Report: Fitting Sigmoidal Output Nonlinearities to STRF Models

Mike Schachter

December 17, 2010

1 Introduction

We have stimuli and response data from ~600 neurons in MLd (midbrain), OV (thalamus), and higher regions (field L, CM), and wish to find optimal models for their stimulus-response functions. Let $\mathbf{s}_t \in \mathbb{R}^M$ be the stimulus vector at time t, which includes it's spacial and temporal components, and $r(t)$ be the value of the PSTH at time t. We fit STRFs to the stimulus response function, which are linear mappings between the stimulus and response space:

$$\hat{r}(t) = \mathbf{a}^T \mathbf{s}_t + a_0$$

The STRF is given by $\mathbf{a} \in \mathbb{R}^M$, and a_0 is the bias weight. Neurons are not linear, so we make the model more complex by adding a static output nonlinearity.

$$\hat{r}(t) = f(\mathbf{a}^T \mathbf{s}_t + a_0)$$

Here we empirically identify plausible output nonlinearities and use parametric fits and staggard optimization to improve the prediction performance of our models.

2 Methods

2.1 Regions, Stimulus Classes, and Stimulus Preprocessing

30 cells were tested in this report, 10 from each region (MLd, OV, L). For each cell we tested responses to several stimulus classes. First was conspecific birdsong, natural sounds which evoke robust responses in each area. The next was syn-song, which is Gaussian noise filtered to have similar statistics to birdsong. The third was ml-noise, the sound equivalent of pure Gaussian noise. We also compared two stimulus representation methods - the spectrogram and Lyon's cochlear model.

2.2 Probabilistic Relation Between Linear Output and PSTH

Stimuli are random variables drawn from some unknown distribution, implying that the linear output $x = \mathbf{a}^T \mathbf{s}_t + a_0$ is a random variable. We can associate x with a probability density function $P(X = x)$, and values of the PSTH with a probability density function $P(Y = y)$. The output nonlinearity can be related to the conditional density $P(Y|X = x)$ by:

$$f(x) = E[P(Y|X = x)]$$

To produce this distribution, values of the linear output are binned, then the distribution $Y|X$ is constructed from the binned data and visualized to provide intuition as to the shape of the output nonlinearity.

2.3 Sigmoidal Output Nonlinearity: Gompertz Curve

In the results section it's explained that we're guided by the form of $E[P(Y|X = x)]$ to use a parameterized sigmoidal output nonlinearity. An often-used parameterized sigmoid is given by the Gompertz function:

$$f(x; a, b, c) = ae^{be^{cx}}$$

The parameter a sets the saturation value of the sigmoid, which we fix at $a = 1$. The parameter $b < 0$ sets the offset of the sigmoid along the x-axis, the more negative, the further to the right the sigmoid is offset. $c < 0$ sets the slope of the sigmoid, the more negative it becomes, the steeper the slope.

2.4 Fitting Model Parameters with Staggard Optimization

The STRF and output nonlinearity each have paramters which need to be automatically fit using an optimization algorihtm. The error function used is sum-of-squares:

$$E(\mathbf{a}, a, b, c) = \sum_{t=1}^{N} (r(t) - (\mathbf{a}^T \mathbf{s}_t + a_0))^2$$

The addition of nonlinear parameters makes the error surface non-convex and prone to local minima. We avoid minimizing the entire errror function directly, and instead use a staggard approach:

1. Find the STRF by minimizing the error function with respect to $\mathbf{a}$ using a fast linear optimization technique.
2. Use the STRF to project the stimulus into a set of scalar values, represented by the time series $x(t)$.
3. Use constrained nonlinear optimization to fit the parameters $\{a, b, c\}$, by minimizing the sum-of-squares difference between $f(x(t))$ and $r(t)$:

$$E_{NL}(a, b, c) = \sum_{t=1}^{N} (f(x(t)) - r(t))^2$$

4. Find a new STRF by minimizing sum-of-squares between the inverted PSTH and linear model:

$$\mathbf{a} = min_{\mathbf{a}} E_L(\mathbf{a}) = \sum_{t=1}^{N} (f^{-1}(r(t)) - (\mathbf{a}^T \mathbf{s}_t + a_0))^2$$

5. Repeat steps 2-4 until optimal parameters $\{\mathbf{a}, a, b, c\}$ have been found. Here we set the algorithm to stop after 10 iterations, and then choose the best iteration based on the amount of normal mutual information captured by validation data.

3 Results

3.1 Sigmoidal Relationship Between Linear Output and PSTH

For each (cell, stimulus class, preprocessing) combination, we visualized the relationship between linear output and PSTH, given by the functoin $g(x) = E[P(Y|X = x)]$. The shape of $g(x)$ was always of the generalized sigmoid type. Because the mapping was to a PSTH, the range of $g(x)$ was always non-negative, and never above 1. Each $g(x)$ appeared to be well-fit by a Gompertz curve. See figure 1 for two example cells, with spectrograms and the Lyon's model.

3.2 Sigmoidal Output Nonlinearity Improves Performance for Spectrogram Preprocessing

The sigmoidal relationship between the linear output and PSTH suggests that a sigmoid as a good candidate for the output nonlinearity. As described in the Methods section, we used a staggard optimization procedure to determine the parameters for the Gompertz curve that captured the most information between the predicted response and PSTH. Adding a sigmoidal output nonlinearity improved predictions for almost all cells tested, verified using one-sample t-tests on the difference in performance between linear and linear-nonlinear models. Many of the models performed best on the first iteration of the staggard procedure, leaving the linear STRF unchanged.

3.2.1 Region Specific Effects

MLd and Field L had the greatest performance increases, on the order of 10% for training data, and 2-5% for untouched validation data. See figures 2-4 for summary data for each region. MLd and Field L also had steeper gains than OV, evidenced by more negative values for the slope parameter c, averaged across cells. Althogh performance gains were higher for training data than validation data, it's possible that regularization techniques for both the linear and nonlinear parts of the optimization could improve validation data performance.

3.2.2 Performance Gains are a Result of Noise Reduction

The saturating part of the output nonlinearity was rarely reached by the range of scalar outputs. To put it another way, the output nonlinearities, when plotted only in the range of possible linear outputs, looked like exponential functions, not sigmoids. This implies the saturating part of the output nonlinearity did not play a role in improving performance. See figure 5 for some examples of output nonlinearities fit within the range of linear outputs.

Figure 6 illustrates a stereotypical result for the linear-nonlinear models tested. The linear output produces significant noise (spurious low-value predictions) below 0.25. The model fit with a sigmoidal output nonlinearity has less low-amplitude noise, because of the "squashing" effect of the sigmoid when it's value is close to 0. One way to observe this is to subtract the LNL response from the linear response. The mean of this difference was always below 0.25, implying that the main difference in predictions between linear and linear-nonlinear models was a reduction in low-amplitude noise. In the SfN poster presented by Schachter et. al (2010), a static threshold output nonlinearity improved predictions for this same reason.

3.3 Sigmoidal Output Nonlinearities Do Not Improve Lyon's Preprocessing Performance

When the same comparison was made using Lyon's cochlear model preprocessing, there was no statistically significant improvement in model performance. In fact, some regions had a slight decrease in training or validation performance. Because the Lyon's model integrates lateral inhibition and temporal gain control in an effort to remove redundancy and noise, we speculate that the noise-reducing properties of a sigmoidal output nonlinearity are not necessary in this case.

3.4 Lyon's Preprocessing Outperforms LNL Model on Training Data, not Validation

Gill et. al 2006 has already shown that Lyon's model can outperform spectrograms for linear models. We followed up on this result to see if linear-nonlinear models with spectrograms do better against linear models with Lyon's preprocessing. The results are shown in figure 7. Linear models with Lyon's preprocessing outperform linear-nonlinear models with spectrograms on training data by an average of 5%. The result is statistically significant and holds up to a t-test. However, there is no statistically significant difference in predictions between Lyon's preprocessing with linear models and linear-nonlinear models with spectrograms for validation data, at least for the 30 cells tested.

4 Next Steps

4.1 Lyon's Model and Recurrent Neural Networks

The Lyon's cochlear model is intriguing becaues it produces good linear fits without any output nonlinearity, and because it outperforms spectrograms on training data. Perhaps with the proper regularization, prediction performance can be improved further and potential output nonlinearities can be explored in more detail. Lyon's model is implemented as a recurrent nonlinear filter which closely resembles a one-layer recurrent neural network. It has both temporal gain control and lateral inhibitory gain control. We have yet to tease apart the contributions of these types of gain control to performance increases in prediction, but are moving towards doing so now. It may be advantageous to exploit results found in recurrent neural network literature to come up with simplified models of these types of gain control.

4.2 Sigmoidal Output Nonlinearties and Recurrent Neural Networks

A neural network that has FIR filters input on the input layer is called a "time-delay" network (keeping in mind STRFs are FIR filters). Time-delay neural networks have been used since the 90's to reduce the number of parameters to fit. It is very tempting to envision our linear-nonlinear models as single units in a large neural network and treat them as such.

In order to exploit this relationship, the next step could be to fit linear-nonlinear models to all MLd cells, and use their output as a preprocessing stage for an OV cell's STRF. The weights can be trained using the usual STRF-fitting routines, or if all weights are to be adjusted a backpropagation algorithm could potentially be used.

Cells in OV receive feedback from cortical cells in Field L, forming a recurrent neural network. Once an optimal basis for MLd cells is created for fitting STRFs in OV, perhaps it would be advantageous to extend this approach and apply recurrent neural network training algorithms to weightings between Field L cell spike data and OV cell spike data to improve the prediction of OV cell output, and eludicate the function of cortical feedback on thalamic cells.