LETTER **Communicated by Bruno Olshausen**

Efficient Coding of Time-Relative Structure Using Spikes

Evan Smith
evan+@cnbc.cmu.edu
Department of Psychology, Center for the Neural Basis of Cognition,
Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.

Michael S. Lewicki
lewicki@cnbc.cmu.edu
Department of Computer Science, Center for the Neural Basis of Cognition,
Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.

Nonstationary acoustic features provide essential cues for many auditory tasks, including sound localization, auditory stream analysis, and speech recognition. These features can best be characterized relative to a precise point in time, such as the onset of a sound or the beginning of a harmonic periodicity. Extracting these types of features is a difficult problem. Part of the difficulty is that with standard block-based signal analysis methods, the representation is sensitive to the arbitrary alignment of the blocks with respect to the signal. Convolutional techniques such as shift-invariant transformations can reduce this sensitivity, but these do not yield a code that is efficient, that is, one that forms a nonredundant representation of the underlying structure. Here, we develop a non-block-based method for signal representation that is both time relative and efficient. Signals are represented using a linear superposition of time-shiftable kernel functions, each with an associated magnitude and temporal position. Signal decomposition in this method is a non-linear process that consists of optimizing the kernel function scaling coefficients and temporal positions to form an efficient, shift-invariant representation. We demonstrate the properties of this representation for the purpose of characterizing structure in various types of nonstationary acoustic signals. The computational problem investigated here has direct relevance to the neural coding at the auditory nerve and the more general issue of how to encode complex, time-varying signals with a population of spiking neurons.

1 Introduction

Nonstationary and time-relative acoustic structures such as transients, timing relations among acoustic events, and harmonic periodicities provide essential cues for many types of auditory processing. In sound localization,

human subjects can reliably detect interaural time differences as small as 10 μs, which corresponds to a binaural sound source shift of about 1 degree (Blauert, 1997). In comparison, the sampling interval for an audio CD sampled at 44.1 kHz is 22.7 microseconds. Auditory grouping cues, such as common onset and offset, harmonic comodulation, and sound source location, all rely on accurate representation of timing and periodicity (Slaney & Lyon, 1993). Time-relative structure is also crucial for the recognition of consonants and many types of transient, nonstationary sounds. Neurophysiological research in the auditory brainstem of mammals has found cells capable of conveying precise phase information up to 4 kHz or of tracking the quickly varying envelope of a high-frequency sound (Oertel, 1999). The importance of these acoustic cues has long been recognized, but extracting them from natural signals still poses many challenges because the problem is fundamentally ill posed. In natural acoustic environments, with multiple sound sources and background noises, acoustic events are not directly observable and must be inferred using numerous ambiguous cues.

Another reason for the difficulty in obtaining these cues is that most approaches to signal representation are block based; the signal is processed piecewise in a series of discrete blocks. Transients and nonstationary periodicities in the signal can be temporally smeared across blocks. Large changes in the representation of an acoustic event can occur depending on the arbitrary alignment of the processing blocks with events in the signal. Signal analysis techniques such as windowing or the choice of the transform can reduce these effects, but it would be preferable if the representation was insensitive to signal shifts.

Shift invariance alone, however, is not a sufficient constraint on designing a general sound processing algorithm. Another important constraint is coding efficiency or, equivalently, the ability of the representation to capture underlying structure in the signal. A desirable code should reduce the information rate from the raw signal so that the underlying structures are more directly observable. Signal processing algorithms can be viewed as a method for progressively reducing the information rate until one is left with only the information of interest. We can make a distinction between the observable information rate, or the rate of the observable variables, and the intrinsic information rate, or the rate of the underlying structure of interest. In speech, the observable information rate of the waveform samples is about 50,000 bits per second, but the intrinsic rate of the underlying words is only around 200 bits per second (Rabiner & Levinson, 1981). Information reduction can be achieved by either selecting only the desired information (and discarding everything else) or removing redundancy, such as the temporal correlations between samples. This reduces the observable information rate while preserving the intrinsic information.

In this letter, we investigate algorithms for fitting an efficient, shift-invariant representation to natural sound signals. The outline of the letter is as follows. The next section describes the motivations behind this approach

and illustrates some of the shortcomings of current methods. After defining the model for signal representation, we present different algorithms for signal decomposition and contrast their complexity. Next, we illustrate the properties of the representation on various types of speech sounds. We then present a measure of coding efficiency and compare these algorithms to traditional methods for signal representation. Finally, we discuss the relevance of the computational issues discussed here to spike coding and signal representation at the auditory nerve.

2 Representing Nonstationary Acoustic Structure

Encoding the acoustic signal is the first step in any algorithm for performing an auditory task. There are numerous approaches to this problem, which differ in both their computational complexity and in what aspects of signal structure are extracted. Ultimately, the choice about what the representation encodes depends on the tasks that need to be performed. In the ideal case, the encoding process extracts only that information necessary to perform the task and suppresses noise or unrelated information. A generalist approach, like that taken by most mammalian auditory systems, requires a representation that is efficient for a wide range of signals. As natural sounds contain both relatively stationary harmonic structure (e.g., animal vocalizations) as well as nonstationary transient structure (e.g., crunching leaves and twigs), this generalist approach requires a code capable of efficiently representing these disparate sound classes (Lewicki, 2002a). Here we seek an auditory representation that is useful for a variety of different tasks.

2.1 Block-Based Representations. Most approaches to signal representation are block based, in which signal processing takes place on a series of overlapping, discrete blocks. This not only obscures transients and periodicities in the signal, but can also have the effect that for nonstationary signals, small time shifts can produce large changes in the representation, depending on whether and where a particular acoustic event falls within the block. Figure 1 illustrates the sensitivity of block-based representation with small shifts in speech signals. The upper panel shows a short speech waveform sectioned into blocks using two sequences of Hamming windows (solid and dashed curves). Each window spans approximately 30 msecs (512 samples) and successive blocks (A1, A2, and so on) are shifted by 10 msecs. The B blocks offset from the A blocks by an amount indicated by the dot-dash vertical lines ($\sim$ 5 msecs), representing the arbitrary alignment of the signal with respect to the two block sequences. The lower panel shows spectral representations for the three corresponding blocks (solid for the A blocks, dashed for the B blocks). The jagged upper curves show the power spectra for each windowed waveform. The smooth lower curves (offset by -20 dB) show the spectrum of the optimal filter derived by linear predictive coding.

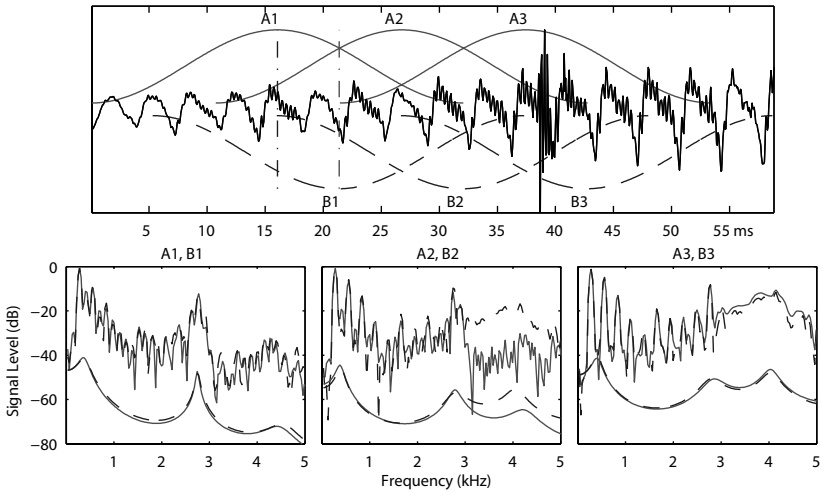

Figure 1: Block-based representations are sensitive to temporal shifts. (Top panel) A speech waveform with two sets of overlaid Hamming windows, A1–3 (continuous lines above waveform) and B1–3 (dashed lines below the waveform). (Lower panels) The power spectrum (jagged) and LPC spectrum (smooth) of hamming windows offset by less than 5 ms are overlaid (A, continuous; B, dashed). In either of these, small shifts (e.g., from A2 to B2) can lead to large changes in the representation.

The sound used in Figure 1 is /et/ in the context of the word *Vietnamese*. The three-block sequence contains an abrupt, transient signal feature, a relatively high-frequency, and high-amplitude /t/ sound occurring at about the 38th msec. The windows preceding the /t/, A1 and B1, contain only the /ee/ vowel waveform. The spectra of these windows nearly overlap, although differences resulting from the slow change in the vowel can be seen. The spectra for windows A2 and B2 show a dramatic difference in the range of 3 to 5 kHz. This results entirely from the arbitrary alignment of each window and the /t/. Because window B2 contains a significant portion of the /t/ waveform, it shows a pronounced increase in powering the higher range. The spectra for the following windows, A3 and B3, are again nearly overlapping, as the /t/ is well represented in both windows. Notice that the increase in the power for window B2 is not as great as that for the final windows. This implies that the alignment of the B sequence will cause a temporal smearing of the constant onset, spreading the energy of the 2 msec transient over a 10 msec window. Discrimination of the phonemes, such as /ba/ and /pa/, is based on differences as small as 5 to 10 msecs in voice-onset time (Liberman, Delattre, & Cooper, 1958). The temporal smear-

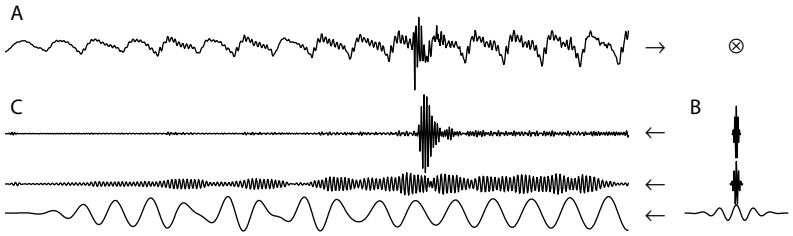

Figure 2: A continuous filter bank produces a shift-invariant representation but does not reduce the information rate. An input signal (A) is convolved with a filter bank (B). The output of the convolution (C) has increased the dimensionality of the input signal.

ing illustrated in Figure 1 could create an ambiguity in the onset of voicing and lead to an alteration in the phoneme perception of a listener.

2.2 Convolutional Representations. One way to minimize the shift sensitivity problem is to increase the block rate. This reduces the variability of the observed spectra but results in a very inefficient code, because there are then several, slowly changing representations of the same underlying acoustic events. In the limit, increasing the block rate simply produces a filter bank in which windowed sinusoids are convolved with the signal. Although this yields a representation that is invariant to shifts, a major drawback is that a filter bank does not reduce the information rate because the dimensionality of the each output is identical to the input; furthermore, there is one output for each filter. This problem is illustrated in Figure 2. The speech waveform in the top row of Figure 2A (the /et/ in *Vietnamese* and identical to that used in figure 1) is convolved with each of the three (time domain) filters shown in the right column (see Figure 2B). The filters are Gabor functions with peak resonance frequencies at the first and second formants (360 and 2750 Hz) and 4000 Hz. The filter outputs (see Figure 2C) show that the formant energy is roughly constant throughout the sound, while energy in the /t/ is relatively localized. Clearly, it would be preferable to have an efficient representation that was insensitive to signal shift, preserving transients and harmonic shifts, but encoded structure in an event-based fashion.

3 A Sparse, Shiftable Kernel Representation

Here we employ a sparse, shiftable kernel method of signal representation (Lewicki & Sejnowski, 1999; Lewicki, 2002b). In this model, the signal $x(t)$ is encoded with a set of kernel functions, $\phi_1, \ldots, \phi_M$, that can be positioned arbitrarily and independently in time. The mathematical form of the repre-

sentation with additive noise is

$$x(t) = \sum_{m=1}^{M} \sum_{i=1}^{n_m} s_i^m \phi_m(t - \tau_i^m) + \epsilon(t), \tag{3.1}$$

where τ_i^m and s_i^m are the temporal position and coefficient of the ith instance of kernel ϕ_m, respectively. The notation n_m indicates the number of instances of ϕ_m, which need not be the same across kernels. In addition, the kernels are not restricted in form or length.

A more general way to express equation 3.1 is to assume that the kernel functions exist at all time points during the signal and let the nonzero coefficients determine the positions of the kernel functions. In this case, the model can be expressed in convolutional form,

$$x(t) = \sum_{m} \int s_m(\tau)\phi_m(t - \tau)d\tau + \epsilon(t), \tag{3.2}$$

where $s_m(\tau)$ is the coefficient at time τ for ϕ_m. By using a sparse coefficient signal $s_m(t)$ composed only of delta functions, equation 3.2 reduces to equation 3.1. A similar approach assuming only sparse coefficients has been used for coding of natural movies (Olshausen, 2002).

The key theoretical abstraction of the model is that the signal is decomposed in terms of discrete acoustic events, represented by the kernel functions, each of which has a precise amplitude and temporal position. Here we assume the kernels are gammatone functions (gamma-modulated sinusoids) whose center frequency and width are set according to an equivalent rectangular band (ERB) filter bank cochlear model using Slaney's auditory toolbox for Matlab (Slaney, 1998). Except where noted, we used a set of 64 kernel functions for the results below. The use of gammatone functions is well motivated by both biology and natural sound statistics (Lewicki, 2002a). In principle, we could also adapt the set of kernel functions to maximize the efficiency of the code.

Figure 3 illustrates the generative model. A signal is represented in terms of a sparse set of discrete temporal events, a spike code. For example, the waveform in Figure 3A consists of three aperiodic "chirps," each composed of discrete acoustic events with differing amplitudes but identical relative temporal alignments. This signal can be represented by nine spikes, each with a precise time and amplitude. We can plot this representation in terms of a spikegram, Figure 3B, where the nine spikes are shown as ovals of varying size, intensity, and position. Each oval indicates the temporal and spectral position (center of mass and center frequency, respectively) of one gammatone kernel function, with oval size and intensity indicating the amplitude of the kernel coefficient. Representing a kernel's temporal position based on its center of mass causes them all to align precisely given a delta function as input. We adopt this convention to help illustrate the temporal precision of the spike code.

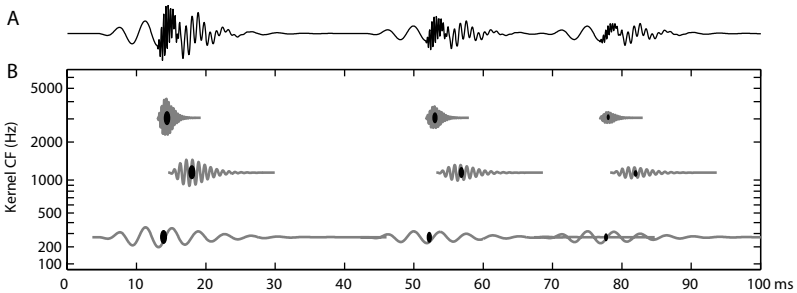

Figure 3: An illustration of generative model and its spikegram representation. The signal (A) is represented in the spikegram (B) as a set of ovals whose size and intensity indicate the amplitude of the spike. The position of the oval indicates the kernel center frequency (CF, y-axis) and timing (x-axis). The gammatone functions corresponding to the spikes (represented by each oval) are overlaid in gray.

3.1 Encoding Algorithms. Equation 3.1 specifies the generative form of the model but does not provide an encoding algorithm, that is, how to compute the optimal values of τ_i^m and s_i^m for a given signal. The computational objective is to minimize the error $\epsilon(t)$ while maximizing coding efficiency. As is the case with most coding algorithms, there is a trade-off between the error of the representation and the computational complexity of the algorithm. For the results here, we used three different encoding algorithms to select values for τ_i^m and s_i^m. These show a clear trade-off between complexity and accuracy, but we can gain some flexibility along these dimensions by hybridizing, using the simpler algorithms to initialize the most complex.

3.1.1 Filter Threshold. One approach to efficient audio coding has been to use filter banks based on the human cochlea (Baumgarte, 2002; Lyon, 1982; Shamma, 1985; Gitza, 1988; Patterson, Holdsworth, Nimo-Smith, & Rice, 1988). The filter-threshold algorithm is a computationally simple approximation of cochlear processing. This is a causal approach, and it begins by convolving the signal with the full set of kernel functions from the gammatone ERB filter bank. (Note that for all of the algorithms described here, the kernels are restricted to have unit norm.) The encoded coefficients and times, s_i^m and τ_i^m, are chosen based on the values and positions of all convolution peaks that exceed a preset threshold. This greatly reduces the size of the observable information rate compared to the convolutional representation, but some degree of (threshold-dependant) temporal and spectral redundancy remains. Filter banks with more than 16 gammatones kernel functions are highly overcomplete, but the filter threshold algorithm does not take the correlations between kernel functions into account during cod-

ing. As a result, it tends to be a poor estimate of the signal given our linear superposition model. We compensate for this to some degree by adding a single parameter to scale the coefficients. Despite its shortcomings under our model, filter threshold is relatively fast and resilient to noise due to its inherent redundancy. These could be desirable properties depending on the task the system must perform.

3.1.2 Matching Pursuit. An obvious improvement on filter threshold would be to account explicitly for the correlations between kernels, iteratively regressing the signal onto the kernels. This is a noncausal approach, but our goal here is to determine the optimal signal representation. One well-studied formalization of this approach is the *matching pursuit* algorithm (Mallat & Zhang, 1993). We employ it here to produce a more efficient estimate of the τ_i^m and s_i^m values for a given signal.

Our goal is to decompose the signal, $x(t)$, over a set of kernels selected from the gammatone filter bank so as to best capture the structure of the signal. Matching pursuit's approach to this problem is to iteratively approximate the input signal with successive orthogonal projections onto some basis (in this case the unit-normed gammatone kernels). The signal can be decomposed into

$$x(t) = \langle x(t)\phi_m \rangle \phi_m + R_x(t), \tag{3.3}$$

where $\langle x(t)\phi_m \rangle$ is the inner product between the signal and the kernel and is equivalent to s_m in equation 3.1. The final term in equation 3.3, $R_x(t)$, is the residual signal after approximating $x(t)$ in the direction of ϕ_m. The projection with the largest inner product will minimize the power of $R_x(t)$, thereby capturing the most structure possible given a single kernel.

Equation 3.2 can be rewritten more generally as

$$R_x^n(t) = \langle R_x^n(t)\phi_m \rangle \phi_m + R_x^{n+1}(t), \tag{3.4}$$

with $R_x^0(t) = x(t)$ at the start of the algorithm. With each iteration, the current residual is projected onto the gammatones. A single kernel is selected such that

$$\phi_m = \arg\max_m \langle R_x^n(t)\phi_m \rangle. \tag{3.5}$$

This best-fitting projection is subtracted out, and its coefficient and time are recorded. This projection and subtraction leaves $\langle R_x^n(t)\phi_m \rangle \phi_m$ orthogonal to the residual signal, $R_x^{n+1}(t)$. It is relatively straightforward to see that each projection is orthogonal to all previous and future projections (Mallat & Zhang, 1993). As a result, matching pursuit codes are composed of mutually orthogonal signal structures.

Assuming the kernels span the signal space, the power of the residual, $R_x^n(t)$, is guaranteed to decrease on each iteration of the algorithm (Mallat & Zhang, 1993; Goodwin & Vetterli, 1999), and so, in the limit, matching pursuit codes will have arbitrarily small error. For most practical purposes, however, some halting criteria should be defined. The simplest is a lower bound on the inner product between the signal and the kernels. We can also track the signal-to-noise ratio of the code over time and stop at a desired fidelity, or halt when some number of spikes has been recorded. More sophisticated criteria are also possible.

We reduce some of the computational overhead of the algorithm by defining local neighborhoods among the kernels via cross-correlation. If the maximal inner product between two kernels across all time shifts was greater than some value θ, then they were included in each other's neighborhood. Typically, θ was set to 0.001 (all kernels were normalized to have an L_2 norm of 1). These neighborhoods are used for reconvolution with the residual signal (i.e., if the last spike involved kernel ϕ_n, then a new residual was calculated only for the neighborhood around n). This can introduce very low magnitude distortion in the code, but the computational cost is significantly reduced as most of the kernels in the filter bank are orthogonal to one another at all time shifts.

3.1.3 MAP Optimization. A probabilistic method for inferring spike amplitudes and times was described in Lewicki and Sejnowski (1999) and Lewicki (2002b). This approach makes no heuristic assumptions about where spikes should occur, for example, selecting convolution maxima as in the previous two algorithms. Instead, the problem is recast in a Bayesian probabilistic framework in which we attempt to maximize the a posteriori distribution of coefficients.

To describe this approach, we begin by expressing the model in matrix form using a discrete sampling of the continuous time series:

$$x = As + \epsilon. \tag{3.6}$$

The rows of the basis matrix, A, contain each gammatone kernel replicated at each sample position making the basis highly overcomplete.

The optimal set of τ_i^m and s_i^m for a signal is found by maximizing the posterior distribution of coefficients given the signal and the gammatones,

$$\hat{s} = \arg\max_s P(s|x, A) = \arg\max_s P(x|A, s)P(s). \tag{3.7}$$

We make two assumptions in modeling the distributions in equation 3.7. First, the noise, ϵ, is gaussian and so the data likelihood, $P(x|A, s)$, is also gaussian. Second, the prior, $P(s)$, a function of the spike times and amplitudes, is very sparse. Given these assumptions, the gradient of equation 3.7

is given by

$$\frac{\partial}{\partial s} \log\ P(s|A, x) \propto\ A^T(x - As) + z(s), \tag{3.8}$$

where $z(s) = (\log\ P(s))'$. $P(s)$ was assumed to follow a Laplacian, but other distributions are possible. The assumption of sparseness of the kernel coefficients means that optimizing equation 3.7 essentially selects out the minimal set of gammatones that best accounts for the structure of the sound signal for a given noise level.

Although optimally efficient codes are possible in theory, in practice only the briefest sounds can be encoded in this manner. For example, using 64 kernels to encode a signal sampled at 44.1 kHz requires approximately 2.8 million coefficients to be optimized per second of signal. We can reduce most of the computational overhead by using filter threshold or matching pursuit to initialize the maximum a posteriori (MAP) optimization. Instead of optimizing over the entire parameter space, these hybrid algorithms search for optimal amplitude values, s, over a set of spike times τ selected by one of the two approximative algorithms. The departure from optimality is a function of the number and "quality" of the spike times selected by the initializing algorithm. In the results that follow, these hybrid algorithms are evaluated alongside the other algorithms as approximations of the true optimally efficient code.

4 Spike Code Signal Representation

The sparse, shiftable kernel representation and a set of decomposition algorithms have now been formalized. To evaluate the model, we will present both examples of the codes it generates and an objective comparison between those and other codes. The following section contains specific examples of spike codes to illustrate its qualities as a method for signal representation and some benefits of time-relative coding.

4.1 Comparison of Encoding Algorithms. There are five possible encoding algorithms described in the previous section: filter threshold, matching pursuit, MAP optimization, optimized filter threshold, and optimized matching pursuit. Figure 4 shows the spike code for a short section of speech (three glottal pulses from the vowel /a/ sampled at 16 kHz) using four of these different encoding algorithms. Even at timescales of 480 to 800 samples, the optimization problem is prohibitive, and an example is not presented here. These spikegrams are formatted identically to Figure 3, with ovals representing the time, center frequency, and magnitude of a spike; only the kernel function overlays are removed. For each, we measure the quality of its representations in terms of signal-to-noise ratio (SNR). To compute this, a reconstruction of the input is generated from the code, and a

residual error is computed between the original and reconstruction. The SNR in decibels (dB) is then $SNR = 10 \log_{10}(P_o/P_e)$, where P_o is the power of the original and P_e is the power of the residual error.

The spikegram in Figure 4A is generated using filter threshold. A high degree of redundancy in both time and frequency is quite evident in the correlated waves of spikes that code each glottal pulse. This redundancy may serve to enhance structural similarity between sound events (e.g., the glottal pulses) and increase the representation's resistance to noise, but it lacks a succinct description of the temporal and spectral characteristics of the sound. Filter threshold encodes the sound to 18 dB SNR (using the scaling parameter mentioned earlier). Perceptually, the input sound is noticeably distorted in the reconstruction, though the speech content is quite clear.

Optimizing the filter threshold code has a dramatic effect on the quality of the encoding, pushing the SNR to 90.1 dB, well beyond the point where the original and reconstructed signals are perceptually discriminable. Given that the original .wav file had 16 bits of precision and assuming coding noise on the order of 1 bit, the estimated SNR of the original signal is about 90 dB. In the example shown in Figure 4B, we assumed a very low level of noise in the model. This results in the majority of spike amplitudes being shifted up or down but few pushed to zero; all of the available information is used to encode the signal accurately. Although few spikes are pruned given an assumption of very low noise, the distribution of spike amplitudes does become sparser as a result of optimization. As progressively high noise levels are assumed, the resulting codes become increasingly sparse, sacrificing SNR in order to prune spikes.

Figure 4C shows an example of the spike code produced by matching pursuit. It is obviously vastly less redundant than the filter threshold code in Figure 4A. There is relatively little obvious structure within the representation of each glottal pulse, implying that primarily independent events are being represented; however, the similarity between pulses is evident. Despite the much more compact representation, the signal is encoded to 30.4 dB SNR, with only very subtle distortions perceivable.

The code generated by a matching pursuit–MAP optimization hybrid (see Figure 4D) is nearly identical to that produced by matching pursuit alone. It is likely for a 30 dB SNR code that the optimization simply corrects some of the error introduced by our use of kernel neighborhoods when computing residuals on each iteration. One possible reason for the limited effect of optimizing is that matching pursuit codes represent a deep local minimum in the parameter space and the gradient method fails to find a global optimum. Another factor concerns the nature of signal decomposition using matching pursuit. This will be discussed further in a later section.

4.2 Convergence of Fidelity. When encoding a signal with matching pursuit, MAP optimization or any hybrid, the SNR of the code increases monotonically with the number of spikes (this is not necessarily true of

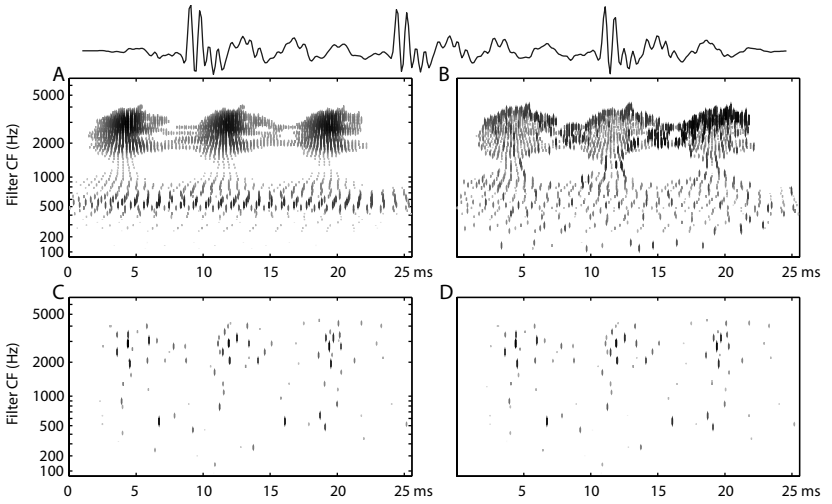

Figure 4: Spikegrams created from an input signal (top) using each of the four algorithms: (A) filter threshold encoded to 18.7 dB SNR (see text), (B) optimized filter threshold encoded to 90.1 dB SNR, (C) matching pursuit encoded to 30.4 dB SNR, and (D) optimized matching pursuit encoded to 33.0 dB SNR.

filter threshold). For the optimized codes, the amount of noise assumed in the model defines the trade-off between sparseness and accuracy. Because these codes are globally optimal, their specific form (the precise location of spikes) may be altered given different noise levels. For matching pursuit, the trade-off is much clearer: lowering the threshold for accepting a spike (or otherwise varying the halting criterion) simply adds additional spikes that code further residual structure. Figure 5 shows the effect of varying the number of spikes in a matching pursuit code. The input signal is a segment of speech (the word *can* sampled at 16 kHz). The spikegram in Figure 5A reflects a very high threshold, producing only 92 actual spikes (about 400 spikes/sec) and a relatively poor representation (10 dB SNR). Above the spikegram is the residual signal from the final iteration of the algorithm. It is apparent that a great deal of structure remains to be coded, although the onset of the consonant, /k/, and the periodicity of the /a/ and /n/ are already revealed. Perceptually, the sound is strongly distorted from the original, but the speech content is quite clear. Figures 5B through 5D show spikegrams and residuals for signals encoded to 20 dB (1600 spikes/sec), 30 dB (3100 spikes/sec), and 40 dB SNR (5500 spikes/sec). By 30 dB, Figure 5C, the distribution of residual amplitudes is not significantly different from a gaussian (based on Lilliefors statistical test to reject gaussian assumption, p-value > 0.2).

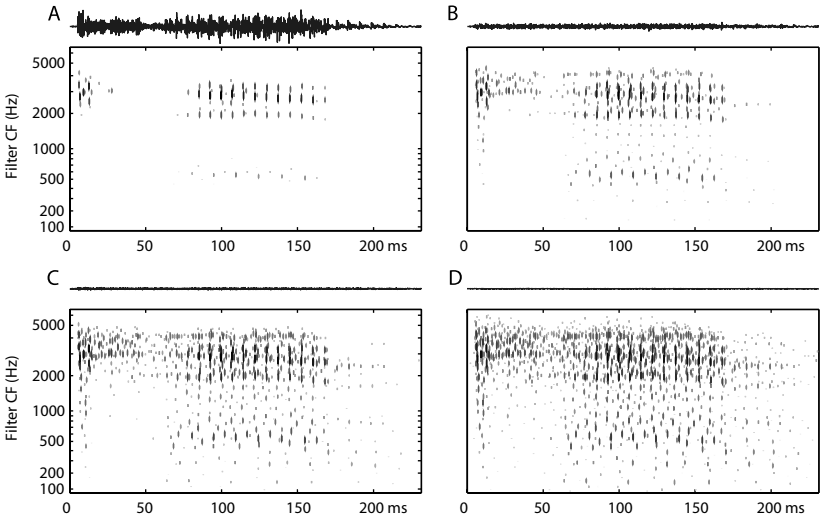

Figure 5: As the spike rate (spikes/sec) increases, the fidelity of the representation increases. The spikegrams above show the improvement of an optimized matching pursuit code with increasing spike rate: (A) 10 dB SNR at 400 spikes/sec, (B) 20 dB SNR at 1600 spikes/sec, (C) 30 dB SNR at 3100 spikes/sec, and (D) 40 dB SNR at 5500 spikes/sec. The residual error is plotted above each spikegram.

4.3 Effect of Kernel Number. Another parameter to be selected for any encoding algorithm is the number of kernel functions. Relatively few gammatones are needed to form a complete basis (i.e., a basis that spans the frequency space of the sounds used), but increasing the number allows greater spectral precision. Figure 6 shows the effect of using matching pursuit with 8, 16, 32, or 64 kernel functions (see Figures 6A–6D, respectively). To be certain that the four sets spanned the frequency space, they were generated independently using Slaney's Matlab toolbox (Slaney, 1998). In each case, the signal is encoded to approximately 40 dB SNR, but the form of the code changes drastically. With relatively few gammatones (see Figures 6A and 6B), the code lacks both spectral and temporal precision. The time-relative coding is largely lost, and the representation becomes nearly convolutional. Using 32 or 64 kernels more clearly segments the acoustic events and begins to show invariant signal structure. Very similar findings were made testing MAP optimization and the hybrid algorithms. In contrast, increasing the number of kernels with filter threshold only shows enhanced spectral precision as spike times are selected independent of one another.

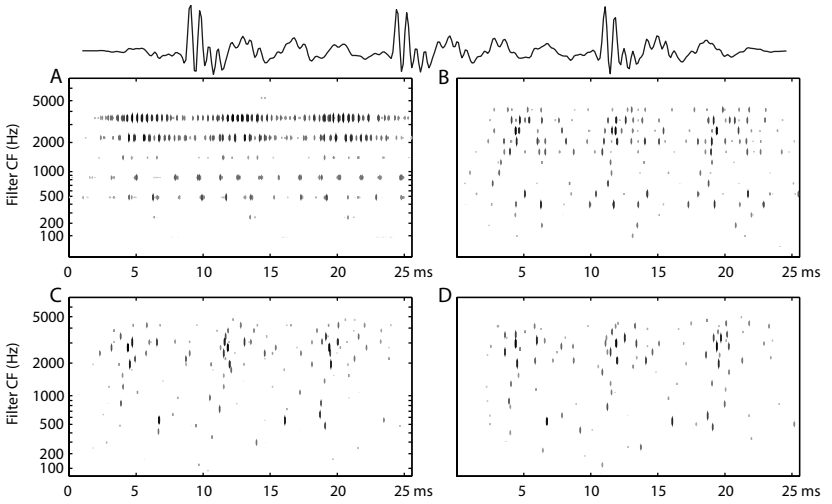

Figure 6: The number of kernel functions affects both the spectral resolution and the temporal sparseness of the spike codes. The input signal (top) was encoded using matching pursuit with 8, 16, 32, or 64 kernel functions (A–D, respectively). The total number of spikes in each is (A) 12011, (B) 1167, (C) 497, and (D) 479.

4.4 Comparison to Spectrograms. Having described some of the details of the spike code, we now look more broadly at its representation. A comparison of a spectrogram and spikegram illustrates many properties of the model. In Figure 7, the upper plot shows the waveform of *pizzerias* spoken by an adult female and sampled at 16 kHz. The spikegram and spectrogram of this signal are shown in the middle and lower plots, respectively. The spikegram was constructed using an optimized filter threshold spike code with 128 ERB-spaced gammatone kernels. Both show the formant and harmonic structure of the vowels (e.g., 320–700 msec). Both also reveal the broad spectral and temporal characteristics of the signal, such as the diffuse energy of the /s/ from 700 to 800 msec. However, while the spectrogram is composed of 10 msec shifted windows (as illustrated in Figure 1), the spikegram possesses precise timing information to the sampling rate of the original signal and retains phase information. This allows it to reveal finely grained synchronous activity across bands. It also possesses a nonlinear frequency axis based on the cochlea. This axis emphasizes the range important to human hearing and is used in many auditory models and speech "front-ends."

4.5 Sparse Representation of Transients. Though the "pizzerias" example demonstrates the large-scale features of the spike code, the fine structure

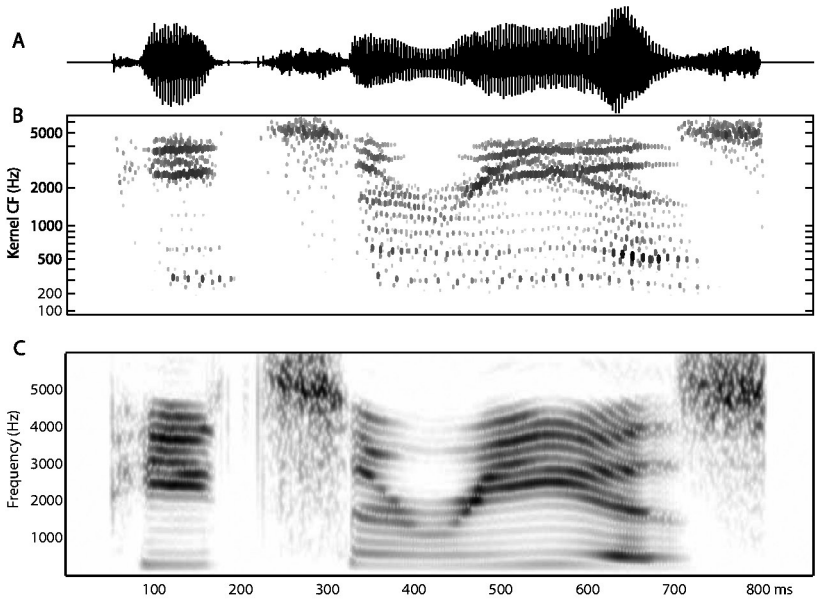

Figure 7: Three representations of the spoken word *pizzerias*. (A) Time-varying waveform. (B) Spikegram. (C) Spectrogram. They are presented on the same timescale (indicated at the bottom). Note that the spikegram and spectrogram use different frequency axes.

is more clearly revealed in a shorter speech segment. The waveform and spikegram of first half of the word *wealth* appear in Figure 8. Here we can see the time-relative coding of nonstationary structure. One hundred msec into the word (about 45 msec from the start of the spikegram), the period between glottal pulses begins to elongate. The spike code maintains a consistent representation of the individual pulses during this period, despite the time dilation. Although there is some slight variability in the representation of each pulse (appropriately reflecting the changes in the underlying signal), the spikes essentially align with the peak of each glottal pulse. Figure 8 also shows the efficiency of the code in representing harmonic structures. The spikegram shown can reconstruct the original signal to 30 db SNR, but it requires only a small number of spikes per pulse. Perceptually, the code contains only very subtle distortions of the original signal. Although the two are distinguishable, it is difficult to judge whether the original or reconstructed sound is the "true" signal. This demonstrates the efficiency of the spikegram with respect to nonstationary harmonic structures.

One of the particularly desirable properties of the model is the efficient coding of transients, where precise temporal coding is most important. Dis-

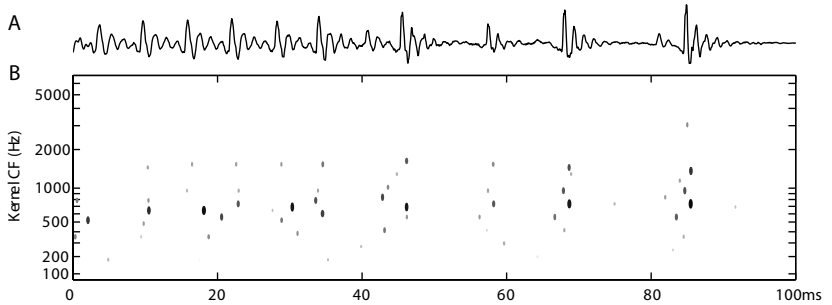

Figure 8: A spikegram shows time-relative coding in the syllable /el/.

tinguishing consonants in continuous speech, for example, requires the detection of rapid, broadband transients. Figure 9 presents an example of a transient, /t/ sound from the word *Vietnamese*. The input signal in Figure 9A consists of an extended vowel with an embedded transient. The entire signal was encoded using matching pursuit and then optimized. The small set of spikes corresponding to the transient /t/ sound is easily distinguishable from the other spikes. We were able to segment them by hand from the rest of the representation. In Figure 9B, a spike code of only four events (magnified in the inset) is sufficient to encode the transient (see Figure 9A, Reconstruction), leaving only the vowel component (see Figure 9A, Residual). These two events are precise in time to within 0.06 msec (the sampling rate of the signal.) In a spectrogram (see Figure 9C), the same transient is smeared over 10 msec of time and a large region of frequency space. Note that the timescale (x-axis) is the same in Figures 9A, 9B, and 9C. Although this particular consonant is unusually short in duration, this example still illustrates the precise timing and localization achievable with a spike code.

5 Coding Efficiency

The previous section shows that spike coding allows time-relative representation of sound structure, but it is not yet demonstrated that this produces an efficient representation of the signal. A complete evaluation of the spike code model requires some objective measure by which to compare the various algorithms and to compare the model against other representational techniques. Shannon's rate distortion theory offers an objective measure of coding efficiency that is widely used in signal coding research (Shannon, 1948). The idea is to vary the rate of a code (typically in terms of bits per second) while measuring the effect that has on some measure distortion, such as mean squared error. For the comparisons between spike coding algorithms, we can start simply, varying the rate in terms of spikes per second and measuring the fidelity of the code (the inverse of distortion) in terms

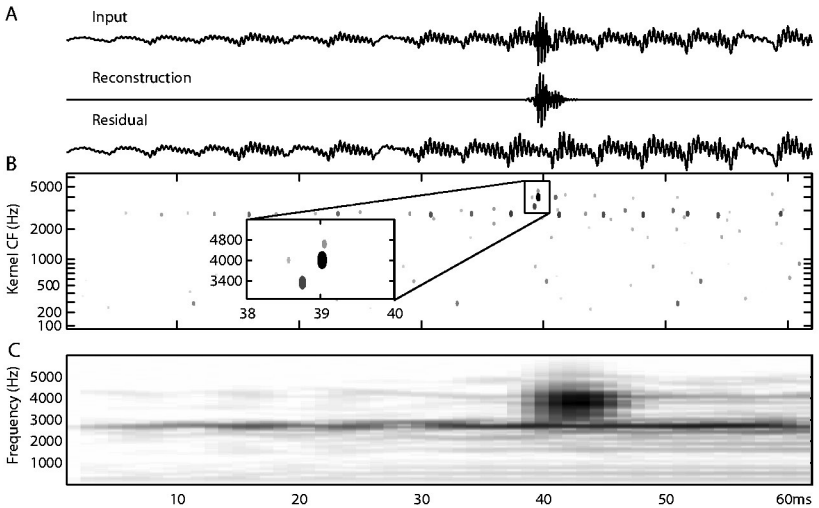

Figure 9: Efficient representation of a speech consonant. An input signal (A, input) is represented as both a spikegram (B) and spectrogram (C). We can reconstruct the signal based on only the four spikes shown in the inset (B) to segment the /t/ sound from the vowel (A, reconstruction and residual).

of dB SNR. We will then address the issue of quantifying coding efficiency more precisely in terms of bits.

5.1 Coding Efficiency in Terms of Spikes. The within-model comparison of encoding algorithms will focus on filter threshold, matching pursuit, and the hybrids, allowing four different algorithm combinations. To generate a measure of coding efficiency, each algorithm was used to encode a large corpus of short (50–200 msec) segments of speech (Garofolo et al., 1990), music, and other "natural" sounds (e.g., birdsong, music and environmental sounds) at various spike rates. All of the stimuli were in .wav format, sampled at 16 kHz, and band-limited to 80 to 6000 Hz. The leading and trailing portions of the stimuli were multiplied by half-Hanning windows to prevent edge artifacts.

The left panel in Figure 10 shows a simplified rate fidelity curve for each algorithm across the entire database. The x-axis indicates the spike rate on a log scale. The y-axis indicates the fidelity in terms of the mean SNR. The computationally simple filter threshold produces a highly redundant, relatively low-fidelity code (less than 11 dB SNR 10,000 spikes/second.) Its decomposition overrepresents large-amplitude components of signals while devoting relatively few spikes to lower-amplitude components, which may represent distinct sound structure. At large spike rates (more than

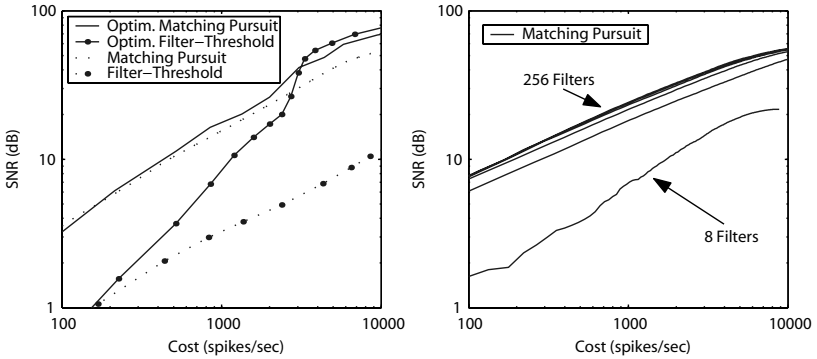

Figure 10: Spike coding efficiency curves. Plotted on the left is the increase in fidelity with increasing spike rate for the four spike code algorithms. Plotted on the right is the increase in fidelity with increasing spike rate for matching pursuit using different numbers of gammatones in the filter bank (8, 16, 32, 64, 128, 256).

50,000 spikes/second) a mean reconstruction fidelity of about 20 dB SNR is possible.

Codes produced by matching pursuit have much greater fidelity at all rates than those from filter threshold. By decomposing signals into sets of orthogonal components, it eliminates all spectral and temporal redundancies in its representation. At low spike rates, where the code tends to represent nonoverlapping signal structures, matching pursuit appears to generate a near-optimal code. However, at higher rates, the fidelity of the greedy algorithm tends to reach a ceiling, with a mean SNR of less than 60 dB.

Although the filter threshold produces a relatively inefficient code, MAP optimization of its spike amplitudes can result in an extremely high-fidelity representation. To achieve this, the filter threshold must generate a set of spike times sufficient to span the signal space. There are two factors responsible for the increased efficiency of optimized codes. First, the gradient-descent optimization eliminates redundant spikes by driving spike amplitudes to zero in accordance with the sparse prior. Second, in minimizing the expected error, the optimization makes use of correlations between kernels, subtly adjusting spike amplitudes rather than eliminating them. The relative contribution of each factor is largely dependent on the amount of noise, $\varepsilon(t)$, assumed in the model. Low-noise models preserve spikes and rely on precise signal fitting; high-noise models eliminate most spikes (pushing their amplitudes to zero).

While optimization of the filter threshold codes greatly increases their efficiency, further optimization of matching-pursuit spike amplitudes leads to relatively small increases in efficiency at lower bit rates. Equation 3.4

shows that the algorithm decomposes signals into orthogonal components. As such, increases in efficiency cannot result from redundancy reduction thorough spike elimination. Instead, spike amplitudes are adjusted to make use of correlations between kernels. Using these correlations can prevent the ceilings in coding efficiency found in the raw matching pursuit codes at higher spike rates.

The right panel in Figure 10 plots the rate fidelity curves for matching pursuit using different numbers of gammatones. ERB filter banks of 8, 16, 32, 64, 128, and 256 gammatones were produced using Slaney's Matlab toolbox (Slaney, 1998). Generating each filter bank separately rather than using subsets of some fixed large filter bank allows each component filter's bandwidth to vary and better tile the frequency space. The sound ensemble (the same used in the between-algorithms comparison) was encoded with each kernel set while varying the spike rate. The resulting curves show a clear relation between coding efficiency and filter bank size. The progression of lowest to highest curves on the plot exactly follows the number of kernel functions used. Although efficiency increases monotonically with the number of size of the filter bank (i.e., number of kernels), the relative gain beyond 64 is extremely small. Additionally, as shown by example earlier in Figure 6, codes produced by 32 or fewer kernels lack a sparse temporal structure in addition to their relatively course spectral representation.

Figure 10 shows that matching pursuit is highly efficient at low spike rates but is surpassed by the hybrid optimized filter threshold beyond about 25 dB SNR. The reason for this inefficiency at high rates is that matching pursuit often fails to accurately describe true signal structure (Gribonval, Depalle, Rodet, Bacryf, & Mallatt, 1994; Goodwin & Vetterli, 1999). Because each component of its code is constrained to be orthogonal (by equation 3.4); see also Mallat & Zhang, 1993), it cannot capture independent signal structure, which closely overlaps in time-frequency space. To test matching pursuit's ability to separate overlapping signal structure, a test signal was created by summing pairs of gammatone kernels separated systematically in time (first third of the signal) and in frequency (latter two-thirds of the signal). The spikegrams in Figure 11 show the potential for time-frequency separability using both matching pursuit and MAP optimization. The ideal sparse representation would consist of pairs of spikes at each of the signal "events" except in the two instances where the kernels perfectly overlap. The MAP optimization algorithm generates just such an encoding (top panel). Looking closely at the representation of one pair of clicks separated 20 msec (top panel, inset), it is clear that two independent events have been coded (allowing perfect reconstruction).

In contrast, matching pursuit cannot separate kernels that closely overlap in time and frequency (bottom panel, around 400 and 3600 msec). Matching pursuit's representation of the same 20 msec separated click pair (bottom panel, inset) is clearly very different from the optimal. On the first iteration of the algorithm, it selects a kernel that is lower frequency than the gam-

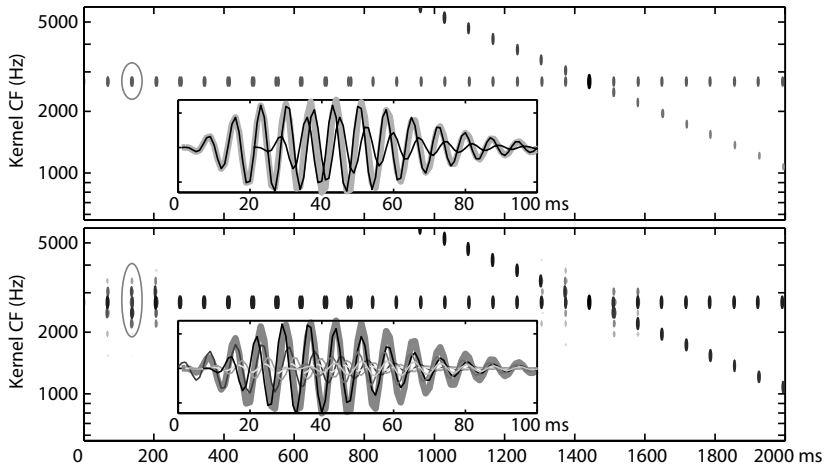

Figure 11: Spikegrams for a signal made of overlapping gammatones. (Top) MAP optimization finds the underlying structure. An example from one pair of clicks (circled) is shown in the inset. The thick gray curve shows the signal, two approximately 2.8 kHz gammatones separated by 20 msec. The thin dark lines are the kernels found by MAP optimization. (Bottom) Matching pursuit cannot separate kernels that closely overlap in time frequency. Given the same two clicks, it generates a single high-amplitude event centered between the chirps and numerous low amplitude events.

matones used to make the signal and centers it between them in time. This means that the representation underestimates the frequency and describes an event when none actually took place. To compensate for this inaccuracy, a large number of additional, low-amplitude kernels are selected on subsequent iterations. With six spikes, it still produces only a 17 dB SNR representation. Nonetheless, the first spike is the single best choice to reduce the residual power. As such, matching pursuit is extremely efficient at low rates. The signal structure initially encoded is typically well separated in time. For example, Figure 5 showed that the initial encoding involves structure that was largely well separated in time and frequency. Tracking the decomposition spike by spike, the kernel that satisfies equation 3.5 on each iteration tends to occur once at each glottal pulse, capturing the largest amount of signal structure possible with a single spike, before returning to encode residual local structure around the same time point. This efficient, low-rate representation can also be generated by MAP optimization by assuming an appropriate degree of noise in the model, but this parameter can be difficult to determine a priori, and the algorithm is much slower.

5.2 Coding Efficiency in Terms of Bits. The sparse, shiftable kernel model and a set of algorithms for spike coding have been described in some detail. We now want to quantify the coding efficiency in bits so as to evaluate the model objectively and compare it quantitatively to other signal representations. Rate fidelity again provides a useful objective measure for comparison. Computing the rate-fidelity curves begins with the associated pairs of coefficients and time values, $\{s_i^m, \tau_i^m\}$, which are initially stored as double-precision variables. Storing the original time values referenced to the start of the signal is costly because their range can be arbitrarily large and the distribution of time points is essentially uniform. Storing only the time since the last spike, $\delta\tau_i^m$, greatly restricts the range and produces a variable that approximately follows a gamma distribution.

Rate fidelity curves are generated by varying the precision of the code, $\{s_i^m, \delta\tau_i^m\}$, and computing the resulting fidelity through reconstruction. A uniform quantizer is used to vary the precision of the code between 1 and 16 bits. At all levels of precision, the bin widths for quantization are selected so that equal numbers of values fall in each bin. All s_i^m or $\delta\tau_i^m$ that fall within a bin are recoded to have the same value. We use the mean of the unquantized values that fell within the bin. s_i^m and $\delta\tau_i^m$ are quantized independently. We found that $\delta\tau_i^m$ for gammatones with low center frequencies required much less precision than for higher-frequency gammatones. Accordingly, temporal precision for the kernel functions was normalized with respect to its wavelength so that the same error during quantization would produce the same relative displacement with respect to a kernel's wavelength.

Treating the quantized values as samples from a random variable, we estimate a code's entropy (bits/coefficient) from histograms of the values. Rate is then the product of the estimated entropy of the quantized variables and the number of coefficients per second for a given signal. At each level of precision, the signal is reconstructed based on the quantized values, and an SNR for the code is computed. This process was repeated across a set of signals, and the results were averaged to produce rate fidelity curves. Matching pursuit was used to estimate the $\{s_i^m, \delta\tau_i^m\}$ pairs for these rate fidelity curves.

Coding efficiency can be measured in nearly identical fashion for other signal representation. In addition to spike codes, rate fidelity curves were generated for four other signal representation methods using the same set of sounds. The two most common methods for signal processing are Fourier and wavelet transform. Fourier coefficients were obtained for each signal via fast Fourier transform. The real and imaginary parts were quantized independently, and the rate was based on the estimated entropy of the quantized coefficients. Reconstruction was simply an inverse Fourier transform on the quantized coefficients. Similarly, coding efficiency using eighth-order Daubechies wavelets was estimated using Matlab's discrete wavelet transform and inverse wavelet transform functions. As a baseline for comparison, rate fidelity curves were produced for the waveform of time-varying am-

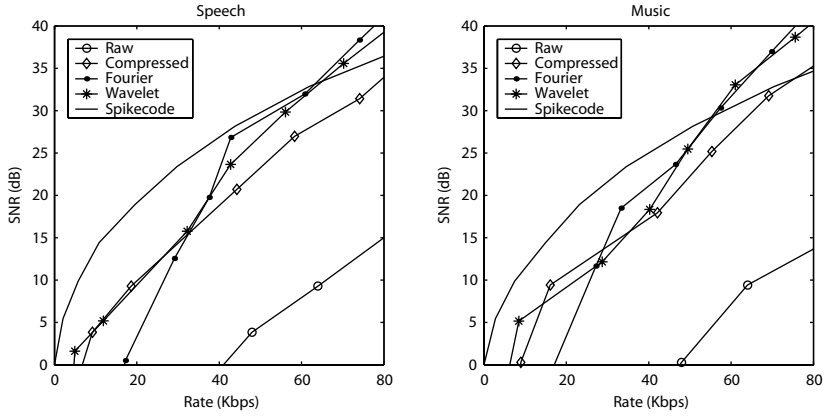

Figure 12: Rate fidelity curves for the raw (uncompressed) time-varying signals, compressed signals, Fourier transform, discrete wavelet transform, and spike coding using matching pursuit for speech (left) and music (right).

plitude values. The fidelity was determined by quantizing the amplitude values and computing the SNR directly from the resulting signal. Two different methods were used to determine the rate. In the compressed case, the rate was based on the estimated entropy of the quantized values, just as above. In the raw case, the cost of each coefficient was equal to the quantization level rather than the estimated entropy.

Figure 12 shows the rate fidelity curves calculated for two classes of sounds: speech (from the TIMIT database; Garofolo et al., 1990) and music (a wide variety of instrumental pieces collected from the Internet.) Compressed, Fourier, wavelet, and spike representations are all far more efficient than the raw signal over the range of rate fidelity parameters we analyzed. At rates below 40 Kbps, spike codes produce the most efficient representations of both speech and music. For example, between 10 and 20 Kbps, the fidelity of the spike representation of speech is approximately twice that of either Fourier of wavelet transformations. At higher bit rates (above 60 Kbps), the Fourier and wavelet representations produce much higher rate fidelity curves than either spike codes or compressed signals. At high rates (off the scale shown), the wavelet representation is most efficient followed by Fourier. In Figure 10, the filter threshold hybrid code also exceeded the matching pursuit–generated code, in this case beyond 25 dB SNR. This offers some evidence that optimized spike codes might exceed the efficiency of Fourier and wavelets throughout the range of perceptually discriminable fidelities.

6 Discussion

We have presented a theoretically motivated model for sound coding in which the computational goal is to form an efficient, time-relative representation of the time-varying-amplitude signal. Signals are represented by decomposing them into a minimal number of gammatone acoustic events, each with an associated time and amplitude. We have shown that this yields efficient representations of transient signals, allowing highly precise representations of sound onset. We have also shown that over a large range of fidelities, this representation is more efficient than Fourier or wavelet representations.

The research presented here adds to the literature on efficient sound representation in several ways. It expands on the original spike coding model (Lewicki & Sejnowski, 1999; Lewicki, 2002b) by presenting a new encoding algorithm, objectively measuring efficiency of spike codes, and comparing them against Fourier and wavelets, and providing a detailed analysis of the representation with specific examples of its strengths and weaknesses. This work also addresses a number of issues relating to matching pursuit. Goodwin and Vetterling (1999) proposed using damped sinusoids with matching pursuit to avoid problems arising from using symmetric wavelets to represent sound. We demonstrated the potential of using a physiologically derived set of gammatone kernels for sound representation rather than abstract sinusoids. The success of this approach lends support to the idea that these functions are adapted to the statistics of natural sounds (Lewicki, 2002a). Additionally, we presented an analysis of the relationship between matching pursuit codes and those derived via MAP optimization. Finally, we used rate distortion to compare matching pursuit against other approximative spike coding algorithms.

Another view of these results is that we have shown a method for achieving significantly greater coding efficiency using what is in effect an overcomplete basis. A motivating goal for so-called overcomplete representations or atomic decompositions with overcomplete dictionaries (Mallat & Zhang, 1993; Chen, Donoho, & Saunders, 1996; Goodwin & Vetterli, 1999; Lewicki & Sejnowski, 2000) is to draw from a rich vocabulary of signal descriptors to find compact signal representations. The method we have described here can be viewed in these terms, because the kernel functions $\phi_m(t)$ can be arbitrarily numerous in their center frequencies and can be placed at arbitrary points in time (Lewicki & Sejnowski, 1999; Lewicki, 2002b). In previous studies, it has been difficult to show that the sparse representations made possible with overcomplete representations could actually yield demonstrably more efficient codes. The general problem is that the added cost of coding an overcomplete set of coefficients often outweighs any gain achieved in representation sparseness (Lewicki & Olshausen, 1999; Lewicki & Sejnowski, 2000). The advantage offered by the approach here is that it is not necessary for the code to describe each implicit coefficient (i.e., at every

sample position). Instead, it is sufficient to describe the time intervals between the spikes. This yielded a much more efficient code for two reasons. The first is that the use of gammatones is well matched to the underlying structure of speech and music. The second is that the matching-pursuit algorithm achieves highly sparse representations. This is crucial, because optimization with filter threshold yields a highly redundant code that does not show increased coding efficiency. It is possible that improvements in either the kernel functions or the encoding algorithms could yield spike codes with even greater efficiencies, and thus provide improved methods for representing the underlying signal structure.

We will address a few assumptions made in our model. The first is that we have assumed an explicit generative model and assessed performance by computing the fidelity of the reconstructed signal. It is possible, for example, that the simple filter threshold algorithm does achieve a high-fidelity encoding of the signal, but we do not know how to reconstruct it. It was our motivation, however, to develop efficient codes that can reveal the intrinsic information of a signal. Our assumption of a linear superposition of kernel functions is a simple means of evaluating the degree of redundancy in the representation.

Another assumption concerns our choice of distortion measure for evaluating coding efficiency. The reported signal-to-noise values are based on the sum-squared error between the original signal and the reconstruction. It is well known that this measure of distortion does not agree closely with human perception. For example, gaussian white noise signals would tend to appear very dissimilar using this measure, yet they all sound the same to a listener. Although a perceptual distortion measure will offer greater insight into the relationship between the algorithms described here and human perception, the sum-squared error is adequate to show their relative coding efficiency as general signal representations.

Beyond the questions of general signal processing, we are particularly interested in the use of spike coding algorithms as models of neural auditory processing. The analog amplitude values in the model can also be interpreted as representing a local population of auditory nerve spikes. As a theoretic model of auditory coding, this posits that the purpose of the (binary) spikes at the auditory nerve is to encode as accurately as possible the temporal position and amplitude of underlying acoustic features, which are compactly described by gammatones. Analog spikes are a useful theoretical abstraction, and it is simple to convert these individual spikes into a population of probabilistically firing, binary units that can carry the same information. Some possible neural network architectures along these lines are described in Lewicki (2002b).

One potential concern for describing the cochlear processing with the model presented here is that this model lacks the explicit representation of the nonlinearities found in detailed cochlear models. Our goal, however, was to construct a coding algorithm motivated by higher-level principles.

We have not yet investigated whether any cochlear nonlinearities can be interpreted in terms of the assumed computational objective. For example, matching pursuit and gradient optimization can be viewed as means of selecting out a subset of the spikes that removes interspike redundancy and yields an efficient representation. This could offer a novel theoretical interpretation of two-tone inhibition.

The addition of nonlinearities may still not help describe the intrinsic coding processes of the system. For example, the model by Yang, Wang, and Shamma (1992) is spike based and contains numerous biophysically motivated nonlinearities, but the fidelity of their reconstructions falls in the same range as filter threshold and is much lower than that of matching pursuit or the hybrids. They reported SNR ranging from 11 to 25 dB, depending on the extent of processing. This underscores the complexity of the signal encoding problem solved by the peripheral auditory system.

We have sought to develop an algorithm that extracts the intrinsic information in a signal from the stream of observable information. As was stated earlier, information reduction can be achieved by either selecting only the desired information or removing redundancy. The choice of kernel functions biases our model to "select" certain types of signal structure over others. Our choice of a gammatone filter bank, a highly efficient basis for natural sounds, biases the model to select the underlying structures of natural sounds. The algorithms for generating spike codes take the approach of redundancy reduction. Reducing the temporal correlations shows promise for yielding better methods to extract intrinsic structure from raw acoustic signals.

Acknowledgments

E.S. was supported by NIH training grant MH19983. This material is based on work supported by the National Science Foundation under grant 0238351 to M.L.

References

Baumgarte, F. (2002). Improved audio coding using a psychoacoustic model based on a cochlear filter bank. *IEEE Transactions on Speech and Audio Processing, 10*(7), 495–503.

Blauert, J. (1997). *Spatial hearing* (rev. ed.). Cambridge, MA: MIT Press.

Chen, S. S., Donoho, D. L., & Saunders, M. A. (1996). A composite model of the auditory periphery for the processing of speech. *SIAM Review, 43*(1), 129–159.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., Pallett, D. S., Dahlgren, N. L., & Zue, V. (1990). TIMIT acoustic-phonetic continuous speech corpus. Audio CD.

Gitza, O. (1988). Temporal non-place information in the auditory-nerve firing patterns as a front-end for speech recognition in noisy environments. *J. Phonetics, 16*, 109–124.

Goodwin, M. M., & Vetterli, M. (1999). Matching pursuit and atomic signal models based on recursive filter banks. *IEEE Transactions on Signal Processing, 47*(7), 1890–1902.

Gribonval, R., Depalle, P., Rodet, X., Bacryf, E., & Mallatt, S. (1994). Sound signals decomposition using a high resolution matching pursuit. In *Proceedings of International Computer Music Conference* (pp. 293–296).

Lewicki, M. S. (2002a). Efficient coding of natural sounds. *Nature Neuroscience, 5*(4), 356–363.

Lewicki, M. S. (2002b). Efficient coding of time-varying patterns using a spiking population code. In R. P. N. Rao, B. A. Olshausen, & M. S. Lewicki (Eds.), *Probabilistic models of the brain: Perception and neural function* (pp. 241–255). Cambridge, MA: MIT Press.

Lewicki, M. S., & Olshausen, B. A. (1999). A probabilistic framework for the adaptation and comparison of image codes. *Journal of the Optical Society of America, 16*(7), 1587–1601.

Lewicki, M. S., & Sejnowski, T. J. (1999). Coding time-varying signals using sparse, shift-invariant representations. In M. S. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in neural information processing systems, 11* (pp. 730–736). Cambridge, MA: MIT Press.

Lewicki, M. S., & Sejnowski, T. J. (2000). Learning overcomplete representations. *Neural Computation, 12*, 337–365.

Liberman, A., Delattre, P., & Cooper, F. (1958). Some cues for the distinction between voiced and voiceless stops in initial position. *Language and Speech, 1*, 905–917.

Lyon, R. F. (1982). A computational model of filtering, detection and compression in the cochlea. In *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing* (pp. 1282–1285).

Mallat, S. G., & Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing, 41*(12), 3397–3415.

Oertel, D. (1999). The role of timing in the brain stem auditory nuclei of vertebrates. *Annual Review of Physiology, 61*, 497–519.

Olshausen, B. A. (2002). Sparse codes and spikes. In R. P. N. Rao, B. A. Olshausen, & M. S. Lewicki (Eds.), *Probabilistic models of the brain: Perception and neural function* (pp. 257–272). Cambridge, MA: MIT Press.

Patterson, R., Holdsworth, J., Nimo-Smith, I., & Rice, P. (1988). *Implementing a gammatone filterbank* (Tech. Rep. No. 2341). Cambridge, UK: MRC Applied Psychology Unit.

Rabiner, L. R., & Levinson, S. E. (1981). Isolated and connected word recognition: Theory and selected applications. *IEEE Trans. Communications, COM-25*(5), 621–659.

Shamma, S. (1985). Speech processing in the auditory system: II. Lateral inhibition and the central processing of speech-evoked activity in the auditory nerve. *Journal of the Acoustical Society of America, 78*, 1622–1632.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal, 27*, 379–423, 623–656.

Slaney, M. (1998). *Auditory toolbox* (Tech. Rep. No. 1998-010). Palo Alto, CA: Interval Research Corporation.

Slaney, M., & Lyon, R. F. (1993). On the importance of time—A temporal representaton of sound. In M. Cooke, S. Beet, & M. Crawford (Eds.), *Visual representations of speech signals* (pp. 95–116). New York: Wiley.

Yang, X., Wang, K., & Shamma, S. (1992). Auditory representations of acoustic signals. *IEEE Trans. Inf. Theory, 38*, 824–839.

Received March 11, 2004; accepted June 4, 2004.