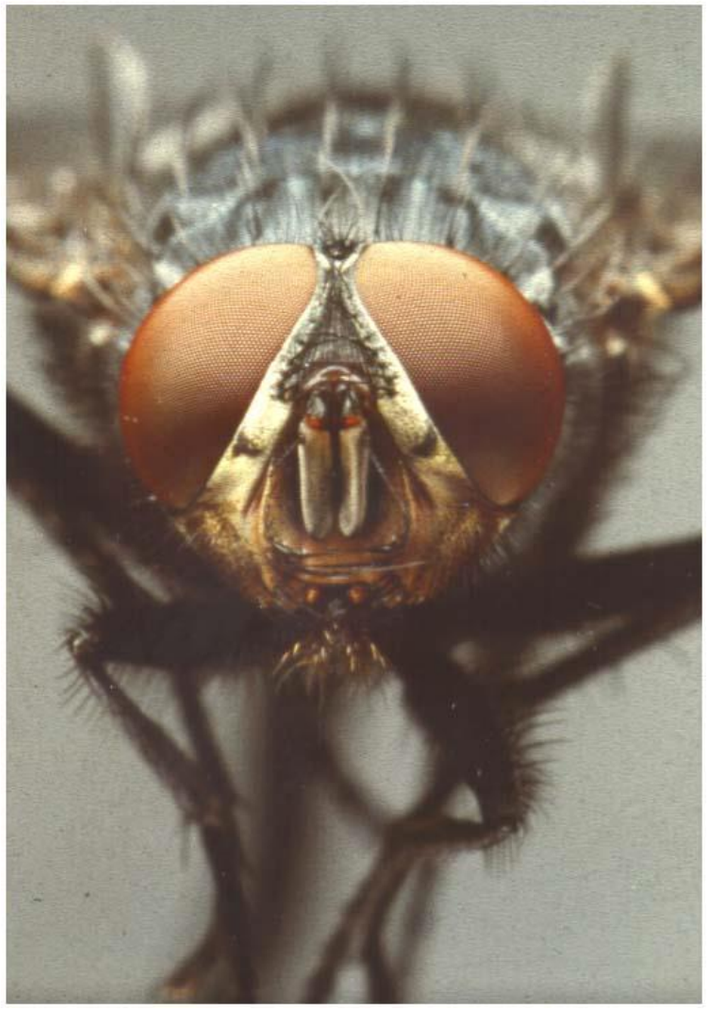
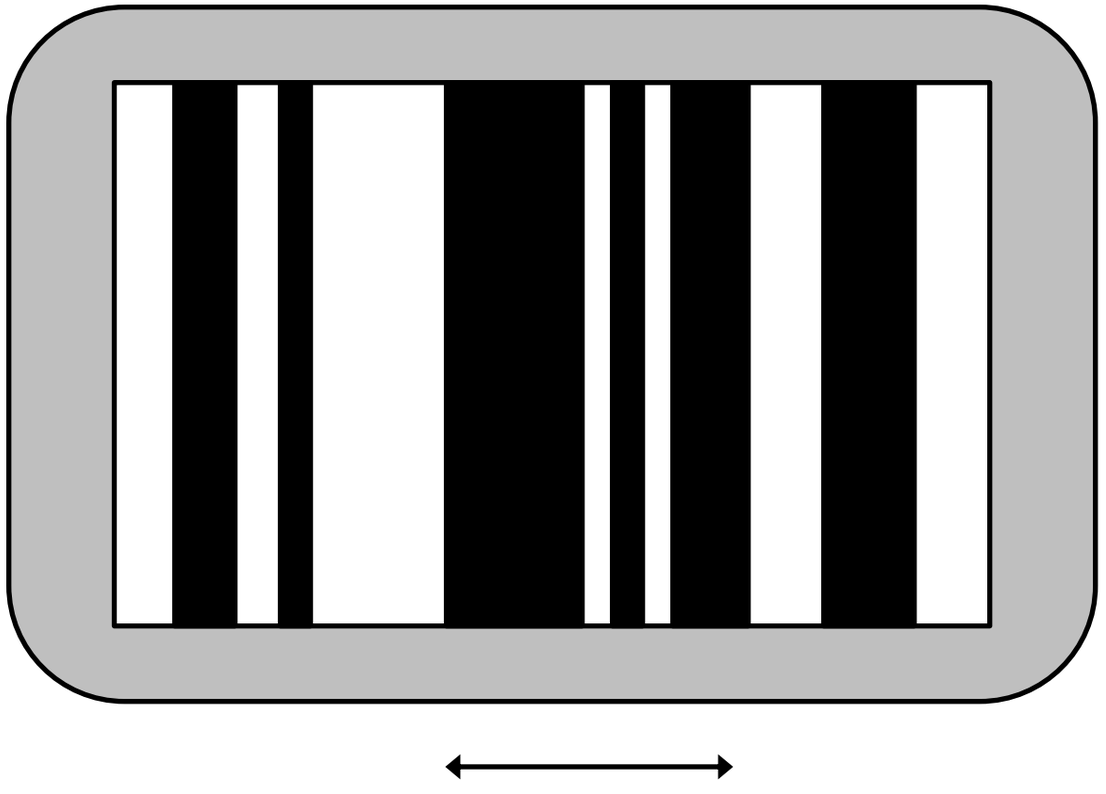
Characterizing coding in the H1 neuron

The data set you were provided with is recorded from the blowfly H1 neuron. In this experiment, an electrode is placed in the fly's brain– in a region called the lobula plate– where there is a set of large neurons that encode information about wide-field motion, motion of the entire visual field. We are recording from one of these neurons, H1, which encodes horizontal motion.

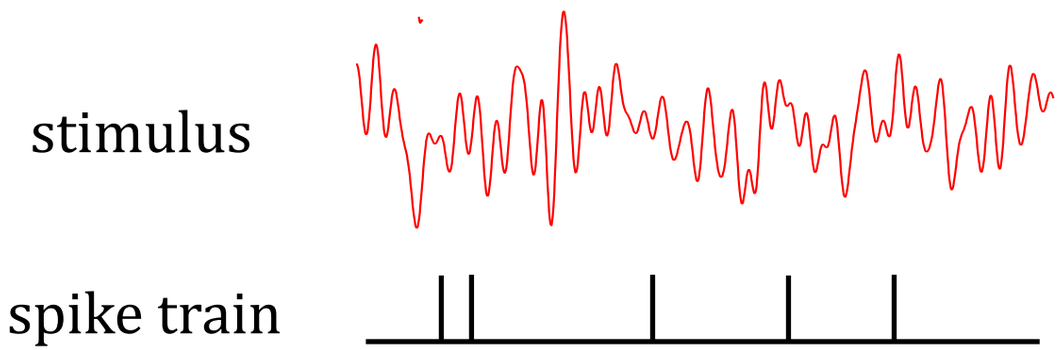


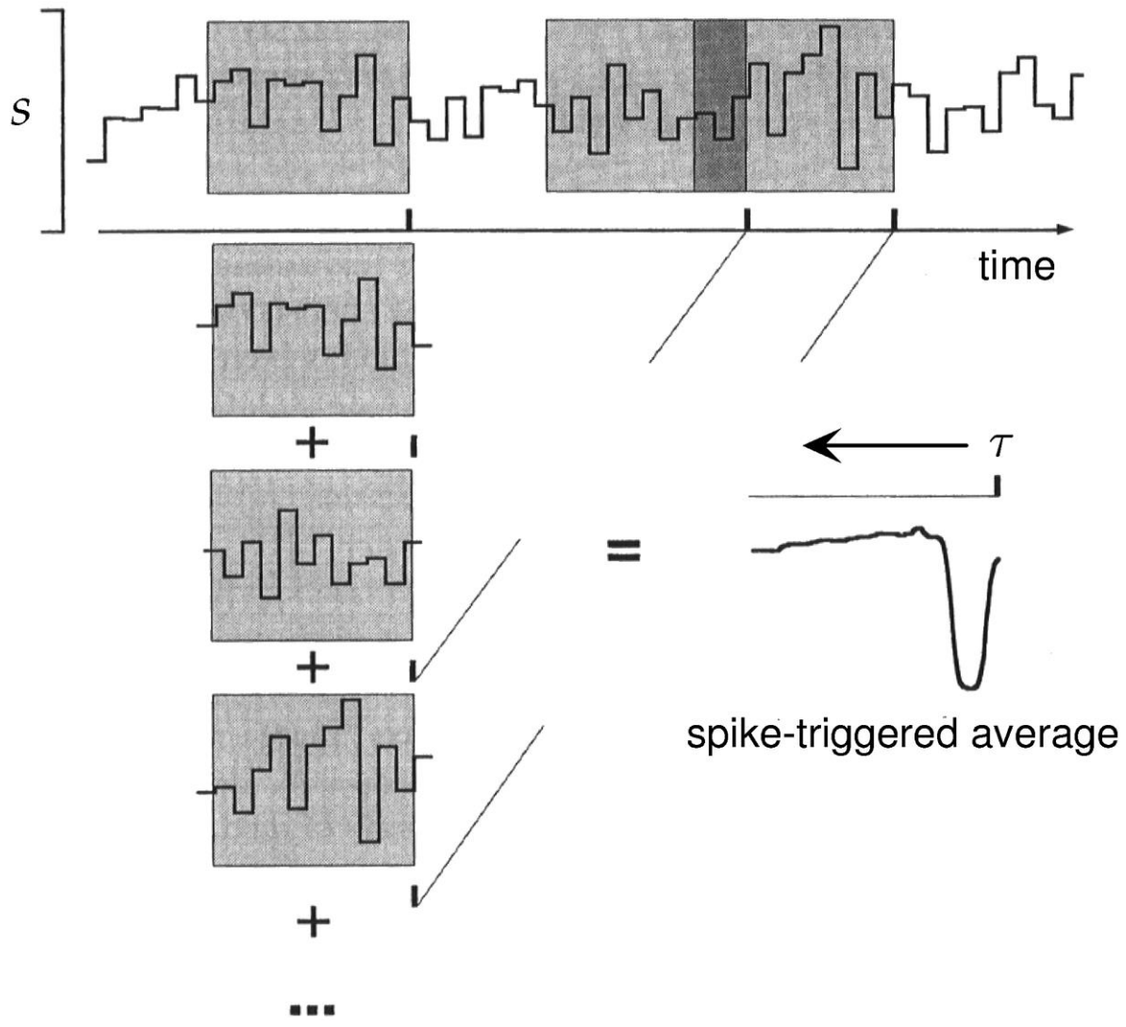


The stimulus you have been provided with is the *velocity* of a random bar pattern like this that moves back and forth on a screen in front of the fly. The stimulus is created by choosing a random velocity (either positive or negative) every 2 milliseconds. The screen then moves with that velocity for 2 ms. This is an example of a "white noise" stimulus.

Reverse correlation: the spike-triggered average

Your job is to calculate the spike-triggered average. The spike-triggered average is the *average stimulus that precedes a spike.* To calculate it, you need to know the spike times, aligned with the stimulus.





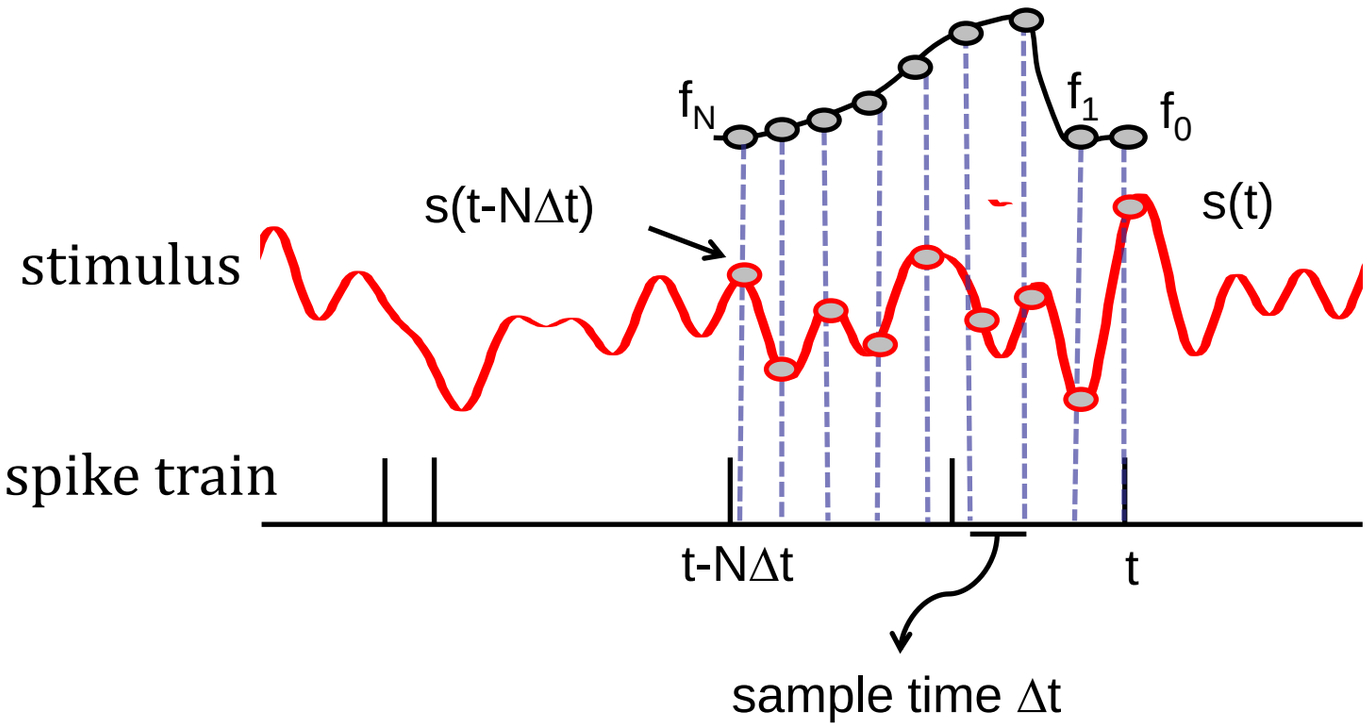
The spike train data that you were given has been simplified. In the experiment, the voltage was recorded, but we have first downsampled the signal to one value every $\Delta t = 2$ ms, to match up with the stimulus sampling, and the voltage has been replaced with simply a 1 in the time bins where there was a spike, and a zero otherwise.

To compute the spike triggered average, you need to collect chunks of the stimulus, let's say $T = N \Delta t$ ms long, preceding each spike, and to average them, as in the figure to the right. Note that you are not averaging over time– it's exactly the timecourse of the stimulus that you are interested in-- but over spikes.
So if you have M spikes, you will have M stimulus chunks of T ms each; and your average will be N time samples long.

What do we learn from the spike-triggered average?

What is the spike-triggered average useful for? We can use it to predict the times of spikes given a new random piece of stimulus. Let's first approximate our response to depend *linearly* on the stimulus. What does it mean to be linear? $r = a\ s$, where a is a constant.

But we need to consider a bit more general notion of linearity. The probability of a spike does not depend on the stimulus at the exact same time– it can't, because there is some finite response delay, so it has to depend on the stimulus at some time back in the past. But the fact is the probability of a spike depends not just on one time back, but the entire sequence of stimulus values over the recent past. If it depends on the stimulus linearly, then the response is approximately a *weighted sum* of those recent stimulus values. In the diagram, we look backward in time from time t, moving in steps of the sample time, Δt, and multiply together the value of the stimulus at each time step back with a corresponding set of weights which we will call f_i, i ranging from 1 to N. Then we sum up all of those weighted stimulus values to get our *filtered* stimulus.



$$s_f(t) = \sum_{i=0}^{i=N} f_i\, s(t - i\,\Delta\, t)$$

This procedure is called **filtering**, or **convolution**. If we think about that N-point chunk of stimulus s as a vector, it is also equivalent to **projecting** s onto f, interpreted also as an N-dimensional vector.

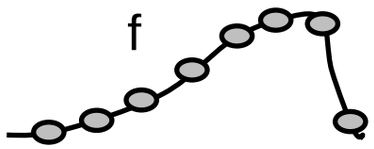
In lectures, we wrote this sum in its integral form; Here the index i is replaced by a time variable τ.

$$s_f(t) = \int s(t-\tau)\, f(\tau)\, d\tau$$

Interpreting features as temporal filters

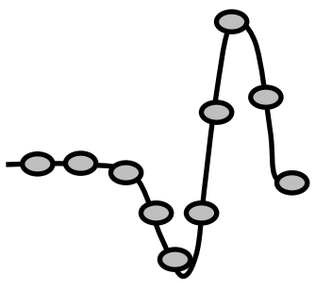
One of the advantages of a spike-triggered average is that by looking at it, we can see what kind of operation is being carried out on the stimulus to trigger that spike. Thinking about it as a filter allows us to understand what transformation of the input drives the neuron to fire.

An example was mentioned in the lecture: the idea that the neuron is making a running average of the input, but with a weight that is decreasing as one goes further back from the time of the spike.



That's pretty similar to what is going on in this example.

What if we instead saw a feature like this?



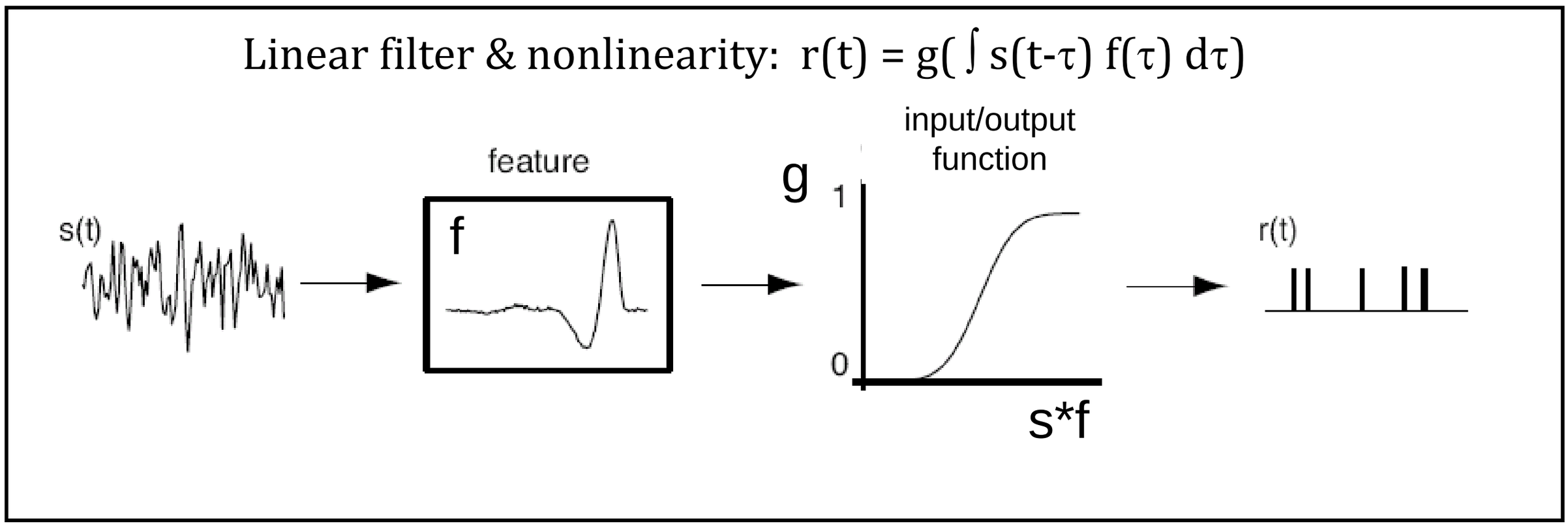
In this case, the neuron responds to a fluctuation that first goes down then goes up: that is, to a local *change* in the stimulus. Seen as a filter, this takes the stimulus at some time back, weights it with a negative weight, and adds to it the more recent stimulus. This is an approximation of a derivative in time. This is what a retinal ganglion cell does, both in space (Mexican hat) and in time.

The linear/nonlinear model

Approximating the rate as this linear filter acting on the stimulus,

$$r(t) \sim \sum_{i=0}^{i=N} f_i \, s(t - i \, \Delta \, t)$$

can sometimes work well, especially if there is a background firing rate, r_0, and r(t) is taken to be the deviation from that mean rate, so that it can go either positive or negative. Generally, though, the model works much better if one includes a nonlinear stage, so that the firing rate is a nonlinear function of the filtered stimulus. Such a nonlinearity can account for the fact that firing rates are necessarily positive, make sure that they saturate at some maximum, and generally allow the response to depend in a more general way on the input strength. The key point is that this procedure separates the identification of the stimulus component that is relevant to the cell (the linear filter, *f*) from the shape of the "tuning curve" relevant to that feature (the function *g*).



Generalizations of the linear/nonlinear model

The key take-home from the remainder of Lecture 2 is that one can go on and generalize this model in a couple of different ways that serve to give increasingly good fit to data. We sketched out the idea of principal component analysis (PCA), which is a method that discovers the "dimensions" of a stimulus that best capture the variation in a set of data. The idea of eigenvectors of a matrix and PCA is likely to arise again in the context of linear networks.

As usual, wikipedia has a good reference on this.