

## **Non-linear nature of the spike coding algorithm**

Non-linear coding is necessary when the fundamental structure of the signal lies in a non-linear subspace, i.e. one that is not a linear projection of the original subspace. The smaller this subspace in relation to the dimensionality of the original signal space, the greater the potential coding efficiency. In the case of sounds, if we imagine an underlying generative model in which acoustic events occur at random (and sparse) times, then coding is the inference process of recovering the times and waveforms of the original acoustic events. If the acoustic events are rare, (so that they rarely overlap), and the noise low, this is a relatively easy non-linear inference problem. A simple filter and threshold algorithm will suffice to recover the true underlying structure of the acoustic signal. As the events become more common or the noise becomes higher, the inference problem becomes harder.

The real acoustic environment is not a mixture of discrete events, but, as the spike coding model demonstrates, it can be efficiently described that way. In this case, the acoustic events are not actually present in the external environment, but instead they acoustic features that occur sparsely in time. The actual degree of sparseness can be measured by making histograms of the spike time intervals. This analysis shows that spikes follow a gamma distribution with an average mode of about 10 milliseconds (the actual distribution varies as a function of the kernel length). In 10 milliseconds there are 160 samples at 16 kHz, so in these discrete terms, each spike is being placed on average in 1 out of 160 possible positions. The spike coding algorithm fundamentally non-linear, because it must determine which of these positions is best. Furthermore, the position of each spike must be coordinated with other spikes so that the information in the waveform is represented non-redundantly. The spike coding algorithm described in the main text is an efficient method of finding an approximate solution to this problem. Quasi-linear models such as integrate and fire (or convolution followed by probabilistic spiking) lead to redundancy in both time and across kernels and thus do not yield efficient codes. It has been shown that this simple spike coding model yields a typical coding fidelity of less than 10 dB SNR. Purely linear models, such as a bank of filters, are even less efficient, because they transform a single analog waveform into a set of waveforms.

## **Comparison of learned kernels and auditory revcor filters**

Auditory revcor filters are estimates of the impulse response functions of auditory nerves, and are thus a first-order characterization of the auditory nerve response. Are these comparable to the spike code kernels? The goal the spike coding algorithm is to determine the precise temporal locations of the underlying acoustic features represented by the kernels. The first step in this algorithm is to convolve (or filter) the signal with kernels. Convolution is also the first step in the implicit model used in reverse correlation. The two models differ in how the analog filter output is converted to a spike train. In the case of the revcor model, spiking is simply a probabilistic function of the filter output. In the case of spike coding, the spikes are chosen so as to represent the signal with max-

imal efficiency. In both cases, the kernels (i.e. the acoustic features) can be recovered by reverse correlation (spike-triggered averaging) in response to Gaussian noise, because the resulting spike trains are uncorrelated. For the revcor spiking model, the spike times are uncorrelated, because the input signal is uncorrelated and probabilistic spiking does not introduce any correlations. In the case of the spike coding algorithm, the spikes are uncorrelated because the code is non-redundant (both across time for an individual kernel and across kernel functions, although the later is irrelevant for purposes of reverse correlation). Thus, for each of these models, we would expect reverse correlation to recover the underlying acoustic features. Supplementary figure S3 confirms that this is the case for the spike coding model.

What about the case of an unknown non-linear coding algorithm (such as the peripheral auditory system)? The revcor filters themselves do not directly tell us whether the auditory spike code is efficient – that would require an analysis of the auditory spike trains in both time and across nerve fibers and a more detailed model of how the acoustic waveform is encoded by binary spikes – but the close match between the theoretically ideal kernels and the auditory revcor filters is consistent with what would be expected in a system that was forming an efficient spike code.