Characterization of Nonlinear Neuron Responses

Matt Whiteway
Department of Applied Mathematics and Scientific Computing
whit8022@umd.edu

Advisor
Dr. Daniel A. Butts
Neuroscience and Cognitive Science Program
Department of Applied Mathematics and Scientific Computing
Biologicial Sciences Graduate Program
dab@umd.edu

October 4, 2013

Abstract

A common approach to characterizing a sensory neuron's response to stimuli is to use a probabilistic modeling approach. These models use both the stimulus and the neuron's response to the stimulus to estimate the probability of the neuron spiking, given a particular stimulus. This project will investigate some of the well known approaches, starting with simple linear models and progressing to more complex nonlinear models. One technique for fitting the parameters of the model uses the idea of a "linear receptive field" to characterize a feature subspace of the stimulus in which the neuron's response is dependent. Another parameter fitting technique that has been used with success is a likelihood-based method that can lead to efficient model parameter estimation. This project will examine both linear and nonlinear models, and use both the feature subspace technique and the maximum likelihood technique to fit the parameters of the appropriate model.

1 Introduction

A fundamental area of interest in the study of neurons is the relationship between the stimuli and the neural response. This relationship has been modeled using various approaches, from the estimation of the neuron's firing frequency given by the integrate-and-fire model [1] to the set of nonlinear differential equations given by the Hodgkin-Huxley model [2].

More recent approaches have utilized a probabalistic modeling framework for sensory neurons (neurons in the visual or auditory pathways, for example). Since techniques in neuroscience are now able to supply us with increasingly detailed physiological data, and since the nervous system itself is probabalistic, this statistical description of a neuron seems like a natural avenue of exploration [4]. The following sections detail both linear and nonlinear models, as well as two of the techniques used to estimate the parameters of the given models.

2 Linear Models

The goal of the linear models is to represent the selectivity of the neuron's response to particular features of the stimulus. This selectivity means that certain regions of the stimulus space, called the feature subspace or the receptive field, are more important than others in determining the resulting action of the neuron. The linear models operate by projecting the stimulus onto these lower dimensional subspaces, then subsequently mapping this projection nonlinearly into a firing rate. This firing rate is interpreted as a rate parameter for an inhomogeneous Poisson process that will then give us the probability of the neuron spiking [4].

For the remainder of the paper we will assume that we are modeling a sensory neuron in the visual cortex. The stimulus is a single grayscale pixel that stays the same value for a fixed time interval Δ, then changes value instantaneously. This stimulus is represented by a stimulus vector $\bar{s}(t)$, where each entry is the pixel's value for a time duration of Δ. The response data is a list of times that the neuron spiked during presentation of the stimulus.

It should be noted than when describing this model as "linear", we are refering to the manner in which the stimulus is processed before a nonlinear function maps this into the rate parameter. The form of the linear model described above is called the Linear-Nonlinear-Poisson (LNP) model, and is given by the equation

$$r(t) = F(\bar{k} \cdot \bar{s}(t)) \tag{1}$$

where $\bar{k}$ is the linear receptive field, $s\bar{(t)}$ is the stimulus, F is a nonlinear function, and $r(t)$ is the resulting rate parameter. The linear receptive field $\bar{k}$ is what we wish to find, $\bar{s}(t)$ is a portion of the stimulus whose size depends on the size of $\bar{k}$, and F is generally a sigmoidal function that ensures the firing rate is not negative. The next three sections will develop different ways in which to estimate $\bar{k}$ and F, the unknowns of this equation.

2.1 The LNP using Spike-Triggered Average

The first way in which I will be estimating the parameters of the LNP is through a technique called the Spike-Triggered Average (STA). The STA assumes that the neuron's response is completely determined by the stimulus presented during a predetermined time interval in the past, and is defined as the mean stimulus that elicited a spike [7]:

$$STA = \frac{1}{N}\sum_{n=1}^{N} \bar{s}(t_n) \tag{2}$$

where N is the number of spikes elicited by the stimulus and $\bar{s}(t_n)$ is the stimulus that elicited the n^{th} spike. Defined in this way, the STA is a linear receptive field that the neuron is responsive to, and filtering the stimulus through the STA projects the stimulus onto a lower-dimensional subspace. As long as the input stimulus is spherically symmetric, we can use the STA as the estimate for the linear filter $\bar{k}$ in the LNP model [7].

Now that the filter has been determined all that is left is to estimate the nonlinearity F. In theory we can choose a parametric form of F and fit the parameters using the given data; however, in practice an easier solution is sought. The literature commonly uses what is known as the "histogram method", which essentially creates a discretized version of F[3]. The input space of $\bar{k} \cdot \bar{s}(t)$ is divided into bins and the average spike count of each bin is calculated using $\bar{k}$, $\bar{s}(t)$ and the known spike times. In this way we recover a function that has a single value for each bin, and new data can be tested on the model by filtering the new stimulus with $\bar{k}$ and using the resulting bin value to estimate the firing rate.

2.2 The LNP using Maximum Likelihood Estimates

The drawback to the STA technique is that it requires the stimulus data to be spherically symmetric. In order to make the LNP model more powerful we need to develop a new way to estimate the parameters of the model without this restriction. The Generalized Linear Model (GLM) is one such way to accomplish this task by using Maximum Likelihood Estimates to estimate the parameters of equation (1).

The LNP model of neural response produces a rate parameter $r(t)$ to an inhomogeneous Poisson process. If y is the number of observed spikes in a short time Δ, the conditional probability of this event is given by

$$P(y|r(t)) = \frac{(\Delta r(t))^y}{y!} e^{-\Delta r(t)}. \tag{3}$$

If we are now interested in observing an entire spike train Y, which is a vector of spike counts y_i binned at a time resolution of Δ, the conditional probability of this event is given by

$$P(Y|r(t)) = \prod_{i=1}^{N} \frac{(\Delta r(t))^{y_i}}{y_i!} e^{-\Delta r(t)} \tag{4}$$

where the product runs over each element of the spike count vector Y. Normally we would view this equation as the probability of the event Y given the rate parameter $r(t)$. Instead we want to look at it from a different perspective: what is the likelihood the the rate parameter is $r(t)$, given that the outcome was Y (and that we are using a Poisson distribution specifically)? Viewed in this way equation (4) becomes a function of the rate parameter $r(t)$, and is known as the *likelihood* function [5]:

$$P(r(t)|Y) = \prod_{i=1}^{N} \frac{(\Delta r(t))^{y_i}}{y_i!} e^{-\Delta r(t)}. \tag{5}$$

The maximum value of this function, known as the maximum likelihood, will be the parameters of equation (1) that are most likely to produce the spike train Y, and these are the parameters that we wish to find.

In practice it is easier to work with the log-likelihood function, since it transforms the product into a sum. The parameters that maximize the log-likelihood function will be the same parameters that maximize the likelihood function due to the monotonicity of the log function. The log-likelihood is often denoted using $\mathcal{L}$ so that, after taking the log of the likelihood function and ignoring constant terms, equation (5) becomes

$$\mathcal{L}(r(t)|Y) = \sum_{i=1}^{N} y_i log(r(t)) - \Delta \sum_{i=1}^{N} r(t). \tag{6}$$

At this point we have an optimization problem to solve involving the linear filter $\bar{k}$ and the nonlinear function F. Fortunately, it has been shown by Paninski in [6] that with two reasonable restrictions on the nonlinear function F the log-likelihood function is guaranteed to have no non-local maxima, which avoids computational issues associated with numerical ascent techniques. The restrictions are 1) $F(u)$ is convex in its scalar argument u and 2) $log(F(u))$ is concave in u.

In the literature ([6],[9],[11]) it is common to choose a parametric form of F that follows these restrictions, like $F(u) = e^u$ or $F(u) = log(1 + e^u)$, and then optimize the function over the filter $\bar{k}$ and the parameters of the function F. The use of maximum likelihood estimates for the parameters of the model is a powerful technique that extends the nonlinear models considered later in the paper. I will also be interested in employing some form of regularization in my implementation of the GLM (and nonlinear models), which reduces overfitting of the model to noisy data.

2.3 The Spike-Triggered Covariance

The Spike-Triggered Covariance (STC), much like the STA, uses the idea of projection onto linear subspaces of the stimulus to reduce the dimensionality of the input to the model while still allowing the reduced input to maintain the salient features of the original.

The STA can be interpreted as the difference between the means of the raw stimulus data and the spike-triggering stimulus data. The STC builds on this idea and is defined as the difference between the variances of the raw stimulus data and the spike-triggering stimulus data:

$$STC = \frac{1}{N-1} \sum_{n=1}^{N} \left[\bar{s}(t_n) - STA\right] \left[\bar{s}(t_n) - STA\right]^T \tag{7}$$

Once we have constructed the STC from the data, we want to perform what is essentially a principal component analysis on the STC to ascertain which directions in stimulus space have the smallest and largest variances. For the purpose of this project I will only be interested in the direction with the smallest variance, though the technique is not limited to this. The direction of smallest variance is the eigenvector associated with the smallest eigenvalue. Any stimulus vector with a significant component along the direction of this eigenvector has a much lower chance of inducing a spike response, hence this direction is associated with an *inhibitory* neural response.

With this information in hand we can now use the STA, associated with an *excitatory* neural response, and this new vector recovered from the STC analysis, to construct a model that uses both of these subspace features to filter the data. The new model becomes

$$r(t) = F(\bar{k}_e \cdot \bar{s}(t), \bar{k}_i \cdot \bar{s}(t)) \tag{8}$$

where $\bar{k}_e$ and $\bar{k}_i$ denote the excitatory and inhibitory filters, respectively. Now all that remains is to fit the nonlinear function F. Again we could fit a parametric form to the function and estimate its parameters, but like the STA technique we will use the histogram method, binning the possible values of $(\bar{k}_e \cdot \bar{s}(t), \bar{k}_i \cdot \bar{s}(t))$ and computing the average spike count for each bin. Notice that this method will work with at most two filters (visually, at least); with more than two filters parameter estimation would be a better choice.

3 Nonlinear Models

What makes the linear models attractive are their ability to fit the data well, the tractability in estimating their parameters, and the fact that some of these parameters can be interpreted biologically. However, this method of linear stimulus processing fails to capture some of the more subtle features of a neuron's response; this is where the nonlinear models come in.

The nonlinear models are a natural extension of the linear model, and in their general form are given by the equation

$$r(t) = F\big(f_1(\bar{k_1} \cdot \bar{s}(t)), f_2(\bar{k_2} \cdot \bar{s}(t)), \ldots, f_n(\bar{k_n} \cdot \bar{s}(t))\big) \tag{9}$$

where the f_i's can be combined in any manner. Increasing evidence in neuroscience literature suggests that neural processing is performed by summing over excitatory and inhibitory inputs [11]; this fact, combined with increased ease of parameter estimation, leads us to make the assumption that the inputs of the nonlinear models will be combined as a weighted sum, in which case the nonlinear models are given by the equation

$$r(t) = F\Bigg(\sum_i f_i(\bar{k_i} \cdot \bar{s}(t))\Bigg). \tag{10}$$

The next two sections will examine particular instances of equation (10) and how the parameters are fitted.

3.1 The Generalized Quadratic Model

The Generalized Quadratic Model (GQM) is an intuitive first step into the realm of nonlinear models. It simply adds a quadratic term and a constant term to the LNP model considered in equation (1), given by

$$r(t) = F\big(\frac{1}{2}\bar{s}(t)C\bar{s}(t) + \bar{b}^T\bar{s}(t) + a\big), \tag{11}$$

where C is a symmetric matrix, $\bar{b}$ is a vector, and a is a scalar [9]. Similar to the implementation of the GLM, we want to choose a parametric form for the nonlinearity F and maximize the resulting log-likelihood function to estimate the parameter values of C, $\bar{b}$ and a.

3.2 The Nonlinear Input Model

The implementation of the Nonlinear Input Model (NIM) is the overarching goal of this project. The NIM considers the Poisson rate parameter to be a function of a sum of nonlinear inputs that are weighted by ± 1, corresponding to excitatory or inhibitory inputs [11]. The equation, similar to equation (10), is given by

$$r(t) = F\Bigg(\sum_i w_i f_i(\bar{k_i} \cdot \bar{s}(t))\Bigg), \tag{12}$$

where the values of w_i are restricted to ± 1. This model can also be thought of as a two-layer LNP model, or an LNLN model: The stimulus $\bar{s}(t)$ is projected onto various subspaces by the filters k_i; the functions f_i transform these projections nonlinearly, and the results are summed and used as an input to the larger nonlinear function F, which in turn gives a rate for the inhomogeneous Poisson process.

For the purposes of this project I will assume parametric forms of the f_i and F to make parameter fitting easier, though in practice the NIM can also fit these functions without an assumed parametric form using a set of basis functions. The f_i's will be rectified linear functions, where $f_i(u) = max(0, u)$; F will be of the form $F = log(1 + e^u)$, which guarantees no non-global maxima in the case of linear f_i's and will in practice be well-behaved for the rectified linear functions [11]. With these assumptions made, the gradient ascent routine will only need to optimize the filters k_i.

4 Databases & Implementation

The initial dataset that I will use to develop the above models is from a Lateral Geniculate Nucleus neuron's response to a single pixel of temporally modulated white noise stimuli, found at http://www.clfs.umd.edu/biology-/ntlab/NIM/. The data consists of two parts, one for fitting and one for testing. The first is a stimulus vector that contains the pixel value, changing every 0.0083 seconds, for 120 seconds which gives a total of $14,391$ values. Along with this is a vector that holds the time in seconds at which a spike was recorded. The second part of the data consists of a 10 second stimulus, again changing every 0.0083 seconds, and 64 separate trials during which spike times were recorded.

Additional datasets for testing and validation will be simulated data from other single neuron models. In general the models that I will be developing will require fitting one or two filters, each of which can contain up to 100 parameters that need to be fitted.

All software development will be performed on my personal computer, a Dell Inspiron 1525 with an Intel Core 2 Duo processor and 3GB of RAM. The programming language will be MATLAB.

5 Validation

Implementation of all the above models will be performed in a modular fashion, allowing the individual validation of each module.

For the STA model, it will be necessary to know if the averaging technique is implemented properly. To validate this part of the code, I will artificially create a stimulus pattern and the resulting spike times. Each spike will occur after a particular spatiotemporal stimulus, which will be the same each time. The rest of the stimulus will be comprised of Gaussian noise. The implementation of the STA will be validated if it is able to recover the artificial STA.

The STC model can be validated in a similar manner. I will arbitrarily choose two filters and combine them in such a way that one is excitatory (increases the firing rate) and one is inhibitory (decreases the firing rate). I will again artificially create a stimulus pattern and use the Nonlinear Inputs Model to simulate spike generation. From this stimulus pattern and the resulting spike times I can use the STC analysis to produce two linear filters as described above, which should match the artificially created filters to within some tolerance.

Validating the implementation of the maximum likelihood estimates will be of great importance, the main component of which is the gradient ascent algorithm. I will implement my own version of a gradient ascent algorithm using the Newton-Raphson method or another comparable method. I can then validate my implementation of this algorithm by using test functions for which I know the correct answer and also by comparing my code's results with several of MATLAB's built-in functions.

Furthermore, once the GLM and the GQM have been validated they can be used to validate the NIM by choosing either linear or quadratic forms for the upstream nonlinear functions, respectively, and comparing the results the these earlier models.

6 Testing

Testing of the various models will be performed using two metrics: k-fold cross validation and fraction of variance explained.

k-fold cross-validation will be used on the log-likelihood of the model LL_x, minus the log-likelihood of the "null" model LL_0. The null model predicts a constant firing rate independent of the presented stimulus. This testing will be performed on all models at the end of the project.

The fraction of variance explained is a scalar metric that compares the mean square error of the model R to the variance of the output Y:

$$FVE = 1 - \frac{MSE(R)}{Var[Y]} \tag{13}$$

The simplest model of R is the constant function equal to the mean of Y. In this case the mean square error of R is equal to the variance of Y, and the fraction of variance explained by our naive model is zero. However, if our model perfectly predicts the values of Y, the mean square error of R is equal to zero and the fraction of variance explained is equal to one.

7 Project Schedule and Milestones

The project schedule will be broken into two phases, roughly corresponding to the fall and spring semesters.

PHASE I - October through December

- Implement and validate the LNP model using the STA (October)

- Develop code for gradient ascent method and validate (October)
- Implement and validate the GLM with regularization (November-December)
- Complete mid-year progress report and presentation (December)

PHASE II - January through May

- Implement and validate the LNP model using the STC (January-February)
- Implement and validate the GQM (January-February)
- Implement and validate the NIM with linear rectified upstream functions (March)
- Develop software to test all models (April)
- Complete final report and presentation

8 Deliverables

At the end of the year I will be able to present the following deliverables:

- Implemented MATLAB code for all of the models - LNP-STA, LNP-GLM, LNP-STC, GQM, NIM
- Implemented MATLAB code for the validation and testing of all the models
- Documentation of all code
- Results of validation and testing for all the models
- Mid-year presentation and report
- Final presentation and report

References

[1] Abbott, L.F. (1999). Lapicque's introduction fo the integrate-and-fire model neuron (1907). *Brain Research Bulletin, Vol. 50*, (5), 303-304.

[2] Hodgkin, A.L., Huxley, A.F. (1952). A quantitative description of membrane current and its application to conduction and excitation in the nerve. *The Journal of physiology, 117*, (4), 500-544.

[3] Chichilnisky, E.J. (2001). A simple white noise analysis of neuronal light responses. *Network: Comput. Neural Syst.*, 12, 199-213.

[4] Paninski, L., Pillow, J., and Lewi, J. (2006). Statistical models for neural encoding, decoding, and optimal stimulus design.

[5] Shlens, J. (2008). Notes on Generalized Linear Models of Neurons.

[6] Paninski, L. (2004). Maximum Likelihood estimation of cascade point-process neural encoding models. *Network: Comput. Neural Syst.*, 15, 243-262.

[7] Schwartz, O. et al. (2006). Spike-triggered neural characterization. *Journal of Vision*, 6, 484-507.

[8] Schwartz, O., Chichilnisky, E. J., & Simoncelli, E. P. (2002). Characterizing neural gain control using spike-triggered covariance. *Advances in neural information processing systems*, 1, 269-276.

[9] Park, I., and Pillow, J. (2011). Bayesian Spike-Triggered Covariance Analysis. *Adv. Neural Information Processing Systems*, 24, 1692-1700.

[10] Butts, D. A., Weng, C., Jin, J., Alonso, J. M., & Paninski, L. (2011). Temporal precision in the visual pathway through the interplay of excitation and stimulus-driven suppression. *The Journal of Neuroscience, 31* (31), 11313-11327.

[11] McFarland, J.M., Cui, Y., & Butts, D.A. (2013). Inferring nonlinear neuronal computation based on physiologically plausible inputs. *PLoS Computational Biology.*