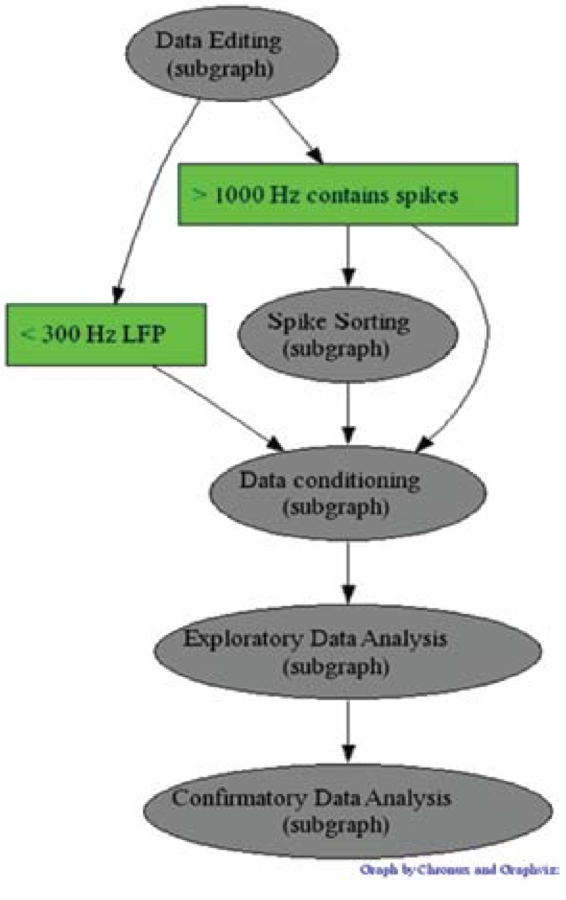

Data Editing (subgraph)
> 1000 Hz contains spikes
< 300 Hz LFP
Spike Sorting (subgraph)
Data conditioning (subgraph)
Exploratory Data Analysis (subgraph)
Confirmatory Data Analysis (subgraph)
Graph by Chronux and Graphviz

Figure 1. Electrophysiological data analysis protocol.

Typically, a first step to take in exploratory data analysis is to construct a summary of neural activity that is aligned with the appearance of a stimulus or some behaviorally relevant event. Recall that in the example described in "Spectral Analysis for Neural Signals," the monkey is challenged with a delay period during which it must remember the location of a visual target that was cued at the beginning of a trial. Each

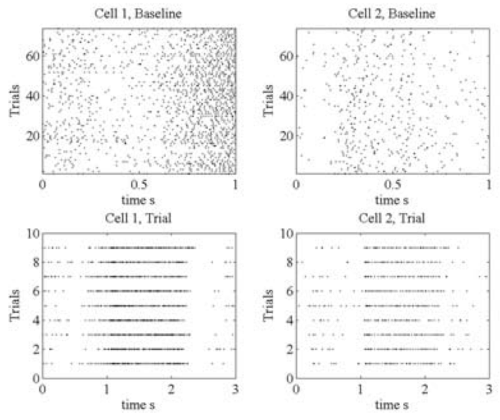


Figure 2. Raster plots for 2 isolated spikes recorded in monkey LIP. Each dot represents the time of occurrence of a spike. Each row details the spike times from a different trial in the experiment. Top row: baseline period (first second of each trial) for all trials. Bottom row: complete trials, including the baseline period (0-1 s) and delay and response periods (1-3 s) for a subset of trials associated with a single target.

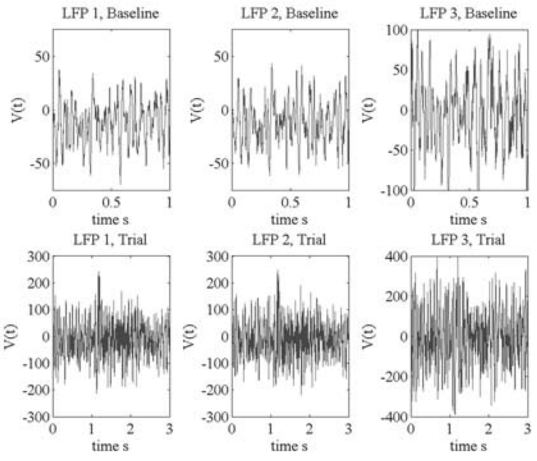


Figure 3. Local field potentials (LFPs) recorded concomitantly with the spikes shown in Fig. 1. The LFPs are averaged over the trials elaborated in the spike raster plots in Fig. 1. Top row: baseline period; bottom row: complete trials. Voltage values for the LFPs are in units of microvolts.

3 s trial is composed of a 1 s baseline period followed by a 2 s period containing the delay and response periods. The neural signals, contained in the tutorial data file *DynNeuroLIP.mat*, include three LFP signals and two point-process time series (i.e., two channels of spike times). Nine trials associated with one target direction are included, as are 72 trials of the baseline period combined across eight possible target positions. We will first produce an estimate of the firing rate associated with the delay period and then proceed to look for interactions between the two spikes and for any temporal structure in the LFPs.

A script that will take us through these steps can be launched by typing

```
>> lip_master_script
```

at the command prompt. The first figure generated by the script (Fig. 2) details the spike times (as raster plots) of the two single units for all the baseline periods (top row) and for the subset of trials (bottom row) associated with one particular target. Note that the number of spikes increases roughly 1 s after the start of the trial. This increase indicates that something is indeed happening at the start of the delay period and suggests that these neurons may play a role in establishing a working memory trace of one target's location. The tutorial script will also produce results as in Figure 3, where we see the three LFP signals that were recorded alongside the spikes. These signals likewise demonstrate a change in activity at the start of the delay period. As we proceed through the tutorial, we will see how Chronux can be used to further characterize the neural activity in these recordings.

Regression

Figure 4 illustrates one characterization that is often used for depicting spike data. The top subplot of the figure illustrates a standard frequency histogram, using a bin size of 104 ms, for a single trial of spike response. The rate is calculated by dividing the spike count in each bin by the bin width. The bottom subplot of the figure shows a smooth estimate of the firing rate generated by applying a local regression algorithm, *locfit*. In order to plot the regression fit and produce 95% confidence bounds for the rate

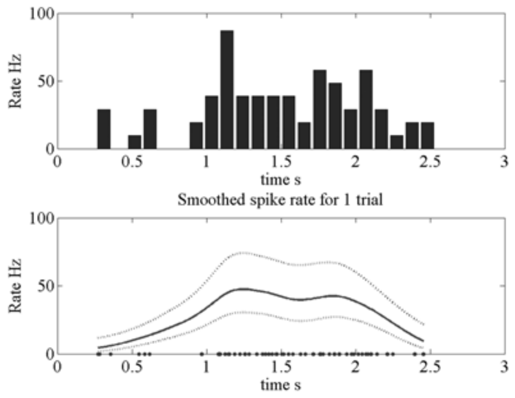


Figure 4. Spike rate estimates. Top row: frequency histogram constructed from a single trial for one isolated spike. Bin size = 104 ms. Bottom row: output from Chronux script ***locfit***. The solid line depicts an estimate of the spike rate. The dashed lines indicate the 95% confidence interval for this estimate. The dots along the time access represent the spike times. The nearest neighbor variable bandwidth parameter, ***nn***, is set to 0.7 for ***locfit***.

estimate, our tutorial script has run *locfit* using the following syntax:

```
>> fit=locfit(data,'family','rate')
```

followed by

```
>> lfplot(fit)
```

and

```
>> lfband(fit)
```

(Fig. 4, dashed lines in bottom subplot). In this case, we have opted to fit our single-trial spike train to a rate function by setting the *family* parameter of *locfit* to *rate*. Alternatively, we could have smoothed the spike data by choosing to fit it to a density function in which the smoothing is meant to determine the probability of firing a spike as a function of time. We generated density estimates by setting the *family* parameter in *locfit* to *density* instead of *rate*.

Note the dots that appear along the time axis of the bottom subplot: These are the spike times for the trial under consideration. *Locfit* will fit a linear, quadratic, or other user-specified function to some subset of the spikes, using each spike time in turn as the center point for the least-squares fit. The number of spikes in each subset can be stipulated in one of two ways: (1) as a fixed "bandwidth", i.e., time interval. For example, if the parameter *h*=1 (and the spike times are given in units of seconds), then each local fit to the data will include 1 s of the trial; or (2) with *h*=0, and *nn* (nearest neighbor parameter) set to some fraction such as 0.3, in which case the time interval surrounding each spike will expand (or shrink) until 30% of the total number of spikes in the time series is included.

```
>> fit=locfit(data,'family','rate','h',1)
```

will produce a fit using a fixed bandwidth of 1 s. The bottom subplot of Figure 4 was produced with a nearest neighbor variable bandwidth,

```
>> fit=locfit(data,'family','rate','nn',0.7)
```

where *nn* was set to 0.7. If we change this value to a smaller fraction, say 0.3, then the smoothing will be done more locally, thereby revealing more of the temporal fluctuations in the spike rate (Fig. 5).

The data input to *locfit* can include multiple trials (following the data format rules outlined in the Appendix to this chapter). As seen in Figure 6 (which our tutorial script will also generate), the resulting fit to our two spike trains appears smoother than the fit to the single trial, even though the nearest neighbor parameter (*nn*) is still 0.3. Although the regression is always done on a single time series, in this case, all the spike times from all the trials for each single unit are collapsed into one vector. Note how a smoother estimate arises in Figure 6 than in Figure 5 owing to the greater continuity across the samples (spike times) of the underlying rate function. Jackknife confidence limits can be computed for the multiple trials case by holding out each trial in turn from the regression fit and then calculating the mean and standard error from the ensemble of drop-one fits.

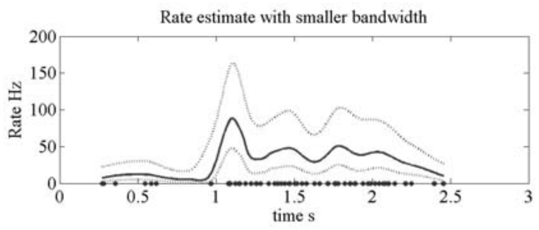


Figure 5. Spike rate estimate using *locfit* in Chronux. Here the nearest neighbor variable bandwidth parameter, *nn*, is set to 0.3.

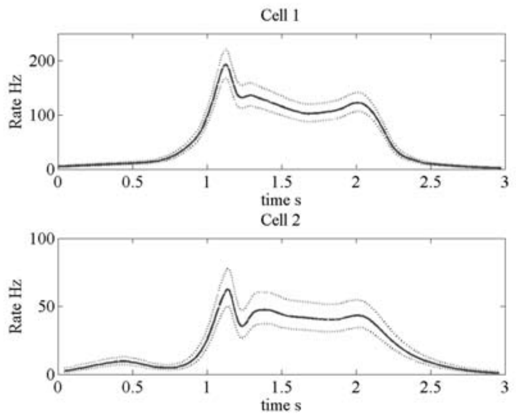


Figure 6. Spike rate estimates using *locfit*. Estimates are constructed using all spike times from all trials shown in the bottom row of Figure 2. The *nn* parameter of *locfit* is set to 0.3.

Spectra

We now begin our frequency–domain exploration of the dynamics of the LFPs and spikes in our data set. Our tutorial script will now generate Figure 7, which illustrates a multitaper spectrum calculated from the continuous voltage record in one LFP channel (sampling rate = 1 KHz) for a single trial. Only the delay period of the trial is included in the data array. The tutorial script *lip_master_script.m* includes the following three lines, which can also be run from the command prompt:

```
>>params.Fs=1000;
>>[S,f]=mtspectrumc(data,params);
>> plot_vector(S,f);   .
```

The first line sets the sampling rate and, therefore, the frequency resolution and range of the spectrum. Many Chronux functions use a structure, *params*, that contains a number of fields for assigning values to the parameters governing the Fourier analysis routines (see Appendix for more about the fields for *params*). The spectrum *S* and frequency range *f* used in the calculation are the outputs of *mtspectrumc*. A Chronux script (the third line in this code segment) can be used to perform special plotting. The default setting for *plot_vector* produces a plot with a log transform of *S* as a function of a linear frequency range. To plot the spectrum on a linear-linear set of axes, use

```
>> plot_vector(S,f,'n')    .
```

In Figure 7, the spectrum is plotted over a default range: from 0 Hz to the Nyquist limit for the sampling rate, 500 Hz. The output range of *S* and *f* is restricted by setting another field in the *params* structure, *params.fpass*. Figure 8 presents the LFP spectrum from the single trial but now with

```
>>params.fpass=[0 100]   ,
```

and then, as before

```
>>params.Fs=1000;
>>[S,f]=mtspectrumc(data,params);
>> plot_vector(S,f);   .
```

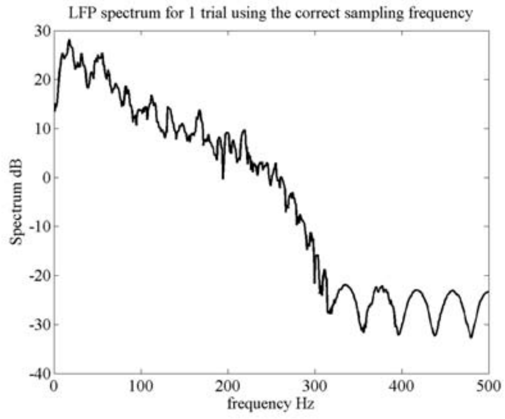


Figure 7. Multitaper spectrum for a single trial LFP; data selected from the delay period. The *y*-axis of the spectrum is in units of dB=10*log$_{10}$(S). ***params.Fs***=1000, ***params.tapers***=[3 5], ***params.fpass***=[0 params.Fs/2], ***params.pad***=0.

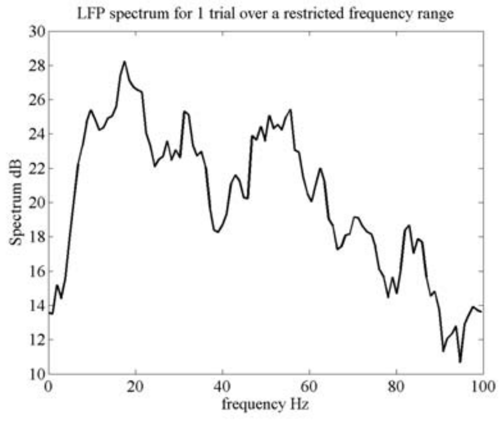


Figure 8. Multitaper spectrum for a single trial LFP. Data selected from the delay period. ***params.Fs***=1000, ***params.tapers***=[3 5], ***params.fpass***=[0 100], ***params.pad***=0.

The tutorial script will generate other examples of band-limited spectra after you choose lower limits and upper limits for the frequency range.

The spacing in the frequency grids used by the fast Fourier transforms (FFTs) called by Chronux can be adjusted through another field in the structure *params*. If *params.pad* = *–1*, then no zeros will be appended to the time series, and the frequency grid will be set by the defaults imposed by MATLAB. With *params.pad* = *0*, the time series will be padded with zeros so that its total length is 512 sample points. For *params.pad* = *1,2,…* the zero padding produces time series that are 1024, 2048,…, samples in length, respectively. As one can see by executing the next code segment,

```
>> params.pad=1;
>>[S,f]=mtspectrumc(data,params);
>> plot_vector(S,f,'y')
>> params.pad=3;
>>[S,f]=mtspectrumc(data,params);
>> plot_vector(S,f,'m')    ,
```

the spectrum generated with a padding factor of 3 (Fig. 9, red) is computed on a much denser grid than the spectrum computed with a padding factor of 1 (Fig. 9, blue plot).

One advantage of taking the multitaper approach to spectral analysis is the ability to control the degree of smoothing in the spectrum. This is accomplished by adjusting the time-bandwidth product of the data windowing, which in turn is established by the choice of the number of Slepian tapers to use. The tutorial script *lip_master_script.m* again calculates the spectrum of the single trial LFP signal, but now with two different degrees of smoothing. As before, the number of tapers to use is set by a field in the

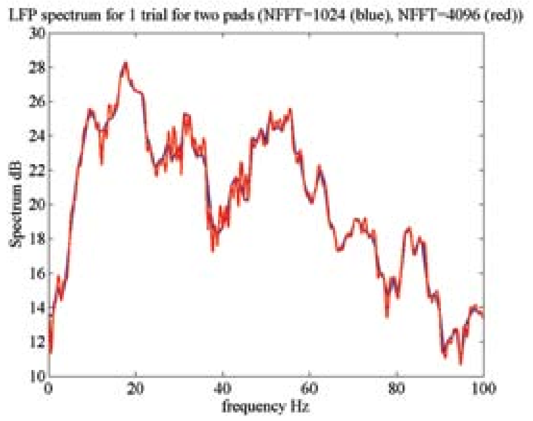


Figure 9. Multitaper spectrum for a single trial LFP; data selected from the delay period. ***params.Fs***=1000, ***params.tapers***=[3 5], ***params.fpass***=[0 100], ***params.pad***=1 (blue), ***params.pad***=3 (red).

structure *params: params.tapers=[TW K]*, where *TW* is the time-bandwidth product and *K* is the number of tapers. For example, if

```
>> params.tapers=[3 5]    ,
```

then the time-bandwidth product is *TW* = 3 and the number of tapers used is 5. The rule *K* = *2*TW* – 1 sets the highest number of tapers that can be used while preserving the good time-frequency concentration of the data windowing available from the Slepian taper sequences. Fewer tapers than the limit of five can be employed, and Chronux will produce a flag when the number of tapers requested is inconsistent with the *TW*. *T* is the length (typically in seconds) of our data segment. One can also think of this value as being established by the [number of samples in data segment] × 1/Fs (inverse of the sampling rate). *W* is the half-bandwidth of the multitaper filter, and if we do not change *T*, we can demonstrate changes in smoothing as a function of changes in the half-bandwidth, as shown in Figure 10. The tutorial script will prompt the user to try other degrees of spectral smoothing by entering new values for the time-bandwidth product.

To compute the spectrum over a number of trials and return an average thereof, we set the *trialave* field in the structure *params* to 1. The tutorial script will carry out the following steps:

```
>> params.trialave=1;
>>[S,f]=mtspectrumc(data,params);
>> plot_vector(S,f)    .
```

If *trialave* = 0, and the structured array *data* have any number of trials, the output *S* will be a matrix where each column is the spectrum computed from one trial's neural signal.

Chronux will also calculate and plot error bars for multitaper spectra. Two different types of confidence interval estimates are available. If we set the field *err* in *params* to

`>> params.err=[1 p], with p=0.05,` and then execute

```
>>[S,f,Serr]=mtspectrumc(data,params);
>>plot_vector(S,f,[],Serr);    ,
```

Chronux will plot the spectrum bracketed by the theoretical 95% confidence limits for that estimate. The array *Serr* contains the (1 – *p*)% limits, with the lower limit in the first row and the upper limit in the second row of the array. In this case, the confidence bounds are based on the parametric distribution for the variance of a random variable, i.e., the chi-square, with two degrees of freedom. If instead, we set the field *err* in *params* to

```
>> params.err=[2 p], with p=0.05    ,
```

the 95% confidence bounds will be derived from a jackknife estimate of the standard error for the sample spectra. Thus, if we run the lines of code given above for the theoretical confidence interval, and continue with

```
>> hold
>>p=0.05;
>>params.err=[2 p];
>>[S,f,Serr]=mtspectrumc(data,params);
>>plot(f,10*log10(Serr(1,:)),'r');
>>plot(f,10*log10(Serr(2,:)),'r');    ,
```

a figure similar to that seen in Figure 11 should be rendered by the tutorial script.

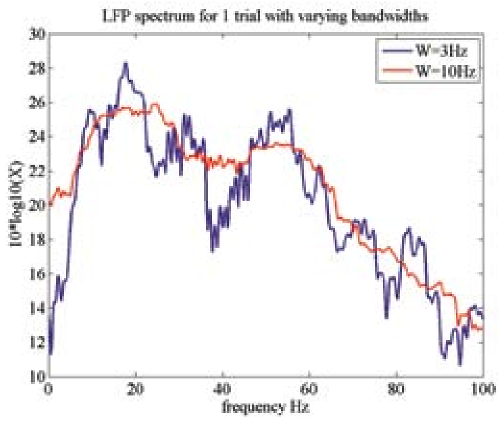


Figure 10. Multitaper spectrum for a single trial LFP; data selected from the delay period (1 s duration, so T = 1). ***params.Fs***=1000, ***params.tapers***=[3 5] (blue), ***params.tapers***=[10 19] (red), ***params.fpass***=[0 100], params.pad=2

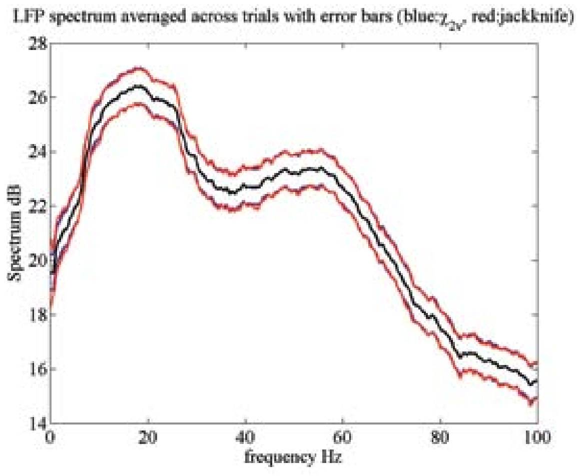


Figure 11. Multitaper spectrum for LFP using all trials associated with one target; data selected from the delay period. ***params.Fs***=1000, ***params.tapers***=[10 19], ***params.fpass***=[0 100], ***params.pad***=2, ***params.trialave***=1 (average spectrum shown in black), ***params.err***=[1 .05] (blue), ***params.err***=[2 .05] (red).

Note that for these data, the jackknife confidence interval (in red) is in good agreement with the so-called theoretical interval (in blue).

As discussed in detail in other chapters, multitaper spectra can be calculated for point process data. As described in the Appendix herein, Chronux contains a whole set of analogous scripts for point processes that match those for continuous data. However, the suffixes of the script names carry a *pt* or *pb*, for point times and point binned, respectively, instead of a *c*, for continuous. For example, the script *mtspectrumpt.m* will compute the multitaper spectrum for data represented as a series of spike times. The following section of MATLAB code will extract a data segment of interest from the trials, set the appropriate params fields, compute the spectrum for the spike data, and plot the results:

```
data=dsp1t;                   % data from 1st cell
delay_times=[1 2];            % start and end time
                              of delay period
data=extractdatapt
(data,delay_times,1);         % extracts spikes within
                              delay period
params.Fs=1000;               % inverse of the spacing
                              between points on the
                              grid used for computing
                              Slepian functions
params.fpass=[0 100];         % range of frequencies
                              of interest
params.tapers=[10 19];        % tapers
params.trialave=1;            % average over trials
p=0.05;                       % p value for errors
params.err=[1 p];             % chi2 errors
[S,f,R,Serr]=mtspectrumpt
(data,params);
```

The output should be similar to that presented in Figure 12. The tutorial script should be able to produce this figure. One thing to note here is the high number of tapers, 19, used for computing the spectrum. Owing to the inherent complexity of even a single spike's power spectrum, extensive smoothing often helps represent spike spectra. The output variable *R* is something unique to spike spectra: It is the high-frequency estimate of the spike rate derived from the spectrum. This estimate is either made on a trial-by-trial basis or based on the average, depending on the setting of the parameter *params.trialave*. In Figure 12, the mean rate estimates appear as dotted horizontal lines.

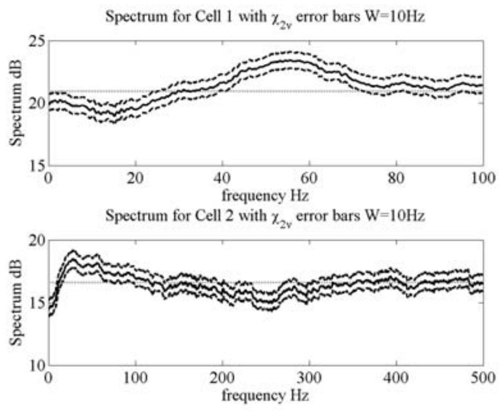


Figure 12. Multitaper spectrum for two spikes recorded in area LIP; delay period activity only. Top: Cell 1, ***params.Fs***=1000, ***params.tapers***=[10 19], ***params.fpass***=[0 100], ***params.pad***=0, ***params.trialave***=1 (average, heavy line), ***params.err***=[1 .05] (dashed lines), mean rate estimate (dotted horizontal line). Bottom: Cell 2, ***params.Fs***=1000, ***params.tapers***=[10 19], ***params.fpass***=[0 500], ***params.pad***=0, ***params.trialave***=1 (average, heavy line), ***params.err***=[1 .05] (dashed lines), mean rate estimate (dotted horizontal line).

Spectrograms

This section will illustrate how Chronux controls the calculation of time-frequency representations of neural data. Chronux can generate spectrograms for continuous data (like EEGs and LFPs) as well as point process activity (spike times and binned spike counts). An important component of the Chronux spectrogram is the sliding window, which sets the width of the data window (usually specified in seconds) and how much the window should slide along the time axis between samples. Within each

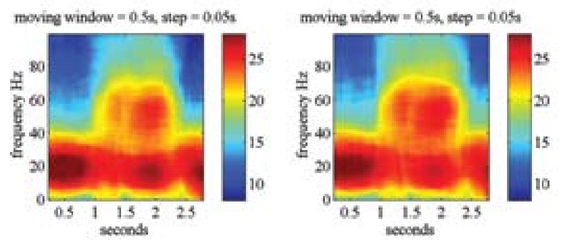


Figure 13. Time-frequency spectrograms for two LFP channels. Activity from all trials, over the entire trial (3 s) used for the analysis. ***Movingwin***=[.5 .05], ***params.Fs***=1000, ***params.tapers***=[5 9], ***params.fpass***=[0 100], ***params.pad***=0, ***params.trialave***=1, ***params.err***=0.

window, multitaper spectral analysis of the data proceeds as it does when calculating standard spectra. However, one must remember that the spectral analysis is restricted to the temporal window for the data. Thus, the number of tapers used for the spectrogram should reflect the time-bandwidth product of the window, not the dimensions of the entire data segment of interest. Extensive temporal overlapping between successive windows will tend to produce smoother spectrograms. The following code fragment from the tutorial script (*lip_master_script.m*) will help generate spectrograms for two of the LFP channels in our data set (Fig. 13):

```
movingwin=[0.5 0.05];        % set the moving
                             window dimensions
params.Fs=1000;              % sampling frequency
params.fpass=[0 100];        % frequencies of
                             interest
params.tapers=[5 9];         % tapers
params.trialave=1;           % average over trials
params.err=0;                % no error
                             computation

data=dlfp1t;                 % data from channel 1
[S1,t,f]=mtspecgramc
(data,movingwin,params);     % compute
                             spectrogram

subplot(121)
plot_matrix(S1,t,f);
xlabel([]);                  % plot spectrogram
caxis([8 28]); colorbar;

data=dlfp2t;                 % data from channel 2
[S2,t,f]=mtspecgramc
(data,movingwin,params);     % compute
                             spectrogram

subplot(122);
plot_matrix(S2,t,f);
xlabel([]);                  % plot spectrogram
caxis([8 28]); colorbar;
```

Note the use of the special Chronux plotting routine *plot_matrix*. Here the window is set to 500 ms in duration with a slide of 50 ms along the time axis between successive windows.

The same sets of parameters used for continuous LFP signals can be employed for calculating the spike spectrograms (Fig. 14). However, one useful modification to make when plotting spike spectrograms is to normalize the spike power *S* by the mean firing rate *R*. For example,

```
data=dsp1t;                  % data from 1st cell
[S,t,f,R]=mtspecgrampt
(data,movingwin,params);     % compute
                             spectrogram
figure;
subplot(211);
                             % plot spectrogram
                             normalized by rate
plot_matrix(S./repmat(R,
[1 size(S,2)]),t,f);xlabel([]);
caxis([-5 6]);colorbar;

data=dsp2t;                  % data from 2nd cell
[S,t,f,R]=mtspecgrampt
(data,movingwin,params);     % compute
                             spectrogram
subplot(212);
                             % plot spectrogram
                             normalized by rate
plot_matrix(S./repmat(R,
[1 size(S,2)]),t,f);
caxis([-5 6]);colorbar;
```

The normalized spectrograms demonstrate how the spike power fluctuates across the trials with respect to the mean rate. Here one can readily observe that, while there is an increase in gamma-band power in the spike discharge with respect to the mean rate during the delay period (Fig. 14, yellow-orange colors, top subplot), the power in the lower-frequency fluctuations in the spike discharge is suppressed with respect to the mean rate (blue colors).

Coherence

As an example of the use of Chronux software for evaluating the strength of correlations between different neural signals, we will calculate the spike-field coherence for pairs drawn from the three LFP channels and two spike channels in our monkey parietal lobe data set. As discussed in "Spectral Analysis for Neural Signals," spike-field coherence is a frequency-domain representation of the similarity of dynamics between a spike train and the voltage fluctuations produced by activity in the spiking neuron's local neural environ-

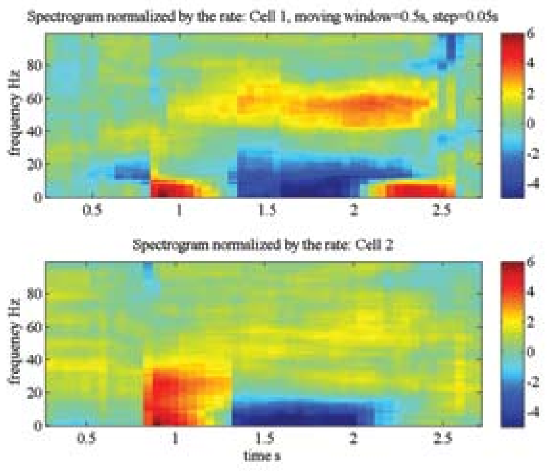


Figure 14. Time-frequency spike spectrograms for two spikes recorded in LIP. Activity from all trials, over the entire trial (3 s) used for the analysis. Spectrograms are normalized by the mean rates of the two single units. ***Movingwin***=[.5 .05], ***params.Fs***=1000, ***params.tapers***=[5 9], ***params.fpass***=[0 100], ***params.pad***=0, ***params.trialave***=1, ***params.err***=0.

ment. As before, we start by setting the values of the parameters carried by the structure *params*:

```
params.Fs=1000;               % sampling frequency,
                              same for LFP and spike
params.fpass=[0 100];         % frequency range
                              of interest
params.tapers=[10 19];        % emphasize smoothing
                              for the spikes
params.trialave=1;            % average over trials
params.err=[1 0.05];          % population error bars

delay_times=[1 2];            % define the delay period
                              (between 1 and 2
                              seconds)
datasp=extractdatapt
(dsp1t,delay_times,1);        % extract the spike data
                               from the delay period

datalfp=extractdatac
(dlfp1t,params.Fs,
delay_times);                 % extract the LFP data
                               from the delay period

[C,phi,S12,S1,S2,f,
zerosp,confC,phistd]=
coherencycpt
(datalfp,datasp,params);      % compute the coherence
```

Note that the script for computing the coherency is *coherency**cpt***, a function that handles the hybrid case mixing continuous and point process data. For the outputs, we have the following:

- C, the magnitude of the coherency, a complex quantity (ranges from 0 to 1);
- *phi*, the phase of the coherency;
- *S1* and *S2*, the spectra for the spikes and LFP signals, respectively;
- *f*, the frequency grid used for the calculations;
- *zerosp*, 1 for trials for which there was at least one spike, 0 for trials with no spikes;
- *confC*, confidence level for C at (1 – *p*)% if *params.err*=[1 *p*] or *params.err*=[2 *p*]; and
- *phistd*, theoretical or jackknife standard deviation, depending on the *params.err* selection

These are used to calculate the confidence intervals for the phase of the coherency.

This code segment, which is called by the tutorial script, should generate a graphic similar to Figure 15. The top row of the figure shows the spike-field coherence for spike 1 against the three LFP channels. The bottom row has the spike-field coherence estimates for spike 2 against the three LFP channels. The figure depicts the confidence level for the coherence estimates as a horizontal dotted line running through all the plots; coherence values above this level are significant. We see from this figure that the spikes and LFPs in the monkey parietal cortex showed an enhanced coherence during the delay period for the frequency range from ~25 Hz to more than 100 Hz for all the matchups, except LFP3, with both spikes. For the coherence measures involving LFP3, the coherence is not significant for very fast fluctuations (>90 Hz).

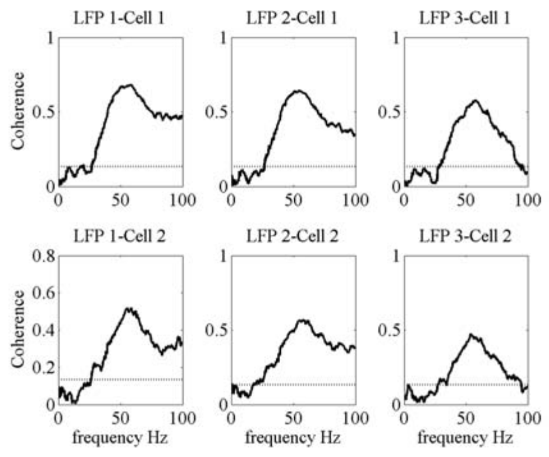


Figure 15. Spike-field coherence. Top row: coherence estimates between cell (spike) 1 with LFP channel 1 (left), LFP channel 2 (middle), and LFP channel 3 (right). Bottom row: coherence estimates between cell (spike) 2 with LFP channel 1 (left), LFP channel 2 (middle), and LFP channel 3 (right). Significance level for the coherence estimates: horizontal dotted line running through all plots.

Using Chronux, we can also estimate the coherence between two spike trains. The setup of the parameters for the calculation is very similar to that required for the hybrid script:

```
>> params.err=[2 p];
>> [C,phi,S12,S1,S2,f,zerosp,confC,phistd,
Cerr]=coherencypt(datasp1,datasp2,params);
```

Here, *phistd* is the jackknifed standard deviation of the phase, and *Cerr* is the (1 – *p*)% confidence interval for the coherence. Figure 16 shows a plot of the spike-spike coherence, comparing the delay period activity (in blue) with the coherence during the baseline (in red).

Denoising

In Figure 17, we expand the Data Conditioning subgraph of the electrophysiology analysis protocol first introduced in Figure 1. The branch for the LFP data carries us through two stages of processing:

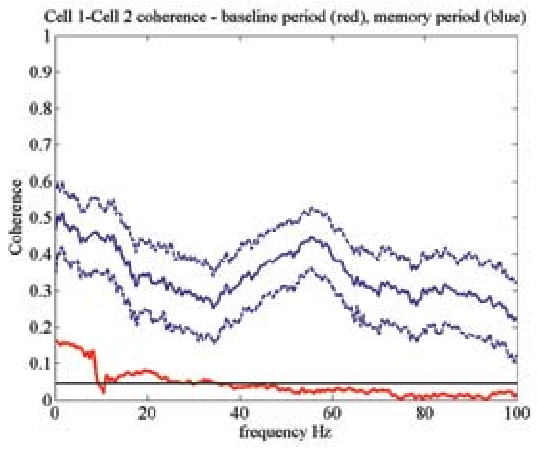


Figure 16. Coherence between spikes of cell 1 and cell 2. Blue traces: data restricted to the delay period of each trial (solid line, average coherence; dashed lines, 95% jackknife confidence interval for this estimate of the coherence). Red trace: data restricted to the baseline period of each trial. Horizontal line: significance level for the coherence estimates. ***params.Fs***=1000, ***params.tapers***=[10 19], ***params.fpass***=[0 100], ***params.pad***=0, ***params.trialave***=1, ***params.err***=[2 .05].

local detrending and the testing and removal of 60 Hz line noise. Electrophysiological recordings, both in the research laboratory and in clinical settings, are prone to contamination. 60 Hz line noise (50 Hz in Europe), slow drifts in baseline voltage, electrical transients, ECG, and breathing movements all contribute different types of distortion to the recorded signal. Methods for removing particular waveforms, such as ECG and large electrical transients, have good solutions that are treated elsewhere (Perasan B, "Spectral Analysis for Neural Signals"; Mitra and Pesaran, 1999; Sornborger et al., 2005; Mitra and Bokil, 2008). We will focus here on slow fluctuations in electrophysiological signals and line noise.

If we add a sinusoidal voltage fluctuation to one of

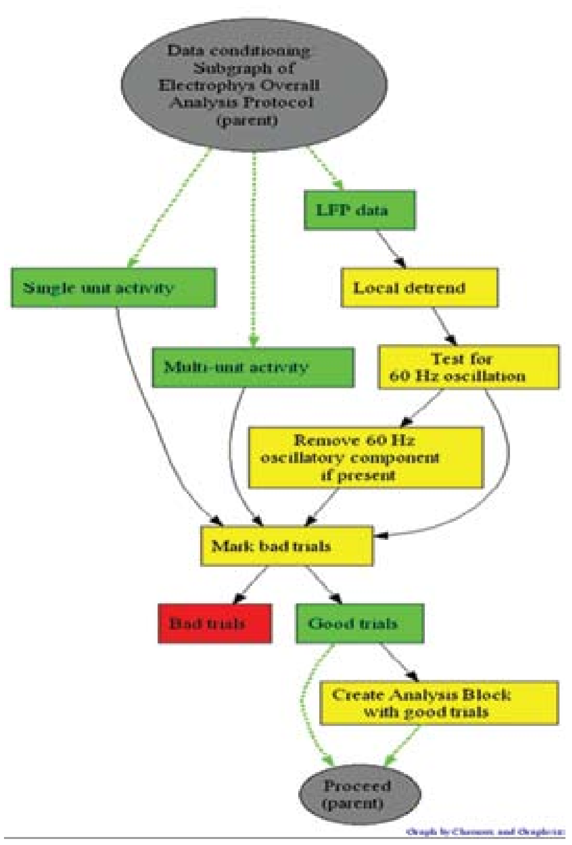

Data conditioning Subgraph of Electrophys Overall Analysis Protocol (parent)
LFP data
Single unit activity
Local detrend
Multi-unit activity
Test for 60 Hz oscillation
Remove 60 Hz oscillatory component if present
Mark bad trials
Bad trials
Good trials
Create Analysis Block with good trials
Proceed (parent)
Graph by Chronux and Graphviz

Figure 17. Data conditioning component of electrophysiological analysis protocol.

our LIP LFP recordings, the result will look something like Figure 18 (top left). Such a slow fluctuation could be entirely the result of changes in the electrostatic charges built up around the recording environment. Therefore, it is noise and we should try to remove it. With Chronux, we use a local linear regression to detrend neural signals. The script *locdetrend* utilizes a moving window controlled by *params* to select time samples from the signal. The best fitting line, in a least-squares sense, for each sample is weighted and combined to estimate the slow fluctuation, which is then removed from the data signal.

For Figure 18 (top middle),

```
>> dLFP=locdetrend(LFP,[.1 .05]).
```

The dimensions of the sampling window are 100 ms

NOTES

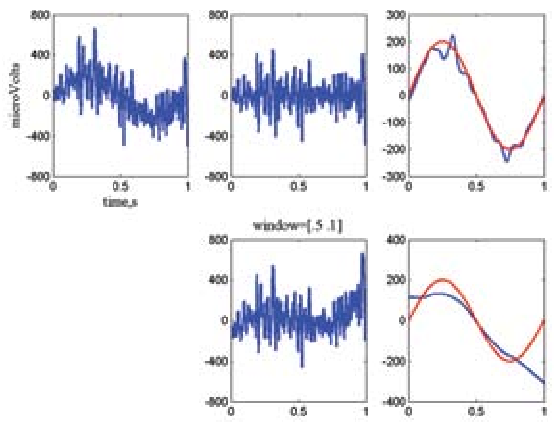


Figure 18. Application of *locdetrend*.

in duration, with a window shift of 50 ms between samples. Note that the detrended signal has much less low-frequency fluctuation than the original signal (Fig. 18, top left). In Figure 18 (top right), we see that the estimate of the slow fluctuation (blue) does a pretty good job of capturing the actual signal (red) that was added to the LFP data. However, if the sampling window parameters are not well matched to changes in the signal, the detrending will not be successful.

For Figure 18 (bottom, center),

```
>> dLFP=locdetrend(LFP,[.5 .1]).
```

Window duration = 500 ms, half the sample length with a window shift of 100 ms. Here the estimate of the slow fluctuation (blue) does a poor job of capturing the sinusoid (Fig. 18, red, bottom right).

Chronux accomplishes the removal of 60 Hz line noise by applying Thomson's regression method for detecting sinusoids in signals (Thomson, 1982). This method does not require that the data signal have a uniform (white) power spectrum. The Chronux script *rmlinesc* can either remove a sinusoid of chosen frequency or automatically remove any harmonics whose power exceeds a criterion in the F-distribution (the F-test). Figure 19 demonstrates the application of the F-test option of *rmlinesc*.

```
>>no60LFP=rmlinesc(LFP,params).
```

Here the LFP (Fig. 19, top left) has been contaminated with the addition of a 60 Hz sinusoid. The multitaper spectrum of this signal is shown in Figure 19 (top right panel). Note the prominent 60 Hz element in this spectrum (broadened but well defined by the application of the multitaper technique). The spectrum of *no60LFP* is shown in the figure's bottom left panel; the time series with the 60 Hz noise removed, the vector returned by *rmlinesc*, is shown in the bottom right panel.

Appendix: Chronux Scripts

While not an exhaustive list of what is available in Chronux, the scripts enumerated here (discussed in this chapter) are often some of the most useful for trying first during the early phase of exploratory data analysis. This section describes the means for setting some of the more important parameters for controlling multitaper spectral calculations, as well as the basic rules for formatting input data.

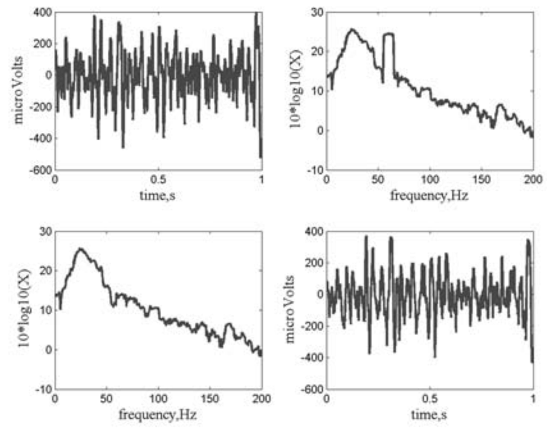


Figure 19. Application of *rmlinesc*.

Denoising

(1) Slow variations (e.g., movements of a patient for EEG data)
   locdetrend.m: Loess method
(2) 50/60 Hz line noise
   rmlinesc.m
   rmlinesmovingwinc.m

Spectra and coherences (continuous processes)

(1) Fourier transforms using multiple tapers
   mtfftc.m
(2) Spectrum
   mtspectrumc.m
(3) Spectrogram
   mtspecgramc.m
(4) Coherency
   mtcoherencyc.m
(5) Coherogram
   mtcohgramc.m

Analogous scripts are available for analyzing time series data organized in alternative formats. Point-process time series data can be analyzed using *mtfft**pt**.m*, *mtspectrum**pt**.m*, etc. Binned spike count data can be analyzed with *mtfft**tb**.m*, *mtspectrum**b**.m*, etc. An additional set of scripts is available for calculating the coherence between a continuous series and a point-process time series (*coherency**cpt**.m*, *coherogram**cpt**.m*, etc.), and for the coherence between a continuous and binned spike counts (*coherency**cpb**.m*, *coherogram**cpb**.m*, etc).

In a typical function call, such as

[S,f,Serr]=mtspectrumc(data,params), a structure *params* is passed to the script. This structure sets values for a number of important parameters used by this and many other algorithms in Chronux.

params.Fs	Sampling frequency (e.g., if data are sampled at 1 kHz, use 1000).
params.tapers	Number of tapers to use in spectral analysis specified by either passing a matrix of precalculated Slepian tapers (using the dpss function in MATLAB) or calculating the time-frequency bandwidth and the number of tapers as [*NW K*], where *K* is the number of tapers. Default values are params.tapers=[3 5].
params.pad	Amount of zero-padding for the FFT routines utilized in the multitaper spectral analysis algorithms. If pad = –1, no padding; if pad = 0, the FFT is padded to 512 points; if pad = 1, the FFT is padded to 1024 points, pad = 2, padding is 2048 points, etc. For a spectrum with a dense frequency grid, use more padding.
params.fpass	Frequency range of interest. As a default, [0 Fs/2] will allow from DC up to the Nyquist limit of the sampling rate.
params.err	Controls error computation. For err=[1 *p*], so-called theoretical error bars at significance level *p* are generated and placed in the output Serr; err=[2 *p*] for jackknife error bars; err=[0 *p*] or err=0 for no error bars (make sure that Serr is not requested in the output in this case).
params.trialavg	If 1, average over trials/channels; if set to 0 (default), no averaging.

Local regression and likelihood

(1) Regression and likelihood
    locfit.m
(2) Plotting the fit
    lfplot.m
(3) Plotting local confidence bands
    lfband.m
(4) Plotting global confidence bands
    scb.m

Data format

(1) Continuous/binned spike count data

Matrices with dimensions: time (rows) × trials/channels (columns)

Example: 1000 × 10 matrix is interpreted as 1000 time-point samples for 10 trials from 1 channel or 1 trial from 10 channels. If multiple trials and channels are used, then add more columns to the matrix for each additional trial and channel.

(2) Spike times

Structured array with dimension equal to the number of trials/channels.

Example: data(1).times=[0.3 0.35 0.42 0.6]
data(2).times=[0.2 0.22 0.35]

Chronux interprets *data* as two trials/channels: four spikes collected at the times (in seconds) listed in the bracket for the first trial/channel, and three spikes collected in the second trial/channel.

Supported third-party data formats

NeuroExplorer (.NEX)
Plexon (both .PLX and .DDT file formats)

References

Mitra PP, Bokil H (2008) Observed Brain Dynamics. New York: Oxford UP.

Mitra PP, Pesaran B (1999) Analysis of dynamic brain imaging data. Biophys J 76:691-708.

Sornborger A, Yokoo T, Delorme A, Sailstad C, Sirovich L (2005) Extraction of the average and differential dynamical response in stimulus-locked experimental data. J Neurosci Methods 141:223-229.

Thomson DJ (1982) Spectrum estimation and harmonic analysis. Proc IEEE 70:1055-1096.

Neural Signal Processing Tutorial II: Point Process Model Estimation and Goodness-of-Fit Analysis

Uri T. Eden, PhD[1], Lakshminarayan Srinivasan, PhD[2], and Sridevi V. Sarma, PhD[3]

[1]Department of Mathematics and Statistics, Boston University
Boston, Massachusetts

[2]Department of Neurosurgery, Massachusetts General Hospital
Boston, Massachusetts

[3]Department of Brain and Cognitive Systems, MIT
Cambridge, Massachusetts

NOTES

Introduction

Statistical models are used to explicitly propose a relationship between neural spiking activity and other measured signals, including biological stimuli and behavioral covariates. Generally speaking, these models can be useful for several tasks: summarizing spike train data, describing structure or patterns in the data, making inferences about the neural system, and estimating signals or dynamic properties of the firing activity. Neural models of spike train activity can also include biological models (e.g., Hodgkin-Huxley–type neurons or networks) that more concretely depict the mechanisms by which these relationships arise.

This tutorial goes through the basic steps for constructing and evaluating a statistical model for a spiking neuron. In particular, we will use generalized linear models (GLMs) to construct an inhomogeneous Poisson model of a place cell in the CA1 region of rat hippocampus. The GLM framework provides a flexible class of conditional intensity models that are easily fit using maximum likelihood methods and can be easily implemented in many standard mathematical and statistical packages, such as Matlab.

As we perform this analysis, here are some focus questions to keep in mind: First, what constitutes a statistical model for spike train data? Second, how do we specify a particular statistical model for a particular neural system? Finally, once a model class is specified, how can we choose the model parameters that are most in line with an observed spike train? We can answer the first question immediately: The statistical model specifies the probability of observing some number of spikes in any time interval, given (i.e., conditioned on) related covariates such as stimulus values or the spiking history. We will explore the answers to the other questions below.

Experimental Setup

In this tutorial, we will analyze spike train data recorded from a CA1 hippocampal neuron in a Long–Evans rat freely foraging in an open circular environment (70 cm in diameter with walls 30 cm high and a fixed visual cue). Both place-cell microelectrode array recordings and position data were captured from the animal. The neural activity was sampled at 1000 Hz, and the rat's position was sampled at 30 Hz. These experimental methods have been previously reported in detail (Brown et al., 1998).

Constructing a statistical model

The first step in performing statistical analysis of this spiking activity is to visualize the data in a way that clarifies the relationship between the spiking activity and the rat's position.

Plotting raw data

To plot the raw data, we must first load in the data from a Matlab-readable file (e.g., .csv, .txt, .mat) using the "load" command. We can see what variables lie in the data set that we loaded by typing the command "who," as shown below. You may type "help load" to see how to load files of different formats.

```
>> load glm_data.mat
>> who
```

Your variables are listed in Table 1.

Table 1. Variables contained in glmdata.mat

spike_times	Time stamps on spike events
*x*_at_spike_times	Animal position (x) at corresponding times stamps
*y*_at_spike_times	Animal position (y) at corresponding times stamps
T	Time
spikes_binned	spike data in 33 ms bins
*x*N	Normalized position (x)
*y*N	Normalized position (y)
*vx*N	Normalized velocity (v_x)
*vy*N	Normalized velocity (v_y)
R	Movement speed = sqrt($v_x^2 + v_y^2$)
phi	Movement direction = atan2(v_y, v_x)

✎ Use Matlab to visualize the spiking activity as a function of time and then as a function of the animal's position.

Using the "plot" command, you should be able to generate the plot shown in Figure 1, which shows the rat's trajectory (blue) over 23 minutes and the spikes (red) for the single neuron. Notice how this particular neuron fires more when the rat is in the bottom, slightly left section of the circular environment.

Choosing a model form

The model is a function that specifies the probability of observing ΔN_k spikes in the kth interval of length Δ milliseconds. It is premised on parameters of the model (θ) and values of other signals of interest, i.e., covariates ($x^{(1)}, x^{(2)}, \ldots, x^{(N)}$), that are postulated to affect the current probability of spiking, such as our stimulus and the occurrence of previous spikes:

$$\Pr(\Delta N_k \mid \theta, x^{(1)}, \ldots, x^{(N)}) = f(\Delta N_k, \theta, x^{(1)}, \ldots, x^{(N)})$$

NOTES

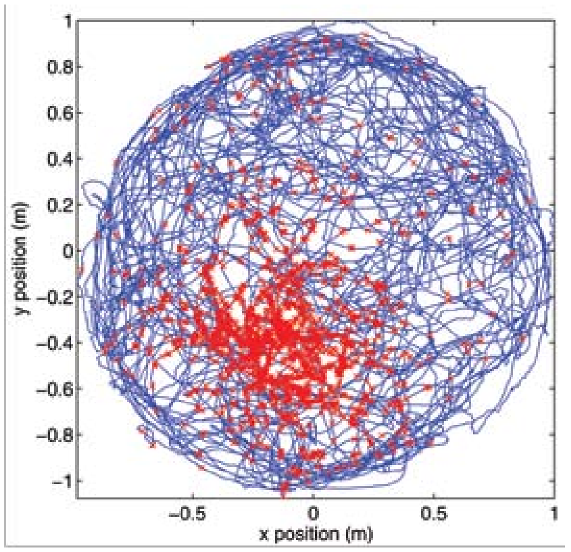


Figure 1. Rat's trajectory (blue) and spikes (red) for the single neuron firing.

Let's explore one way to specify this probability. First, define the function λ_k that can be used to calculate the instantaneous probability that a spike will occur. We can introduce parameters and covariates into the model by defining λ_k in terms of θ and $x^{(1)}, \ldots, x^{(N)}$.

For this exercise, we will assume that the bin size is set small enough that the covariates are approximately constant within the bin, so that λ_k will be approximately constant over each bin. We assume the number of spikes that arrive in the kth bin is distributed as a Poisson distribution with mean and variance $\lambda_k\Delta$.

To make parameter estimation easier, we restrict the relationship between λ_k, θ, and $x^{(i)}$ to follow the GLM framework for a Poisson distribution (Brillinger et al., 1988; Truccolo et al., 2005). This means simply the following:

$$\log(\lambda_k) = \beta_o + \sum_{j=1}^{J} \alpha_j f_j(x(k)) ,$$

where $\theta = \{\beta_0, \alpha_1, \alpha_2, \ldots, \alpha_J\}$, $x(k) = \{x^1(k), x^2(k), \ldots, x^N(k)\}$ and $\{f_1, f_2, \ldots, f_J\}$ is a set of general functions of the covariates. Note that the log of λ_k is linear in θ, and that the functions can be arbitrary, nonlinear functions. This makes the GLM model class quite broad. In addition, if different types of covariates (e.g., extrinsic vs intrinsic) independently impact the firing probability, then the GLM can model these separately as follows:

$$\log(\lambda_k) = \beta_o + \sum_{j=1}^{J} \alpha_j f_j(\text{extrinsic}(k)) + \sum_{m=1}^{M} \gamma_m g_m(\text{intrinsic}(k)).$$

The GLM is an extension of the multiple linear regression model in which the variable being predicted (in this case, spike times) need not be Gaussian (McCullagh and Nelder, 1989). GLM provides an efficient computational scheme for model parameter estimation and a likelihood framework in which to conduct statistical inferences based on the estimated model (Brown et al., 2003).

✎ Pick a set of covariates (remember, this can include any function of the variables above), and write an equation that linearly relates $\log(\lambda_k)$ to your covariates.

Writing down the data likelihood

The data likelihood is simply the probability of observing the specific sequence of spike counts, conditioned on selected values of the parameters, and the actual values of the other covariates: $L(\theta = \Pr(\Delta N_1, \ldots, \Delta N_k \mid x^{(1)}, \ldots, x^{(N)}, \theta)$, where the observed data are held fixed.

✎ Write down an equation for $L(\theta)$.

In order to derive the likelihood function, we consider a case where the time bin Δ is so small that only 0 or 1 spike can occur (e.g., Δ = 1 ms). The spike train then forms a sequence of conditionally independent Bernoulli trials, with the probability of a spike in the kth time interval given by $\Pr(\text{spike in } (k\Delta, (k+1)\Delta] \mid x^{(1)}, \ldots, x^{(n)}\theta) \approx \lambda_k\Delta$. This yields the following joint probability:

$$\Pr(\Delta N_1, \ldots, \Delta N_k \mid x^{(1)}, \ldots, x^{(N)}, \theta) \approx \prod_{i=1}^{k} [\lambda_i\Delta]^{\Delta N_i} [1-\lambda_i\Delta]^{1-\Delta N_i}$$

For small Δ, we get the result that $[1-\lambda_j\Delta] \approx e^{-\lambda_j\Delta}$ and $\log([\lambda_i\Delta][1-\lambda_i\Delta]^{-1}) \approx \log(\lambda_i\Delta)$; therefore,

$$\Pr(\Delta N_1, \ldots, \Delta N_k \mid x^{(1)}, \ldots, x^{(N)}, \theta) \approx \prod_{i=1}^{k} \left[\frac{\lambda_i\Delta}{1-\lambda_i\Delta}\right]^{\Delta N_i} [e^{-\lambda_i\Delta}].$$

Then, taking the log of both sides results in the following data likelihood function:

$$L(\text{Spike Train} \mid \theta) = \exp(\sum_{k=1}^{T} \log(\lambda_k\Delta t)\Delta N_k - \lambda_k\Delta t)$$

More details on deriving the data likelihood function can be found in Brown et al., 2003.

Choosing model parameters to maximize the data likelihood

We would now like to choose model parameters that most likely would have generated the observed data. This is accomplished by choosing the parameter estimate $\hat{\theta}$ that maximizes the data likelihood:

$$\hat{\theta} = \arg\max L(\theta)$$

Our problem of estimating model parameters has become a question of optimizing a function L(θ) over a possibly unconstrained space θ. Several methods can be used to approach optimization. Because we chose models built on the GLM framework, this optimization is computationally simplified (McCullagh and Nelder, 1989).

Use the Matlab function glmfit to find the parameters of your model that maximize L(θ).

```
>> b = glmfit([covariate1 covariate2 …],spikes_binned,'poisson');
```

For small-parameter vectors, it is often useful to plot L(θ) as a function of θ around the parameter values returned by glmfit in order to confirm that glmfit indeed maximized the data likelihood. The shape of the likelihood around the maximum likelihood estimate determines the uncertainty about the estimates from the data.

Visualizing the model

Using the equation for λ you wrote down in step 3, plot the intensity as a function of your covariates. Compare this graph with a plot of the signal values at which spikes occurred. This will visually confirm the plausibility of the model for explaining the relationship between the observed spiking and your covariates.

The script `glm_part1.m` is set up to fit and visualize a GLM for the spiking activity as a function of the rat's position.

```
>> glm_part1
```

This outputs a plot of the raw spiking data (Fig. 1) and another plot of the linear model of spiking rate (covariates = [xN yN]), both as a function of the animal's position (Fig. 2).

Notice that the linear model in Figure 2 cannot capture the phenomenon seen in the raw data (Fig. 1), namely that in the lower, slightly lefthand side of the circle, the spiking decreases. Thus, a quadratic or Gaussian model may be more appropriate for capturing the place-field structure shown in Figure 1.

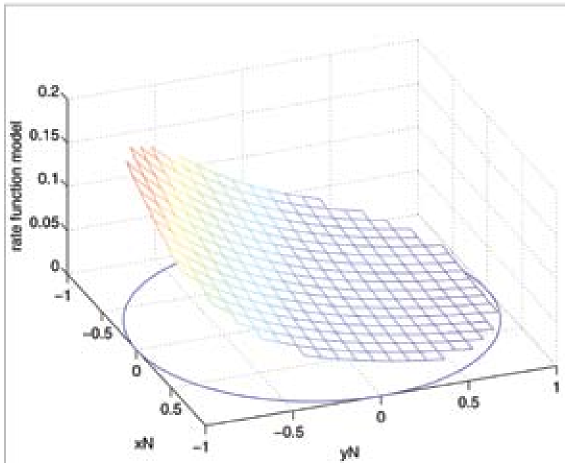


Figure 2. The linear model of spiking rate as a function of the animal's position.

Modify the GLM in the script `glm_part1` to include other covariates and functions of covariates. Modify line 20 by including your new covariates in the model fit, and modify line 30 by writing in the equation for λ. Can you think of a GLM that can capture a Gaussian-shaped place-field structure?

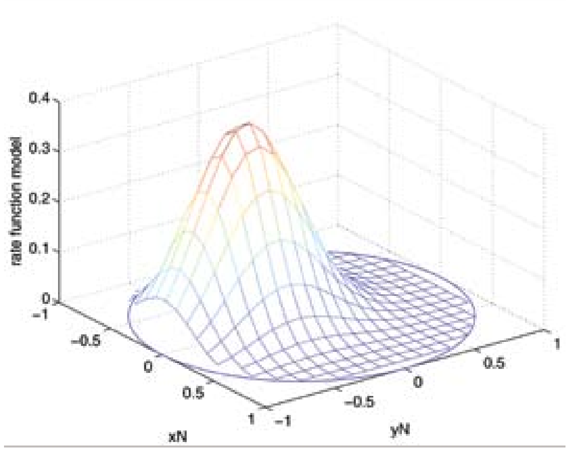


Figure 3. The Gaussian model of spiking rate as a function of the animal's position.

The `script glm_part2.m` is set up to visualize models of spiking as a function of velocity.

```
>> glm_part2
```

NOTES

Currently, this script outputs occupancy-normalized histograms of spiking simply as a function of the velocity covariates, as shown in Figure 4. Examining these histograms can provide insight into possible spiking models.

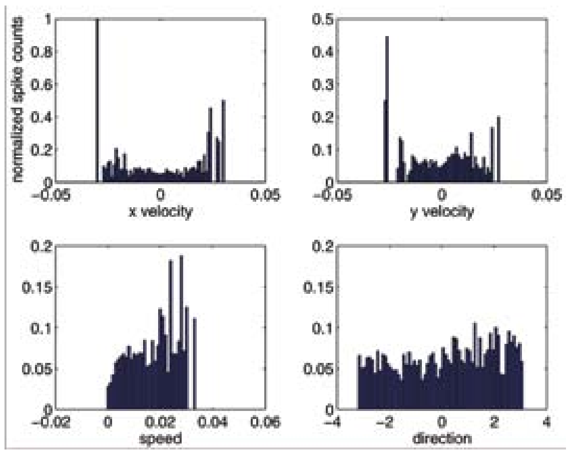


Figure 4. Occupancy-normalized histograms of spiking as a function of the velocity covariates.

Modify the script `glm_part2` to fit GLM models to each of these variables, and then plot the GLM fits along with the occupancy-normalized histograms. What do these model fits suggest about the relationship between spiking and velocity?

Evaluating a statistical model

Now that we've written down a statistical model that explicitly relates neural signals with covariates such as stimulus values, and used maximum likelihood to choose values for parameters that were most consistent with the observed data, what comes next? In this section, we will investigate how to evaluate our model both in relative terms (how well it does compared with other models) and in absolute terms (how well it does in explaining the given data).

Here are some focus questions to consider as we attempt to evaluate the statistical models we just constructed: First, how precise are the model parameters that we just fit? Is it possible to simplify the model and still maintain accuracy? How do we know whether we have too many model parameters, and why might this be a bad thing? Can we quan-titatively determine which of the models we constructed better fits the data and whether any of the models describes the data well enough to make useful inferences about the underlying properties of the neural system?

We will see that the data likelihood can also be used to determine precision in the choice of parameter, based on the Fisher information. We can use the data likelihood to provide confidence intervals for our parameters and eliminate parameters that are not distinguishable from zero. With too many parameters, a model may not generalize well to new data sets. The Akaike Information Criterion (AIC) and other measures quantify the tradeoff between fitting the data better and increasing the number of parameters. The Kolmogorov–Smirnov (KS) plot can help visualize how well the model explains the structure in all the data.

Determining the uncertainty of the parameter estimates

We can obtain confidence intervals around the parameter estimates based on the Fisher information. The observed Fisher information matrix is defined as the second partial derivative (the Hessian) of the log likelihood function log $f(x\,|\,\theta)$ with respect to the individual parameters, evaluated at the following maximum likelihood estimate:

$$I_{obs}(\theta) = \left.\frac{-\partial^2 \log\, f(x_{obs}\,|\,\theta)}{\partial\theta^2}\right|_{\hat{\theta}_{ML}}.$$

This quantity provides a sense for the precision of our model parameters. Recall that the ML estimate of θ was chosen so that $\frac{\partial \log\, f(x_{obs}\,|\,\theta)}{\partial\theta} = 0$. If this derivative gradually goes to zero at around θ_{ML}, then presumably, the neighboring values of θ_{ML} would have been almost as good and might have been chosen depending on the accuracy of the optimization procedure.

Revisit the GLM models that you constructed in Part 1 of this tutorial. Load the data.

```
>> load glm_data.mat
```

Now call glmfit() in the following way to return additional statistics about the model fit:

`[b,dev,stats] = glmfit` (arguments of the function call);

Generate a number of possible models with this command, including one with quadratic terms in *x* and *y* position in relation to the animal.

✎ Examine the confidence intervals computed for each parameter of two models based on the square root of the inverse of the observed Fisher information (σ) given in the variable `stats.se`. Use `errorbar()` to plot the parameters $b \pm 2\sigma$, and determine which parameters are statistically different from zero.

Figure 5, below, plots the confidence intervals for the quadratic model with six parameters ([*x*N *y*N *x*N.^2 *y*N.^2 *x*N.***y*N]) . Note that the sixth parameter has a confidence interval that includes 0; thus, the parameter is not statistically different from zero.

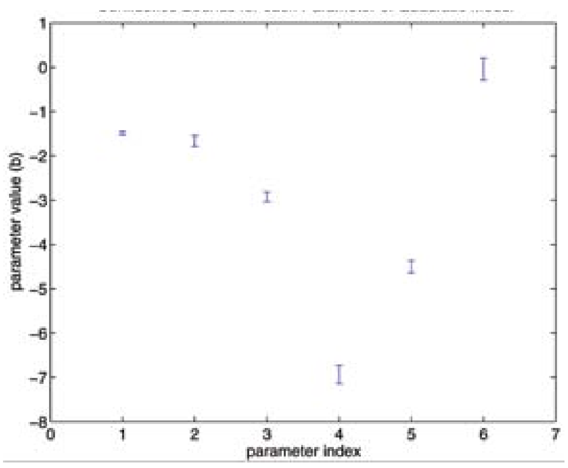


Figure 5. Confidence intervals for parameters of quadratic model.

✎ Examine the corresponding *p* value assigned to each parameter in `stats.p` . Which parameters have significant *p* values?

The corresponding *p* values for the six parameters in the quadratic model are all smaller than 10^–4 except for the sixth parameter, which is 0.8740.

Constructing confidence intervals about your model conditional intensity

In the previous section, we computed the uncertainty about the parameter estimates. Now, we can propagate this uncertainty into the intensity model. This gives us a sense of how certain we can be about the conditional intensity as a function of the covariates.

✎ Use the Matlab function `glmval` to calculate confidence intervals about the conditional intensity.

```
>> [b, dev, stats] = glmfit([xN
xN.^2],spikes_binned,'poisson');
>> [yhat,dylo,dyhi] = glmval(b,[-1:.01:1;
[-1:.01:1].^2]','log',stats);
>> errorbar(-1:.01:1,yhat,dylo,dyhi);
```

If you build the above model, you will get the uncertainty into the intensity model shown in Figure 6, below.

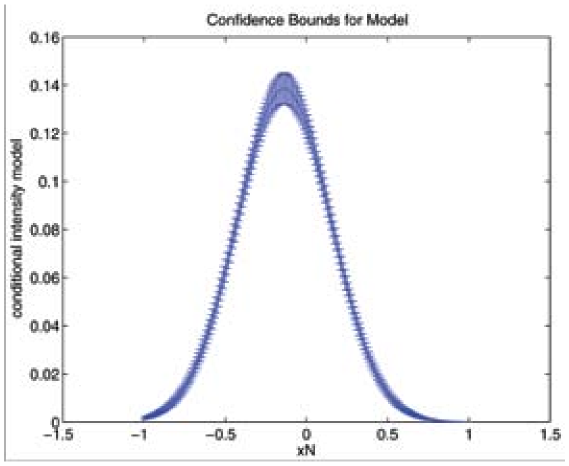


Figure 6. Uncertainty propagated into the intensity model for *x*N-quadratic model.

Characterizing the relative goodness-of-fit among competing models

Now we have a sense of which parameters contributed to the model in explaining the relationship between covariates and the observed neural activity. But how do we compare the quality of competing models, especially when they may have different numbers of parameters?

One answer is to examine the deviance of each model, defined as –2 times the log of the data likelihood:

$$deviance = -2\log f(x \mid \theta)$$

This quantity grows smaller as the data become more likely under a particular model. It is a generalization of the mean-squared error under a Gaussian linear model.

We would also like to penalize our error for having too many parameters. Intuitively, having too many parameters may restrict the ability to allow the model to generalize to (predict) new data. The AIC balances deviance against the number of parameters *P*:

$$AIC = -2\log f(x \mid \theta) + 2P$$

✎ Compare the AIC between two models. For example, consider the GLM models that are linear versus quadratic in relation to the animal's position.

NOTES

The AIC for the linear model where covariates include [*x*N *y*N] is 1.6175e + 004, while the AIC for the quadratic model [*x*N *y*N *x*N.^2 *y*N.^2 *x*N.**y*N] is 1.2545e + 004. These criteria indicate that the quadratic model is considerably better than the linear model.

✎ How can the AIC be improved for the GLM quadratic model of the animal's position?

We saw above that the cross-term between *x*N and *y*N has an associated parameter that is close to zero and therefore does not contribute significantly to the model. If we eliminate this parameter and build a new quadratic model with covariates [*x*N *y*N *x*N.^2 *y*N.^2], we get an AIC of 1.2543e + 004, which is slightly less than that of the previous quadratic model. This suggests that the model without the quadratic cross-term provides a more parsimonious description of the data.

✎ How is the AIC of a given model affected by eliminating statistically significant parameters?

If we remove the first parameter of the linear model (*x*N) and build a GLM model with covariate *y*N, we get an AIC of 1.6315e + 004, which is higher than that of the full linear model. In general, removing statistically significant parameters significantly increases the AIC.

Characterizing the goodness-of-fit between data and model

The AIC allowed us to compare alternative models. But how good is the best model in explaining the statistical structure in the data? The KS statistic provides one absolute measure.

Let's briefly revisit the time rescaling theorem. This theorem states that, with the correct conditional intensity function, interspike intervals (ISIs) can be transformed into an independent, identically distributed, exponential set of random variables. This transforms any point process into a unit rate Poisson process. Consequently, if we did have the correct conditional intensity function, the cumulative distribution on rescaled ISIs would match that of an exponential distribution with $\lambda = 1$.

To form the KS statistic, first rescale the empirical ISIs with your model conditional intensity function. Compare the cumulative distribution of these rescaled ISIs to that of an exponential distribution with $\lambda = 1$. The KS statistic is the maximum of the absolute value of the difference between these cumulative distributions.

✎ Open `glm_part1_ks.m` and adjust the GLM models to your liking. Now scroll down to the KS Plot section of the code. Edit the code to correctly compute your model intensity at each point in time. Running the code will now output the KS statistic and plot, which includes the 95% confidence interval. Compare the KS statistic between two competing models. Which one does better? Does that model do a good job at explaining all the statistical structure in the data?

Figure 7 below shows the KS plots of the linear and full quadratic model with cross-term. Note that the quadratic model has better absolute goodness-of-fit to data. The fact that the KS plot for the quadratic model still does not fit entirely inside the 95% confidence intervals suggests that some statistical structure remains that this model fails to capture.

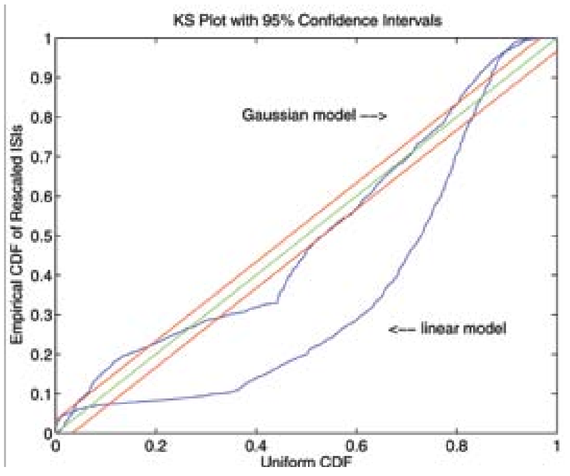


Figure 7. KS plots of linear and quadratic models.

✎ How could this statistical model be further improved? What other variables might be related to the spiking activity of this neuron? Is there a single correct model for these data?

References

Brillinger DR (1988) Maximum likelihood analysis of spike trains of interacting nerve cells. Biol Cybern 59:189-200.

Brown EN, Frank LM, Tang D, Quirk MC, Wilson MA (1998) A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. J Neurosci 18:7411-7425.

NOTES

Brown EN, Barbieri R, Eden UT, and Frank LM (2003) Likelihood methods for neural data analysis. In: Feng J, ed. Computational neuroscience: A comprehensive approach, Chap 9, pp 253-286. London: CRC Press.

McCullagh P, Nelder JA (1989) Generalized linear models, 2nd ed. Boca Raton, FL: Chapman & Hall/CRC Press.

Truccolo W, Eden UT, Fellow MR, Donoghue JP, Brown EN (2005). A point process framework for relating neural spiking activity for spiking history, neural ensemble and extrinsic covariate effects. J Neurophys 93:1074-1089.

SOCIETY FOR NEUROSCIENCE

1121 14th Street, NW • Suite 1010 • Washington, DC 20005 • (202) 962-4000 • www.sfn.org