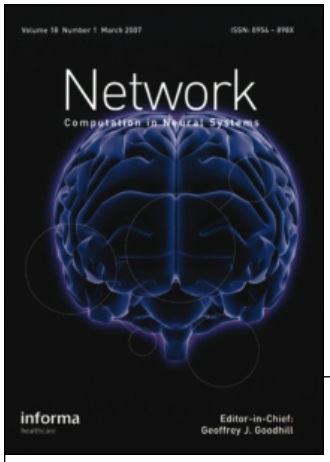
This article was downloaded by:[Columbia University]
On: 19 June 2008
Access Details: [subscription number 788828738]
Publisher: Informa Healthcare
Informa Ltd Registered in England and Wales Registered Number: 1072954
Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



Network: Computation in Neural Systems
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713663148

Inferring input nonlinearities in neural encoding models

Misha B. Ahrens [a]; Liam Paninski [b]; Maneesh Sahani [a]
[a] Gatsby Computational Neuroscience Unit, University College London, London, UK
[b] Department of Statistics and Center for Theoretical Neuroscience, Columbia University, New York, USA

Online Publication Date: 01 January 2008

To cite this Article: Ahrens, Misha B., Paninski, Liam and Sahani, Maneesh (2008) 'Inferring input nonlinearities in neural encoding models', Network: Computation in Neural Systems, 19:1, 35 — 67

To link to this article: DOI: 10.1080/09548980701813936
URL: http://dx.doi.org/10.1080/09548980701813936

PLEASE SCROLL DOWN FOR ARTICLE

Full terms and conditions of use: http://www.informaworld.com/terms-and-conditions-of-access.pdf

This article maybe used for research, teaching and private study purposes. Any substantial or systematic reproduction, re-distribution, re-selling, loan or sub-licensing, systematic supply or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The accuracy of any instructions, formulae and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.

Network: Computation in Neural Systems
March 2008; 19(1): 35–67

informa
healthcare

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Inferring input nonlinearities in neural encoding models

MISHA B. AHRENS[1], LIAM PANINSKI[2], & MANEESH SAHANI[1]

[1]*Gatsby Computational Neuroscience Unit, University College London, London, UK and* [2]*Department of Statistics and Center for Theoretical Neuroscience, Columbia University, New York, USA*

(Received 21 December 2006; accepted 19 November 2007)

Abstract
We describe a class of models that predict how the instantaneous firing rate of a neuron depends on a dynamic stimulus. The models utilize a learnt pointwise nonlinear transform of the stimulus, followed by a linear filter that acts on the sequence of transformed inputs. In one case, the nonlinear transform is the same at all filter lag-times. Thus, this "input nonlinearity" converts the initial numerical representation of stimulus value to a new representation that provides optimal input to the subsequent linear model. We describe algorithms that estimate both the input nonlinearity and the linear weights simultaneously; and present techniques to regularise and quantify uncertainty in the estimates. In a second approach, the model is generalized to allow a different nonlinear transform of the stimulus value at each lag-time. Although more general, this model is algorithmically more straightforward to fit. However, it has many more degrees of freedom than the first approach, thus requiring more data for accurate estimation. We test the feasibility of these methods on synthetic data, and on responses from a neuron in rodent barrel cortex. The models are shown to predict responses to novel data accurately, and to recover several important neuronal response properties.

Keywords: *Auditory system, somatosensory processing, visual system*

Correspondence: M. Sahani, Gatsby Computational Neuroscience Unit, University College London, London, UK.
E-mail: maneesh@gatsby.ucl.ac.uk

ISSN 0954-898X print/ISSN 1361-6536 online/08/010035–67 © 2008 Informa UK Ltd.
DOI: 10.1080/09548980701813936

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Introduction

Neural encoding models predict how the instantaneous firing rate of a neuron varies in response to a dynamic input, such as a jittering bar in the visual field, a time-varying sound, the activity of a group of upstream neurons, or a combination of such an input and the spike history of the neuron itself. One major reason for interest in such models is that, once fitted to neural data, their parameters can be used to investigate the encoding properties of the modeled neuron; this may, in turn, shed light on the function of the corresponding brain area. In one of the simplest and most widely-used models, the predicted firing rate is a weighted linear combination of preceding stimulus values. In many cases, such linear models do not predict the firing rate of a neuron well (Sahani and Linden 2003b; Machens et al. 2004). Thus, while their parameters may sometimes be broadly indicative of the encoding properties of the neuron, the picture they yield is at best incomplete, and may occasionally be radically inaccurate (e.g., Christianson et al. 2007). This suggests that nonlinear encoding models may be needed to provide an accurate description of the neuron's functional response (e.g., Marmarelis and Naka 1973; de Ruyter van Steveninck and Bialek 1988; Schwartz et al. 2002).

Considerable effort has recently been directed towards linear-nonlinear-Poisson (LNP) models, where a linear temporal filter acting on a time-varying stimulus signal is followed by a static nonlinearity (Brenner et al. 2000; Schwartz et al. 2002; Paninski, 2003, 2004; Sharpee et al. 2004; Simoncelli et al. 2004; Pillow and Simoncelli 2006). One motivation for such models is to capture the particular nonlinearity inherent in neuronal spike generation, although some other nonlinearities may also be described this way. By contrast, here we focus on nonlinear transforms that *precede* a temporal linear filtering stage. Such transforms may model nonlinear synaptic or dendritic responses in the neuron being described, but may also capture nonlinearities at earlier stages of processing or in receptor transduction (where, for example, stimulus strength may be encoded logarithmically, or with power-law scaling). Input nonlinearities such as these can, in principle, lead to significant failures of the linear model. To take an elementary example, suppose that a neuron combined filtered inputs from two populations of half-wave rectifying sensors, the populations being sensitive to stimulus deflections in opposite directions, as in Figure 1. If the influence of both populations

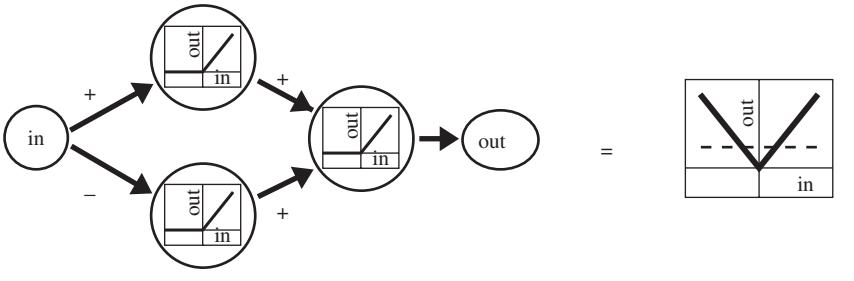


Figure 1. Schematic network with a symmetric input-output relation. −: inhibitory connection. +: excitatory connection. All ''neurons'' are half-wave rectifiers. The output will be insensitive to the sign of the input; hence a linear fit to the I–O function (dashed line) is constant.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

were roughly equal, the neuron would effectively respond to the absolute value of the sensory inputs. In this case, a linear model fitted to a stimulus that contained equal deflections in both directions, could do no better than predict a constant firing rate.

We describe two models designed to capture such input nonlinearities, inspired by techniques that generalize linear regression to the nonlinear setting (e.g., Suits et al. 1978). The first is a bilinear model, in which, prior to a linear combination, a single estimated nonlinear transform is applied to all the stimulus values. In the second model this constraint is relaxed, and a separate nonlinearity is estimated for each input to the linear combination stage. For reasons that will become apparent, we will refer to this as the ''full-rank'' model. It is related to the generalized additive model (Breiman and Friedman 1985; Hastie and Tibshirani 1999). Despite the larger number of parameters involved, the full-rank model is algorithmically more straightforward to fit than the bilinear one. However, the many additional degrees of freedom mean that, in comparison to the bilinear model, many more data are needed to achieve a given level of reliability in the estimated parameters. Furthermore, the resulting description is considerably less compact than the bilinear model, potentially leading to difficulties in interpretation.

Algorithms to estimate the parameters of both models are described at the beginning of the following section. The bilinear model, and implicitly the full-rank model, have appeared before in the context of Hammerstein cascades or NL cascade models (e.g., Narendra and Gallman 1966; Hunter and Korenberg 1986; Juusola et al. 1995; Bai 1998; Westwick and Kearney 2001). Here, we give these models a probabilistic basis, which allows us to develop principled techniques for regularisation (see ''Regularisation'') and estimation of error bars (see ''Error bars through Gibbs sampling''); we draw connections between the full-rank model and the bilinear model (see ''Rank-k models''); and we extend the formulation of the bilinear model to the framework of predicting point-process spike trains, leading to the generalized bilinear model or the ''NLNP model''. (See ''Fitting spike trains: Generalized bilinear or NLNP models''). We then evaluate the models on simulated and real neural data.

The models

The bilinear model

Model Specification. Consider a one-dimensional time-varying stimulus $s(t)$, which evokes a neuronal response $r(t)$. The linear predictive model is given by $\hat{r}(t) = c + \sum_{\tau=0}^{\tau_{\max}} w_\tau s(t-\tau)$, where c is a background firing rate, $\mathbf{w}$ is a $(\tau_{\max}+1)$–dimensional vector of weights – sometimes called the *linear receptive field* of the neuron – and $\hat{r}(t)$ is the predicted firing rate of the cell at time t. The model parameters are set so that $\hat{r}(t)$ matches the real firing rate $r(t)$ as closely as possible. In the following, $r(t)$ will usually represent a peri-stimulus-time histogram (PSTH), i.e., the number of spikes in a time bin around t, averaged over multiple trials in which the same stimulus is presented. In this case, a natural measure of closeness is the average squared difference between $\hat{r}(t)$ and $r(t)$, corresponding to the Gaussian likelihood. In a later section we will consider the case in which $r(t)$ is a

Downloaded By: [Columbia University] At: 22:13 19 June 2008

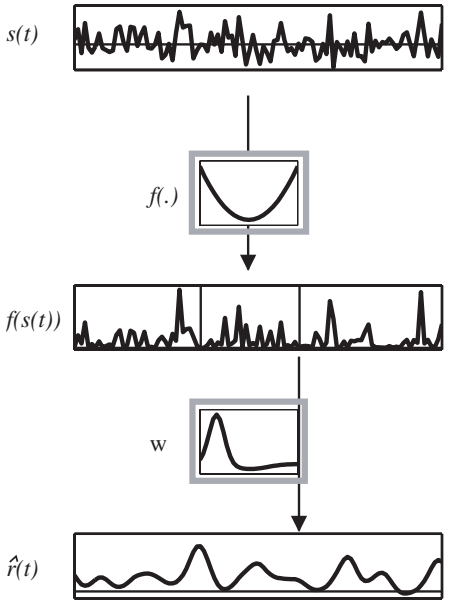


Figure 2. Schematic view of the bilinear model. It consists of two stages, or neural processing operations. First, the stimulus values are transformed by an input nonlinearity $f(\cdot)$, and second, a temporal filter **w** acts on the transformed stimulus values to form a predicted spike rate. **w** and $f(\cdot)$ are both unknown and to be learnt from the data. An output nonlinearity is optional and not shown in the figure. If the input nonlinearity stage is removed, or, equivalently, $f(\cdot)$ set to the identity, then the model reduces to the linear model. The free parameters of the model are surrounded by grey boxes.

single-trial spike train with 1 (spike) or 0 (no spike) in each time bin. Here, the negative log-likelihood (or deviance) of a point-process model will be a more suitable distance metric. Generalization to higher dimensional stimuli is straightforward: in this case, the index τ would range over time and space, for example, instead of just time.

In this section, we introduce an unknown pointwise transformation $f(\cdot)$ of $s(t)$ into the model, and derive an algorithm to estimate the corresponding parameters. This transformation will be referred to as the ''input nonlinearity''. The model has the form

$$\hat{r}(t) = c + \sum_{\tau=0}^{\tau_{\max}} w_\tau f(s(t-\tau)). \tag{1}$$

In many cases, the appropriate f is not known and must be estimated from the data, at the same time as the linear parameters. Once f has been estimated, the resulting model is conceptually almost as simple as the linear model, but can be considerably more powerful. We call this the ''bilinear'' model for reasons that will become clear below. Figure 2 shows the decomposition of the transformation between the stimulus and the predicted neural response. The constant c (not shown in the figure) is important because it allows the other terms of the model to describe fluctuations around the baseline firing rate, rather than the firing rate itself, thus reducing the chances of predicting negative firing rates.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Estimation procedure. For fixed f, it is straightforward to estimate optimal weights $\mathbf{w}$ by linear regression from the transformed stimulus. Thus, one procedure to find a suitable input nonlinearity might be to try a variety of plausible functions, re-estimating $\mathbf{w}$ for each one, and then select the function and weight vector that provide the best overall prediction. However, this approach rapidly becomes impractical as the space of functions to be explored expands, particularly given the possibility that each neuron in a population might be best fit with a different f. An alternative is to *parametrize* f, as a linear combination, with weights b_i, of a fixed set of basis functions $\{f_i\}$:

$$f(\cdot) = \sum_i b_i f_i(\cdot). \tag{2}$$

A similar parametrization is often used in the context of standard regression, and a general discussion of basis set selection may be found in the relevant literature (e.g., Hastie et al. 2001). Basis functions should to be chosen to span as well as possible the space of anticipated input nonlinearities; a suitable choice might be based on background knowledge of the physiology and responses of the brain area, on the particular stimulus used, on computational resources (more basis functions require more memory and time), and possibly some initial exploration. In this article we use 16 piecewise linear basis functions; see Appendix A for the definition. Inserting Equation 2 into the model (Equation 1) gives: $\hat{r}(t) = c + \sum_{\tau i} w_\tau b_i f_i(s(t-\tau))$. Making the abbreviation $M_{t\tau i} = f_i(s(t-\tau))$, the model can be re-expressed in a compact way: $\hat{r}(t) = c + \sum_{\tau i} w_\tau b_i M_{t\tau i}$. This expression is further simplified by redefining M. Let us rewrite the three-dimensional object M as a family of matrices, one for each time point t, so that for all (t, τ, i), the $(\tau, i)^{\text{th}}$ element of the matrix $\mathsf{M}(t)$ is $M_{t\tau i}$– that is, $[\mathsf{M}(t)]_{\tau i} = M_{t\tau i}$. We then augment each $\mathsf{M}(t)$ by one row and one column; in block notation,

$$\mathsf{M}(t) \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & \mathsf{M}(t) \end{pmatrix}, \tag{3}$$

and then reform these augmented matrices into a (now augmented) data array M. Similarly, we augment $\mathbf{w} \leftarrow [w_c \ \mathbf{w}]$ and $\mathbf{b} \leftarrow [b_c \ \mathbf{b}]$. If we now write, using the augmented objects,

$$\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i}.$$

then we see that the constant term c has been replaced by the product $w_c b_c$. In this form, the model has two parameter vectors $\mathbf{w}$ and $\mathbf{b}$, with $\mathbf{w}$ describing the response to time, and $\mathbf{b}$ describing the input nonlinearity. M is the augmented data array and is fixed. Note that it is convenient here to use the same symbols for both the original and augmented objects. In the following, the exact meanings of $\mathbf{w}$, $\mathbf{b}$ and M will vary – how they are defined will be clear from context. Using the same symbols leaves the structure of the model, and the associated algorithms, invariant.

The parameter vectors $\mathbf{w}$ and $\mathbf{b}$ are estimated by minimizing the squared distance between the observed and the predicted spike rates, $\mathcal{E} = \sum_t (r(t) - \hat{r}(t))^2 = \|\mathbf{r} - \hat{\mathbf{r}}\|^2$. (An alternative is to maximise a point-process likelihood; this will be

Downloaded By: [Columbia University] At: 22:13 19 June 2008

described below.) One way to carry out this minimisation is through a sequence of alternating updates. We first group terms in the model to obtain:

$$\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i} = \sum_\tau w_\tau \left(\sum_i b_i M_{t\tau i} \right) = \sum_\tau w_\tau B_{t\tau}$$

or $\hat{\mathbf{r}} = \mathsf{B}\mathbf{w}$, where the matrix B is defined by $\mathsf{B}_{t\tau} = \sum_i b_i M_{t\tau i}$. In other words, if $\mathbf{b}$ is held fixed it can be combined with the other fixed components of the model, which are collected in the data array M, to produce a pseudo-data matrix B. As the resulting expression is linear in $\mathbf{w}$, it can easily be inverted to obtain the estimate $\mathbf{w} = (\mathsf{B}^\top \mathsf{B})^{-1} \mathsf{B}^\top \mathbf{r}$. This is the unique best estimate of $\mathbf{w}$ under the squared-error objective function $\mathcal{E}$ given fixed $\mathbf{b}$, provided $\mathsf{B}^\top \mathsf{B}$ is of full rank (if $\mathsf{B}^\top \mathsf{B}$ is of reduced rank, the inverse in the definition of $\mathbf{w}$ is interpreted as a pseudoinverse, uniquely selecting one of the many optimal estimates).

Alternatively, if $\mathbf{w}$ is held fixed, the model can be written

$$\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i} = \sum_i b_i \left(\sum_\tau w_\tau M_{t\tau i} \right) = \sum_i b_i W_{ti},$$

with $W_{ti} = \sum_\tau w_\tau M_{t\tau i}$. Now the conditional estimate for $\mathbf{b}$ is $(\mathsf{W}^\top \mathsf{W})^{-1} \mathsf{W}^\top \mathbf{r}$, which is again optimal in the sense described above.

Thus, each individual update for $\mathbf{w}$ or $\mathbf{b}$, reduces the squared error in the prediction. By repeating the updates in alternation, starting from an arbitrary initial value for one of the parameter vectors, we obtain an algorithm that is guaranteed to converge to a (local) minimum in the objective function. The values of $\mathbf{w}$ and $\mathbf{b}$ at convergence then yield both the baseline firing rate in their first elements $c = w_c \cdot b_c$, and an optimal temporal filter and input nonlinearity in their remaining elements. The procedure described here has the structure of alternating least squares (ALS) (e.g., Young et al. 1976). Although the matrices B and W were introduced above for clarity, in practice, if the length of the stimulus is large compared to the size of $\mathbf{w}$ and $\mathbf{b}$, there is a faster and more economical implementation of the algorithm. This is described in Appendix B. Figure 3 shows an example of a model fitted to artificial data themselves generated by a fixed bilinear model with a threshold-linear input nonlinearity. The underlying firing rate was determined according to $r(t) = \sum_\tau w_\tau f(s(t-\tau))$, with $s(t)$ a Gaussian distributed stimulus, $\mathbf{w}$ the temporal filter shown and $f(x) = x$ if $x > 0$ and $f(x) = 0$ if $x \leq 0$. The observed firing rate was taken to be the average of five spike trains drawn from this underlying firing rate, rectified so that negative values were set to zero (i.e. $[r(t)]^+$ is the probability of having a spike in the bin around t).

Note that, once the input nonlinearity has been determined for a particular cell, it may be fixed and used to derive extended versions of the linear model such as the generalized linear model (which may include an output nonlinearity) or models with spike history terms (Pillow et al. 2005; Truccolo et al. 2005). If the input nonlinearity is consistent across a subset of a population of cells, it may also be fixed (or approximated by a simpler function) so that estimation in the remainder of the population can be carried out more directly. It is also possible to estimate spike history terms simultaneously with the input nonlinearity and the temporal filter (see "Fitting spike trains: Generalized bilinear or NLNP models").

Downloaded By: [Columbia University] At: 22:13 19 June 2008

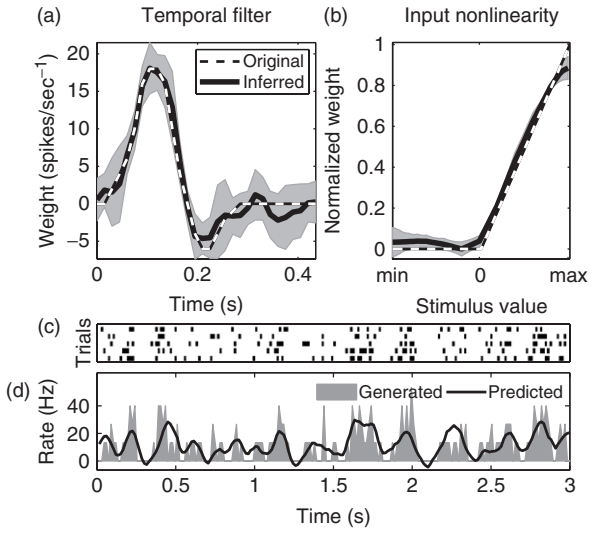


Figure 3. Example of a bilinear model fit. A known bilinear model was used to generate an underlying firing rate, from which five spike trains were drawn according to P(spike at t) $= [r(t)]^+$. The resulting PSTH was then used to infer new bilinear parameters by minimising the squared error. (a): Temporal filter and (b): Input nonlinearity of the original (dashed) and inferred (black) bilinear models. Error bars are shown in grey. The inferred value of the spontaneous rate c, set to zero in the original model, was indistinguishable from zero. (c): Simulated spike trains. (d): Generated PSTH (grey) and the firing rate predicted by the inferred model (black). With the bin size defined to be 15 ms, we used 15 s of the observed firing rate per trial (i.e. 75 s of data in all) to infer the bilinear model. With this bin size, the average firing rate is 16 Hz.

As discussed above, the squared-error objective function for a linear model is guaranteed to have a unique optimum. Unfortunately there is no similar guarantee of uniqueness for the bilinear model. The alternating least squares algorithm is guaranteed to converge, and, at each step there is a unique optimum for each of the parameter vectors **w** and **b** *conditioned* on the other; however, as the other vector also changes during the optimisation, the point of convergence may lie at only a local minimum of $\mathcal{E}$. This issue will be discussed further in the next section, where it will be seen to arise from a non-convexity of the parameter space. However, although we were able to find a few such local minima in numerical simulations using random data arrays M and **r**; in practice, these have not caused difficulty when working with real neural data. For example, at most two different local minima were ever observed in the neuronal data discussed in this article (see ''Demonstration on real data''), and they were almost identical in shape. However, it is possible that in larger models careful initialisation of the parameters might be important (e.g., see ''Rank-k models''). Another important property of the model is the fact that different parameter values may imply identical models (e.g., if **w** is scaled up, but **b** scaled down by the same factor, the model does not change); this will be discussed further below (see ''Degeneracies''). Because of this invariance, we scale the input nonlinearity to have a maximum value of 1 in all the given examples, so that the scale of the full model is carried by the temporal filter.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

The full-rank model

The bilinear model is directly linked to the concept of a ''separable'' receptive field (DeAngelis et al. 1995; Depireux et al. 2001; Linden et al. 2003). Consider, for example, a visual spatiotemporal receptive field. Taking only one spatial dimension for simplicity, this can be described by a matrix W, with elements $W_{\tau x}$, such that the corresponding linear spike rate prediction model would be $\hat{r}(t) = \sum_{\tau=0}^{\tau_{\max}} \sum_x W_{\tau x} S(t-\tau, x)$, where $S(t,x)$ is the (one dimensional) time-varying movie that is played on the retina. Such a receptive field would be called separable if the matrix W was of rank 1; that is, it could be written as the outer product of two separate vectors, **u** in time and **v** in space: $W_{\tau x} = u_\tau v_x$. Even if W were not strictly of rank 1, provided its singular value spectrum were dominated by just one value, it might still be useful to approximate the receptive field in a separable form. A separable model is described by only $\dim(\mathbf{u}) + \dim(\mathbf{v})$ parameters, instead of the, typically much larger, $\dim(\mathbf{u}) \times \dim(\mathbf{v})$ for the full-rank matrix W. Thus, fewer data are needed to estimate the separable model well. On the other hand, the class of separable receptive fields is strictly less flexible than the general class of full-rank receptive fields; for instance, a separable receptive field cannot model direction selectivity in a visual cell.

The parameters of the bilinear model appear as the products $w_\tau b_i$. Thus, this model may be thought of as embodying a separable receptive field in time (τ) and in stimulus value (i), with the data tensor $M_{t\tau i}$, a two-dimensional dynamic stimulus that varies in the τ and i dimensions. This view then suggests a generalization of the bilinear model, in which the rank-1 matrix of elements $w_\tau b_i$, is replaced by a general matrix $C_{\tau i}$ to form the *full-rank model*. As a generalization of the bilinear model, the full-rank model has the potential to capture more intricate structure in the stimulus response function; but as it has more parameters ($\dim(\mathbf{w}) \times \dim(\mathbf{b})$ instead of $\dim(\mathbf{w}) + \dim(\mathbf{b})$), the data requirements and risks of overfitting will be higher.

The full-rank model uses the same data array M as the bilinear model, but is linear in the parameters $C_{\tau i}$ and c,

$$\hat{r}(t) = c + \sum_{\tau i} C_{\tau i} M_{t\tau i} = c + \sum_{\tau i} C_{\tau i} f_i(s(t-\tau)). \tag{4}$$

Thus, the minimum-squared-error parameters can be found in a single step, using standard linear regression methods. This implies that the full-rank model has a unique global optimum in the sense discussed above. In particular, both the squared-error objective function $\mathcal{E}$ (defined as for the bilinear model), and the parameter space (the space of all matrices $C_{\tau i}$ and constant offsets c) are convex: $\mathcal{E}$ is convex because it is quadratic, and the parameter space is convex because adding any two general matrices together produces another valid matrix. This joint convexity guarantees the uniqueness of the optimum. This view also suggests why the bilinear model may have multiple squared-error minima. If we regard the bilinear model as a restriction of the full-rank model to rank-1 parameter matrices, the objective function remains convex, but the space of parameters is no longer so. In particular, a convex combination of two rank-1 matrices is generally of rank 2. This constraint on the parameters then leads to the possibility of multiple optima.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Rank-k models

The bilinear model may be viewed as a rank-1 special case of the full-rank model. In a similar vein, one may also consider rank-k sub-models for higher values of k. Such models may be able to capture more detailed response properties of a neuron than can the rank-1 bilinear model, while needing fewer data for accurate estimation than would the full-rank model. One common method for finding low rank approximations to a matrix is to use the singular value decomposition (SVD, Strang 1988). Thus, taking the leading outer product term in the SVD of the full-rank parameter matrix would yield a rank-1 approximation with the form of the bilinear input nonlinearity model (Bai 1998). However, the parameters $(\mathbf{w}, \mathbf{b})$ found in this way minimize the Frobenius distance between C and $\mathbf{w}\mathbf{b}^\top$, rather than the squared error between the observed and the predicted firing rates. Thus, while the SVD method may provide a good initial guess for $\mathbf{w}$ and $\mathbf{b}$, the alternating least squares procedure described above is still needed to minimise the objective function $\mathcal{E}$.

If the weight matrix C is of rank k, it can be written as the sum of k outer product terms, $C_{\tau i} = w_\tau^1 b_i^1 + w_\tau^2 b_i^2 + \cdots + w_\tau^k b_i^k$. Thus, the rank-$k$ model is equivalent to a sum of k bilinear models, and the firing rate prediction is given by $\hat{r}(t) = c + \sum_k \sum_\tau w_\tau^k f^k(s(t-\tau))$. Again, the parameters $\mathbf{w}^{1\ldots k}$ and $\mathbf{b}^{1\ldots k}$ may be initialized using SVD, but using SVD alone would minimize an inappropriate objective function (the Frobenius distance to the full-rank matrix C). Thus, the alternating least squares procedure is needed to minimize the squared error. This requires that the data array M be redefined once again. For example, if $k = 3$, the matrix $\mathsf{M}(t)$ would be augmented thus

$$\mathsf{M}(t) \leftarrow \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \mathsf{M}(t) & 0 & 0 \\ 0 & 0 & \mathsf{M}(t) & 0 \\ 0 & 0 & 0 & \mathsf{M}(t) \end{pmatrix},$$

(with ''0'' indicating blocks of 0's) and these matrices then reformed into the data array as before. Similarly the vectors $\mathbf{w}$ and $\mathbf{b}$ are redefined as $\mathbf{w} \leftarrow [w_c\ \mathbf{w}^1\ \mathbf{w}^2\ \mathbf{w}^3]$ and $\mathbf{b} \leftarrow [b_c\ \mathbf{b}^1\ \mathbf{b}^2\ \mathbf{b}^3]$. With these redefinitions, the rank-3 model assumes the same form as the simple bilinear model: $\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i}$. Thus, the parameters can be found with the ALS procedure, and the three bilinear models (as well as the constant $c = w_c b_c$) recovered.

Models with multiple stimulus features

The framework discussed above allows for arbitrary transformations of the instantaneous stimulus value, once the stimulus has been expressed as a time-varying quantity $s(t)$. But there may be multiple ways to translate a physical stimulus into $s(t)$ – for instance, in the context of analysis of whisker barrel data, $s(t)$ might correspond to the feature ''position'', or to the feature ''velocity'', of whisker motion. These features lead to different bilinear models with potentially different performances[1] (Pinto et al. 2000). This is not true in a linear model if the stimulus

Downloaded By: [Columbia University] At: 22:13 19 June 2008

features are linear transformations of one another. In the bilinear model, however, that equivalence is broken by the input nonlinearity.

One way to choose the most appropriate stimulus or stimulus feature would be to train sequentially a model for each stimulus, and then find the most predictive of the resulting models. However, it might be that the stimuli features interact or jointly influence the firing rate, or there might be two distinct inputs that might affect the neuronal firing with potentially different input nonlinearities and temporal properties, e.g., the light intensity and sound volume of an audiovisual stimulus. In this case, it would be desirable to have a model that uses all relevant stimuli to construct the predicted firing rate. It is common practice to formulate the design matrices of linear models such that they include multiple stimuli (e.g., Luczak et al. 2004); a similar formulation can be applied to the data tensor M of the bilinear and full-rank models. As an example, consider the following model that assumes an additive combined effect of three stimuli on the firing rate:

$$\hat{r}(t) = c + \sum_{\tau} \left(w^1_\tau f^1(s^1(t-\tau)) + w^2_\tau f^2(s^2(t-\tau)) + w^3_\tau f^3(s^3(t-\tau)) \right),$$

where $s^{1,2,3}(t)$ are three different stimuli or stimulus features, each with their own temporal filter (e.g., $\mathbf{w}^1$) and input nonlinearity (e.g., f^1, determined by $\mathbf{b}^1$). We define tensors $\mathsf{M}^{1,2,3}$ for each stimulus as before, e.g., $M^1_{t\tau i} = f^1_i(s^1(t-\tau))$ (the different stimuli may be assigned the same, or different, basis functions). In tensor notation, the model is

$$\hat{r}(t) = c + \sum_{\tau i} \left(w^1_\tau b^1_i M^1_{t\tau i} + w^2_\tau b^2_i M^2_{t\tau i} + w^3_\tau b^3_i M^3_{t\tau i} \right).$$

Training such a model is similar to training a rank-k model. Again, we define a new stimulus tensor by collecting each of the inputs in a diagonal block at every time:

$$\mathsf{M}(t) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \mathsf{M}^1(t) & 0 & 0 \\ 0 & 0 & \mathsf{M}^2(t) & 0 \\ 0 & 0 & 0 & \mathsf{M}^3(t) \end{pmatrix},$$

where e.g., $[\mathsf{M}^1(t)]_{\tau i} = M^1_{t\tau i}$. (Each block of $\mathsf{M}(t)$ is now associated with a different stimulus feature or stimulus, whereas in the rank-k model each block was associated with the same input.) If we also concatenate the $\mathbf{w}^{1,2,3}$ and the $\mathbf{b}^{1,2,3}$ vectors to form $\mathbf{w} = [w_c\ \mathbf{w}^1\ \mathbf{w}^2\ \mathbf{w}^3]$ and $\mathbf{b} = [b_c\ \mathbf{b}^1\ \mathbf{b}^2\ \mathbf{b}^3]$, then the model again takes on the form of the simple bilinear model: $\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i}$. Thus, the parameters $\mathbf{b}$ and $\mathbf{w}$ can be estimated as before, using the ALS procedure. Again, the optimal $\mathbf{b}$ may be uniquely defined given $\mathbf{w}$, and vice versa: all of our convexity discussion carries through unmodified to this more general setting. Once converged, the individual temporal filters and input nonlinearities, and the constant offset, may be extracted from these concatenated vectors.

The full-rank analogue of the above uses the same stimulus array. The model is

$$\hat{r}(t) = c + \sum_{\tau i} \left(C^1_{\tau i} M^1_{t\tau i} + C^2_{\tau i} M^2_{t\tau i} + C^3_{\tau i} M^3_{t\tau i} \right) = \sum_{\tau i} C_{\tau i} M_{t\tau i}.$$

Downloaded By: [Columbia University] At: 22:13 19 June 2008

This is again the standard linear form of a full-rank model. After estimating the matrix C with the usual method the individual receptive fields are obtained as follows:

$$\mathsf{C} = \begin{pmatrix} c & 0 & 0 & 0 \\ 0 & \mathsf{C}^1 & 0 & 0 \\ 0 & 0 & \mathsf{C}^2 & 0 \\ 0 & 0 & 0 & \mathsf{C}^3 \end{pmatrix}.$$

The 0's of this matrix indicate blocks of entries that do not participate in the regression due to the block-diagonal form of M, because in the model, these entries are multiplied by zeroes.

Regularisation

A concern when fitting models to limited or noisy data, is that the parameters of the model be *overfit* to the particulars of the data set, thereby increasing the error in the parameter values, and in firing rates predictions for novel stimuli. Such concerns may be partially addressed by regularisation.

In minimizing the squared error $\mathcal{E} = \|\mathbf{r} - \hat{\mathbf{r}}\|^2$, we have implicitly been looking for the maximum likelihood (ML) solution for the parameters ($\mathbf{w}_{\mathrm{ML}}$ and $\mathbf{b}_{\mathrm{ML}}$) under a noise model

$$r(t) = \hat{r}(t) + \sigma\eta(t),$$

where $\eta(t)$ is a zero mean and unit variance Gaussian random variable, and σ is the (unknown) noise scale. Gaussian noise is a reasonable assumption because the PSTH is the average of multiple spike trains, so that by the central limit theorem, the noise around the ''true'' rate is approximately Gaussian. The likelihood of the observed spike rate under this model is $(2\pi\sigma^2)^{-T/2}\exp(-\|\mathbf{r} - \hat{\mathbf{r}}\|^2/2\sigma^2)$ and therefore minimizing $\mathcal{E}$ maximizes the likelihood. This probabilistic formulation permits both principled regularisation techniques, and estimation of the uncertainty in the parameter values. Given an observation of $\mathbf{r}$, the noise term induces a probability distribution over the parameters of the model. If we specify prior probability distributions on those parameters that describe our expectations, such as a degree of smoothness in $\mathbf{w}$, we can obtain regularised estimates of the parameters of the model. A convenient (technically, conjugate) choice is a Gaussian prior; e.g., $P^{\mathrm{prior}}(\mathbf{w}) \sim \mathcal{N}(0, \mathsf{S}^w)$ with S^w the prior covariance matrix describing the expected smoothness, size, etc. of $\mathbf{w}$. If the following, the prior will often be specified by the *inverse* covariance D^w so that $\mathsf{S}^w = [\mathsf{D}^w]^{-1}$.

Bilinear model. As described above, the update for $\mathbf{w}$, when $\mathbf{b}$ is fixed, is just the solution to a linear regression problem with design matrix $B_{t\tau} = \sum_i b_i M_{t\tau i}$. Incorporating a Gaussian prior distribution into a linear regression problem with Gaussian noise is quite well understood in the neural encoding setting (Sahani and Linden 2003a; Machens et al. 2004): we simply maximize the log-posterior distribution on $\mathbf{w}$ instead of the log-likelihood. This log-posterior may be written as $\log(P(\mathbf{w}, \mathbf{b}|\mathsf{M}, \mathbf{r})) = -(1/2\sigma^2)\hat{\mathbf{r}}^{\mathsf{T}}\hat{\mathbf{r}} + (1/\sigma^2)\mathbf{r}^{\mathsf{T}}\,\hat{\mathbf{r}} - (1/2)\mathbf{w}^{\mathsf{T}}\mathsf{D}^w\mathbf{w} - (1/2)\mathbf{b}^{\mathsf{T}}\mathsf{D}^b\mathbf{b} +$ const, where the constant does not depend on $\mathbf{w}$ or $\mathbf{b}$, while $\hat{\mathbf{r}}$ is given in terms of

Downloaded By: [Columbia University] At: 22:13 19 June 2008

w and **b** by the model; this expression can be maximised analytically with respect to **w** to obtain the usual regularised least squares solution:

$$\mathbf{w} = (\mathsf{B}^{\mathsf{T}}\mathsf{B} + \hat{\sigma}^2\mathsf{D}^{w})^{-1}\mathsf{B}^{\mathsf{T}}\mathbf{r}.$$

Here, $\hat{\sigma}^2$ is an estimate of the noise scale σ^2, which is often absorbed into the definition of D^{w} or made part of an automatic regularisation method, to be discussed shortly. In a technique known as ridge regression, the matrix D^{w} is a multiple of the identity, so that the values of **w** are encouraged to be small. Another common choice for D^{w} is a matrix with 2's on the diagonal and -1's on the neighbouring positions, all scaled by a parameter λ^{w} which sets the "strength" of the regularisation (equivalently, $1/\lambda^{w}$ sets the reliability of the data). Such a D^{w} will penalise high derivatives of **w**; as can be seen from the corresponding Gaussian log-prior on **w**, which is proportional to $-\mathbf{w}^{\top}\mathsf{D}^{w}\mathbf{w} = -\sum_i[\mathbf{w}(i+1) - \mathbf{w}(i)]^2$. The regularised final estimate of **w**, after the ALS iterations, is now the maximum a posteriori (MAP) estimate, $\mathbf{w}_{\mathrm{MAP}}$.

The case of **b** is more complicated, because our prior expectations may be relevant to the input nonlinearity $f(x) = \sum_i b_i f_i(x)$ rather than to **b** itself: smoothness in **b** is not exactly the same as smoothness in f. In Appendix C we derive expressions for D^{b} that control the first and second derivatives of f. D^{b} is defined there in quadratic form, so that e.g., to penalise the first derivative of f we find D^{b} such that $\mathbf{b}^{\mathsf{T}}\mathsf{D}^{b}\mathbf{b} = \lambda^{b} \cdot \int(\partial f(x)/\partial x)^2 \mathrm{d}x$. (This is the continuous analogue of D^{w} defined above.) Again there is a multiplier λ^{b} which sets the strength of the regularisation. Once the prior has been set, the update for **b** becomes $\mathbf{b} = (\mathsf{W}^{\mathsf{T}}\mathsf{W} + \hat{\sigma}^2\mathsf{D}^{b})^{-1}\mathsf{W}^{\mathsf{T}}\mathbf{r}$, with $W_{ti} = \sum_\tau w_\tau M_{\tau t i}$. The regularised final estimate is now $\mathbf{b}_{\mathrm{MAP}}$.

The priors described above depend on simple scaling parameters λ^{w} and λ^{b}, which determine the strength of regularisation. Further parameters may be used to control other aspects of the priors, such as the extent of smoothing (e.g., Sahani and Linden 2003a). That is, the priors are formulated to depend on one or more parameters $\theta_{w,b}$; these parameters may be determined entirely by prior expectations, or may be chosen by a cross-validation procedure; good values may depend on the size, amount of noise, etc, of the dataset. A more principled way of setting the θ's is through an automatic adaptive regularisation method described in Appendix F. This method is similar to evidence optimization techniques that have been applied to linear models (Sahani and Linden 2003a). After implementation, these automatic techniques can produce good results with no time spent on manual intervention.

Full-rank model. Regularising the full-rank model is not straightforward, because in the time direction it uses a discrete basis, whereas in the stimulus value direction it uses a piecewise linear (i.e. continuous) basis. In Appendix D we adopt the somewhat unusual strategy of penalizing the first derivative in the time direction (this is the standard thing to do) but the second derivative in the stimulus value direction (this still promotes smoothness but allows functions to have steep slopes). Again, the regularisation is done through a Gaussian prior on the model parameters C, specified as a regularisation matrix D^{C} – the analogue of D^{w} or D^{b} for the bilinear model parameters. In Appendix D, we derive expressions for D^{C} so that rough receptive fields are penalized. Linear evidence optimisation techniques can be readily applied here (e.g., using an adaptive prior $\lambda\mathsf{D}^{\mathrm{C}}$ and learning an optimal value

Downloaded By: [Columbia University] At: 22:13 19 June 2008

for λ, or by using other forms of prior covariance matrices; see Sahani and Linden 2003a).

Error bars through Gibbs sampling

Error bars for the parameters of a linear model with Gaussian noise are easy to find. A linear model is defined by a design matrix X (the stimulus), a set of weights $\mathbf{v}$ (the receptive field), an observation vector $\mathbf{y}$ (the spike rate) and a noise scale σ, so that $\mathbf{y} = \mathsf{X}\mathbf{v} + \sigma\eta$, with η independent Gaussian noise with zero mean and unit variance. If D were the inverse prior covariance, then the error bars on $\mathbf{v}$ would be estimated by the square root of the diagonal elements of the posterior covariance matrix $\hat{\sigma}^2(\mathsf{X}^\mathsf{T}\mathsf{X} + \hat{\sigma}^2\mathsf{D})^{-1}$, as these diagonal elements give the marginal variances of the parameters. In the bilinear case, the error bars are determined by the spread of the posterior distribution of the parameters, $P(\mathbf{w}, \mathbf{b}|\mathbf{r}, \mathsf{M}, \mathsf{D}^w, \mathsf{D}^b, \hat{\sigma}^2)$, around the estimated parameters, $\mathbf{w}_{\mathrm{MAP}}$ and $\mathbf{b}_{\mathrm{MAP}}$. Unlike the linear case, there is no simple analytical estimate for this distribution, or for the corresponding marginal variances, due to the dependencies between $\mathbf{w}$ and $\mathbf{b}$. Fortunately, however, it is quite easy to sample from this distribution by a procedure known as Gibbs sampling (e.g., MacKay 2004). This sampling approach, detailed in Appendix E, produces a set of samples $\{\mathbf{w}^n, \mathbf{b}^n\}$, which can be used to derive estimates for the error bars empirically, by finding the pointwise standard deviations of these samples around $\mathbf{w}_{\mathrm{MAP}}$ and $\mathbf{b}_{\mathrm{MAP}}$. Some precautions are needed, described in the next section, to ensure that these estimates are correct. Also note that the estimated error bars for $\mathbf{b}$ are not the error bars on the input nonlinearity $f(\cdot) = \mathsf{F}\mathbf{b} = \sum_i b_i f_i(\cdot)$, where F is the matrix or operator containing the basis elements $f_i(\cdot)$: if $\mathsf{\Sigma}$ is the sample covariance matrix of the Gibbs samples $\{[\mathbf{b}^n]_i\}$ around $[\mathbf{b}_{\mathrm{MAP}}]_i$, then the error bar on $f(x)$ at position x is given by the formula $(\mathsf{F}\mathsf{\Sigma}\mathsf{F}^\mathsf{T})^{1/2}_{x,x}$.

The error bars for the parameters of the full-rank model can be estimated in the same way as for a linear model, but once again they must be converted to error bars on the receptive field. The posterior covariance matrix is $\mathsf{\Sigma} = \hat{\sigma}^2(\mathsf{M}^\mathsf{T}\mathsf{M} + \hat{\sigma}^2\mathsf{D}^\mathsf{C})^{-1}$ (see "Regularization"; M is here the data array and D^C the regularisation array, both with (τ, i) vectorised so that they become matrices). Defining an appropriate basis matrix F similar to the above, the error bars are now $(\mathsf{F}\mathsf{\Sigma}\mathsf{F}^\mathsf{T})^{1/2}_{x\tau,x\tau}$.

Degeneracies

Bilinear model. It is possible for multiple settings of the parameters in a model to be degenerate; that is, for models with those different parameters to produce predictions that are identical in all respects. If such degenerate sets of parameters exist, the model is said to be nonidentifiable: even if infinite data were available, the parameters of the model could not be identified uniquely. The bilinear model contains such degeneracies: the model does not change if $\mathbf{w} \to \lambda \cdot \mathbf{w}$ and $\mathbf{b} \to 1/\lambda \cdot \mathbf{b}$, because the parameters appear only in the product $\mathbf{w}\mathbf{b}^\top$. As we are generally interested in the shape, but not absolute scale, of $\mathbf{w}$ and $\mathbf{b}$, this nonidentifiability need not pose a serious problem to interpretation of the estimated parameters.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

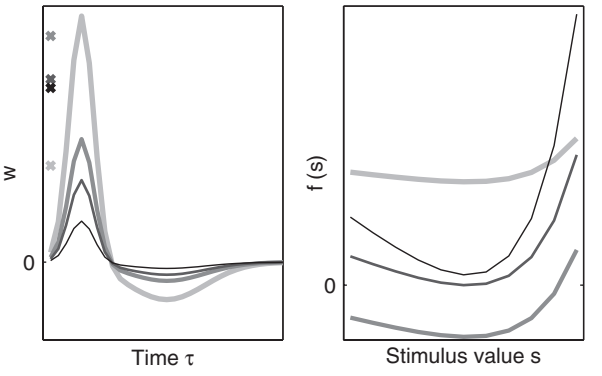


Figure 4. Degeneracies of the bilinear model. Different line styles represent different models, which are all equivalent. The crosses on the left represent the constant terms $c = w_1 \cdot b_1$.

However, it may lead to overestimation of the error bars. Error bars indicate the range of values for which the firing rate predictions of the model are consistent with the observed data; if any change in a parameter can be countered by changes in other parameters so that $\hat{\mathbf{r}}$ is left unchanged, the error bars may potentially be infinitely large (if there is no prior information to constrain the parameter estimates). Probabilistically, such degeneracies induce directions in parameter space for which the likelihood does not change. The Gibbs sampling procedure will draw samples along these directions, and thus the empirical estimates of parameter variance will be overestimated. Figure 4 illustrates this issue by showing four equivalent configurations of the bilinear model.

The bilinear model has a second degeneracy: a constant (say d) added to each element of $\mathbf{b}$ is interchangeable with a change in the constant c:

$$c + \sum_{\tau i} w_\tau (b_i + d) M_{t\tau i} = c + \sum_{\tau i} w_\tau b_i M_{t\tau i} + d \cdot \sum_{\tau i} w_\tau M_{t\tau i}.$$

Here the last term is constant with respect to $\mathbf{b}$:

$$\sum_{\tau i} w_\tau M_{t\tau i} = \sum_{\tau} w_\tau \sum_{i} f_i(s(t-\tau)) = \sum_{\tau} w_\tau \cdot 1, \tag{5}$$

because the piecewise linear basis functions sum to 1 at every point, $\sum_i f_i(x) = 1$. (A number of commonly used bases share this property, e.g., the piecewise linear, discrete, and spline bases. If the basis functions do not sum to 1, the degeneracy will persist in general, but the requisite transformation of $\mathbf{b}$ will be more complicated than a constant offset as in this case.) Thus, an addition of d to each element of $\mathbf{b}$ can be countered by subtracting $d \cdot \sum_\tau w_\tau$ from c. The equivalence of the models thus obtained may again lead to misleadingly large error bars on $\mathbf{b}$ and c. This additive degeneracy is also illustrated in Figure 4.

One way to remove the additive degeneracy and restore the identifiability of the model, is to remove one function f_j from the basis set $\{f_i(\cdot)\}$ before constructing the

Downloaded By: [Columbia University] At: 22:13 19 June 2008

data array M: this forces $f(\cdot) = \sum_{i \neq j} b_i f_i(\cdot)$ to be zero at a specific point (it is zero at the j^{th} node, $f(x_j) = 0$; see Appendix A); Equation 5 does not then hold because $\sum_{i \neq j} f_i(\cdot) \neq 1$. Thus, the additive degeneracy disappears and the Gibbs samples cannot vary in offset. (Another approach to removing the additive degeneracy in the case of the bilinear model would be to remove the constant offset c from the model entirely, absorbing the offset into **b**. However, this would not remove a similar degeneracy of the full-rank model; see below.) The multiplicative degeneracy can be fixed after convergence, by rescaling the Gibbs samples to minimize the squared distance between the samples and $\mathbf{b}_{\text{MAP}}$. The inverse rescaling is then applied to the corresponding **w** samples.

Full-rank model. The full-rank model is also non-identifiable for basis vectors that sum to 1 at each point: changing $C_{\tau i}$ to $C_{\tau i} + d_\tau$, for any vector **d** whose elements sum to zero, gives $\hat{r}(t) \to \hat{r}(t) + \sum_\tau d_\tau \sum_i f_i(s(t-\tau)) = \hat{r}(t) + \sum_\tau d_\tau \cdot 1 = \hat{r}(t)$. That is, there is no change in the input–output relation of the model, but the receptive field does change: $a(\tau,x) = \sum_i C_{\tau i} f_i(x) \to a(\tau, x) + d_\tau$. Once again, this can lead to overestimation of the error bars. Fortunately, this problem can be tackled in the same way as the additive degeneracy in the bilinear model: remove one of the basis functions f_j from the basis set $\{f_i\}$, so that $\sum_{i \neq j} f_i(\cdot) \neq 1$ and the degeneracy disappears.

Rank-*k* models. In the specific instance of a rank-k model, a further degeneracy arises from the model invariance under a permutation of the k parameter vectors. In the example given previously, any permutation of the indices 1,2,3 of $\mathbf{w}^{1,2,3}$ and $\mathbf{b}^{1,2,3}$ leads to the same model. This degeneracy generally poses no problem for error bar evaluation, because the Gibbs sampler will rarely jump between such equivalent parameter arrangements, these being separated by regions of parameter space with very low probability.

Output nonlinearity

It may be helpful at times to extend the input nonlinearity model to also include an output nonlinearity; one clear example might be if the input nonlinearity model by itself predicted negative rates at many times. One approach, although suboptimal, is to first fit a bilinear or full-rank model to give an estimated firing rate $\hat{r}(t)$, and then find a pointwise function $g(\cdot)$ so that $g(\hat{r}(t))$ is a better estimate of $r(t)$ than was $\hat{r}(t)$ alone. One strategy for identifying g, described by Chichilnisky (2001), is to find the average number of spikes that are elicited when $\hat{r}$ falls within a certain interval; $g(\cdot)$ is then defined to take this average value on that interval. This strategy is equivalent to the basis function expansion used for the input nonlinearity, expressing the function g in a discrete basis $g(\cdot) = \sum_j d_j g_j(\cdot)$, and fitting to the data $(\hat{r}(t), r(t))$. The vector of weights **d** is fit by linear regression, just as was **b** in the bilinear model. In place of a discrete set of basis functions, one can choose any set; here, we use piecewise linear basis functions with the same type of regularising prior as was used for the input nonlinearity. The fitting of the output nonlinearity is just one-dimensional nonlinear regression (Suits et al. 1978), and may be applied easily to either the bilinear or full-rank models.

As described, the bilinear model parameters are fit *before* the output nonlinearity is found, and will generally not be optimal. The optimization could be extended by minimizing the squared error $\mathcal{E} = \sum_t |r(t) - \sum_j d_j g_j(\sum_{\tau i} w_\tau b_i M_{t\tau i})|^2$ with respect to

Downloaded By: [Columbia University] At: 22:13 19 June 2008

($\mathbf{d}, \mathbf{w}, \mathbf{b}$), using e.g., gradient descent techniques. On one model data set, generated using a sigmoid g, we found that although this further optimisation did indeed decrease the squared error, most of the benefit had already been achieved by fitting g after the other parameters had been fixed (data not shown). Another option is to fix g instead of inferring it, and optimise $\mathbf{w}$ and $\mathbf{b}$. This leads to a generalized bilinear model, which is discussed in the following section in the context of models for single spike trains, rather than of models for average rates. The developments of that section can be applied to PSTH models as well; with the negative squared error $-\mathcal{E}$ (and its corresponding derivatives) taking the place of the log-likelihood (and its corresponding derivatives) defined below.

Fitting spike trains: Generalized bilinear or NLNP models

The discussion thus far has dealt with the use of input nonlinearity models to fit spike rates, by minimizing the squared-error objective function between the predicted rate and the average time-binned spike counts over multiple repetitions of the same stimulus. Similar models can also be used to fit the probability of spiking in a single trial. This requires two extensions. First, a cost function more appropriate to spike-time data must be adopted, and second, the predictions of the models must be constrained to be probabilistically meaningful. This section thus develops the theory of the input nonlinearity models in the context of (a) a point-process likelihood and (b) a fixed output nonlinearity. As mentioned in the preceding section, the introduction of the fixed output nonlinearity may also be relevant in the context of mean-rate prediction models.

A cost function appropriate to spike-time data is suggested by the theory of point processes, although the continuous-time form must be discretised in practice. Consider a spike train in which the individual spikes occur at times $\{t_k\}$. Let the value of the model prediction, $\hat{r}(t)$, give the expected number of events in the time bin centred at time t; that is, the instantaneous spike rate in a bin of width $\mathrm{d}t$ is predicted to be $\hat{r}(t)/\mathrm{d}t$. Then, if we write $\hat{r}(t_k)$ for the predicted count in the bin in which the kth actual spike falls, the point-process log-likelihood can be approximated (neglecting constant terms; Berman and Turner 1992):

$$\mathcal{L} = \sum_k \log \hat{r}(t_k) - \sum_t \hat{r}(t)$$

In this expression, $\hat{r}(t)$ must be positive at the spike times t_k, and more generally – for instance, if the likelihood of model is to be evaluated on cross-validation data – everywhere. Thus, we consider here models that include an output nonlinearity, taking the form $\hat{r}(t) = g(\sum_\tau w_\tau f(s(t-\tau))) = g(\sum_{\tau i} M_{t\tau i} w_\tau b_i)$ (bilinear) or $\hat{r}(t) = g(\sum_{\tau i} M_{t\tau i} C_{\tau i})$ (full-rank), with g a fixed function (the link function, or output nonlinearity), and the constant offset c is incorporated into M as usual. We do not attempt to estimate g as in the previous section. The argument of g is thus either bilinear or linear. Commonly, a linear model with output nonlinearity, and exponential family likelihood, is called a Generalized Linear Model (GLM; McCullagh and Nelder, 1989) or Linear-Nonlinear-Poisson Model (LNP; Simoncelli et al. 2004); thus the bilinear version is a Generalized Bilinear

Downloaded By: [Columbia University] At: 22:13 19 June 2008

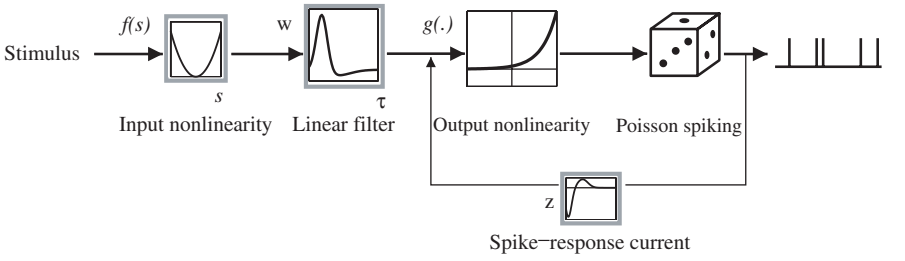


Figure 5. Schematic view of the bilinear model with output nonlinearity. The parameters in the grey boxes are learnt from the data. The model may be used with or without the spike-response term.

Model or NLNP model (the first N standing for nonlinear) as depicted in Figure 5, while the full-rank version is a GLM.

The parameters of these models are as before, i.e. $\mathbf{w}$ and $\mathbf{b}$, or C, and must be found by maximizing $\mathcal{L}$. In this case, there is no analytical solution that maximises $\mathcal{L}$ even for the full-rank model, and so the optimal parameters must be found by iterative methods such as gradient ascent. For the full-rank model, general results for GLMs apply, and a unique optimum of $\mathcal{L}$ is guaranteed if g is convex and log-concave (Paninski 2003, 2004).

For the bilinear model, defining for brevity $y(t) = \sum_{\tau i} M_{t\tau i} w_\tau b_i$ so that $\hat{r}(t) = g(y(t))$, the gradient with respect to $\mathbf{b}$ is given by

$$\frac{\partial \mathcal{L}}{\partial b_j} = \sum_k \frac{g'(y(t_k))}{g(y(t_k))} \sum_\tau M_{t_k \tau j} w_\tau - \sum_t g'(y(t)) \sum_\tau M_{t\tau j} w_\tau$$

with a similar expression for $\partial \mathcal{L}/\partial w_\tau$. Note that this gradient simplifies when g is the exponential function as then $g'(y)/g(y) = 1$. The Gaussian priors over $\mathbf{b}$ and $\mathbf{w}$, introduced for regularisation, are still applicable. Incorporating these, the objective function becomes the log-posterior $\mathcal{L} - (1/2)\mathbf{w}^{\mathsf{T}}\mathsf{D}^{w}\mathbf{w} - (1/2)\mathbf{b}^{\mathsf{T}}\mathsf{D}^{b}\mathbf{b}$, and the gradient with respect to $\mathbf{b}$ becomes $\partial \mathcal{L}/\partial b_j - [\mathsf{D}^{b}\mathbf{b}]_j$ (with the expression for the $\mathbf{w}$ gradient being similar), where D^{w} and D^{b} are the regularisation matrices described above.

Note that if g is convex and log-concave (e.g., $\exp(y)$ or $\log(1 + \exp(y))$), the property of the squared-error case that the objective function is concave in each parameter vector alone (for a fixed setting of the other) is preserved – i.e., if $\mathbf{b}$ is fixed, then the objective is concave in $\mathbf{w}$ and thus has a unique optimum (Paninski 2004). This implies that the model can easily be fit by alternating maximization as before – that is, switching between maximizing $\mathcal{L}$ with respect to $\mathbf{w}$ keeping $\mathbf{b}$ fixed, and with respect to $\mathbf{b}$ keeping $\mathbf{w}$ fixed. In our experience, this alternating maximization appears to converge more rapidly than simultaneous gradient ascent with respect to $\mathbf{w}$ and $\mathbf{b}$ jointly. In addition, it is possible to adapt the iteratively reweighted least squares (IRLS) algorithm, conventionally used for GLMs, to an alternating form suitable for the generalized bilinear model.

One of the main reasons to use spike-time data rather than average spike rates is that the spike history of the neuron can be incorporated in the model in a

Downloaded By: [Columbia University] At: 22:13 19 June 2008

natural way (Paninski 2004). Spike history effects, such as refractory periods and self-excitation, are often modeled as an additive feed-back influence called the ''spike-response current'': after each spike, a term **z**, varying over a small interval of $\mathcal{J}$ time bins, is fed back, so that j time steps after the last spike, the probability of spiking is changed by an amount z_j. Incorporating such a term into the bilinear model gives

$$\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i} + \sum_{j=1}^{\mathcal{J}} z_j \rho(t-j),$$

where $\rho(t)$ is a binned representation of preceding spikes; that is, $\rho(t) = 1$ if a spike occurred in the bin centred at t, and is 0 otherwise. Crucially, the prediction $\hat{r}(t)$ depends only on value of $\rho(\tau)$ for $\tau < t$. When finding parameter values, or for cross-validation, ρ is set to the actual spike train observed. When generating predicted spike trains, it may be sampled from rates predicted in preceding time bins. The term **z** may be estimated together with **w** and **b** from the data, by incorporating the observed spike train into the stimulus array M as follows. If the data array M is initially of size $T \times A \times B$, we define $M_{t,A+j,B+1} = \rho(t-j)$ for $j = 1, \ldots, \mathcal{J}$. Estimation now proceeds as before. At convergence, the spike-response term is given by $z_j = w_{A+j} \cdot b_{B+1}$. Thus, the spike history becomes part of the data array and is treated as a ''stimulus'', although it is not mapped through an input nonlinearity, unlike the true stimulus $s(t)$ (cf. ''Models with multiple stimulus features''). Note that the log-likelihood is generalized linear (and concave, for suitable g) in each pair (**w**, **z**) and (**b**, **z**) simultaneously, and so it is straightforward to estimate **w** and **z** together, given **b**, or conversely **b** and **z**, given **w**.

Experiments on model data

The first set of experiments was carried out with three model data sets. Each data set was constructed as follows. A one-dimensional stimulus $s(t)$ was generated as Gaussian white noise with unit variance, and transformed into a rate $P(t)$ by three different functions, described below. Any negative values of $P(t)$ were set to zero. Five spike trains, corresponding to five repeated ''trials'' of the experiment, were then generated from $P(t)$. A small amount of uniform noise was added to $P(t)$ to reflect non-stimulus locked ''internal processes'', and the resultant signal taken to give the probability of observing a spike in each time bin. The observed firing rate $r(t)$ was obtained by averaging these five spike trains. Thus, the variability of each spike train around $P(t)$ was Poisson-like with a slightly increased variance. The first half of the spike rate and stimulus was used to train the various input nonlinearity and full-rank models. These were then tested on the second half, with their performance being quantified by their predictive power (Sahani and Linden 2003b). This is a performance measure based on the squared error, which takes into account trial-by-trial variability of the response. It has an expected value of 1 for a model that captures all *predictable* fluctuations in the firing rate, and 0 for a model which predicts only the mean. The duration of a trial varied between 300 and 100 000 time points, so as to study how overfitting in the various models depended on the amount of data available.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

The rates $P(t)$ were generated according to the following three processes:

(I) One filter. $P(t) = \sum_{\tau=0}^{\tau_{\max}} k(\tau)s(t-\tau)^2$. That is, stimulus values were squared and then linearly filtered by $k(\tau)$. The weights $k(\tau)$ were non-negative, so that $P(t)$ did not need to be rectified.

(II) Two filters. $P(t) = \left[\sum_{\tau=0}^{\tau_{\max}} k_1(\tau)s(t-\tau) + \sum_{\tau=0}^{\tau_{\max}} k_2(\tau)s(t-\tau)^2\right]^+$. That is, the firing rate is a sum of two linear temporal filtering operations, one acting on the stimulus values, and one acting on their squares. Both filters had non-negative weights, with peak values at different τ. However, negative stimulus values could lead to negative filter outputs, and so $P(t)$ was rectified, as indicated by the brackets $[\cdot]^+$.

(III) Nonlinear feature selective process. Here, the spike rate depended on how closely the recent stimulus approximated a ''sweep'' in stimulus space. Defining **a** to be a vector with evenly spaced increasing values between $\min(s)$ and $\max(s)$,

$$P(t) = c + d \cdot \sum_{\tau} \mathcal{I}(|s(t-\tau) - a(\tau)| < e),$$

with c and d chosen so that $0 \le P \le 1$, and $e = 0.2 \cdot (\max(s) - \min(s))$; $\mathcal{I}$ is the indicator function, equal to 1 when its argument is true and 0 otherwise.

The third process was included to investigate the behaviour of the bilinear and full-rank models when they are fit to data they cannot entirely capture (other processes that could generate such data include, for instance, multiplicative combinations of two different stimulus features). With a bin size defined to be 15 ms, the average firing rate of processes I–III was about 33 Hz. We trained and tested the above models on the following datasets, using minimal regularisation:

1. bilin(1). A one-term bilinear model (i.e., $\hat{r}(t) = c + \sum_\tau w_\tau f(s(t-\tau))$) fitted through the ALS procedure.
2. bilin(2). A two-term bilinear model ($\hat{r}(t) = c + \sum_\tau \left(w_\tau^1 f^1(s(t-\tau)) + w_\tau^2 f^2(s(t-\tau))\right)$), estimated by the method described in section ''Rank-k models''. This model is also combined with a static output nonlinearity in some cases.
3. Full-rank. A full-rank model ($\hat{r}(t) = c + \sum_\tau w_\tau f_\tau(s(t-\tau))$).
4. Full-rank SVD. The leading SVD terms of a full-rank model. This has the same structure of a multi-term input nonlinearity model, but the estimation of the terms is suboptimal. An SVD term was included when its eigenvalue was larger than 1/4 of the leading eigenvalue.

Figure 6 shows the singular values of the SVD of the full-rank models when trained on each of the three spike rates I–III. The singular value spectrum is informative about the nature and complexity of the underlying process: the single temporal filter (I) causes one eigenvalue to dominate all others, while the nonlinear feature selection process (III) results in a gently decreasing spectrum. The two-filter process (II) can be identified in the spectrum by the two dominant singular values. Note that one or a few isolated values are not necessarily the signature of a simple underlying process: there can be complex processes that do not leave their print on the singular value spectrum of a full-rank model. In that case their presence has to be discovered through the predictive performance of the models.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

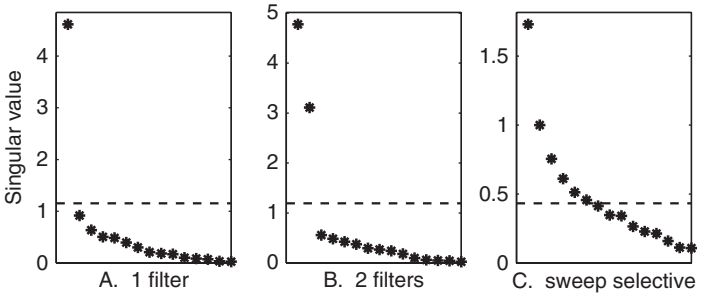


Figure 6. Singular values of the SVD decomposition of the full-rank model trained on model data A–C. Only the first 15 eigenvalues are shown. Note that in these examples, the number of significant singular values are indicative of the complexity of the spike generating process (dashed lines show the threshold for inclusion in the Full-rank SVD model).

What do the models look like? Figure 7 shows the models learnt for process II. The bilinear model correctly picks out (noisy versions of) the two temporal filters and input nonlinearities. The full-rank model also identifies these terms, but the two leading terms of the SVD show some mixing of the linear and quadratic input transforms, and the predictive power is lower than that of the bilinear model. Fitting to process III (not shown) yields a full-rank model with a diagonal excitatory band in the weight matrix. The bilinear model approximates ''sweep selectivity'' in a similar way, by having biphasic parameter vectors whose outer product shows some diagonal structure.

To compare the predictive performance of the various models, their predictive powers are shown as a function of the length of the trial in Figure 8. These graphs confirm our intuitions:

A. All of the models can capture the structure of dataset I. Thus, the performance is governed by the degree of overfitting: the model with fewest parameters, bilin(1), does best. For large data sets the performance of the different models converges. Note that the SVD model performs worse than the bilin(2) model (even when two or fewer SVD terms are used, as was always the case for trial lengths longer than 30 s).
B. The bilin(1) model cannot capture dataset II, as it was generated from a bilin(2) model. The models that are able to capture the data show a similar ordering of performance as in A. The performance of the bilin(2) model can be improved by an output nonlinearity (oNL), as this allows it to capture the rectification in the generative process.
C. None of the models can capture the structure of data set III, but the full-rank models come close. Here, the simpler models perform better at small data volumes, as they overfit less, but the more complex models perform better as more data become available.

Finally, we fitted an NLNP model, or a generalized bilinear model, to a spike train, rather than the spike rate, using the point process likelihood. To generate the spikes, we used a quadratic input nonlinearity $f(s) = s^2$ and an exponential output nonlinearity $g(\cdot) = \exp(\cdot)$, so the model for the underlying spike rate was $r(t) = \exp(c + \sum_\tau w_\tau s(t-\tau)^2)$. With time bins of 15 ms, the duration of the spike

Downloaded By: [Columbia University] At: 22:13 19 June 2008

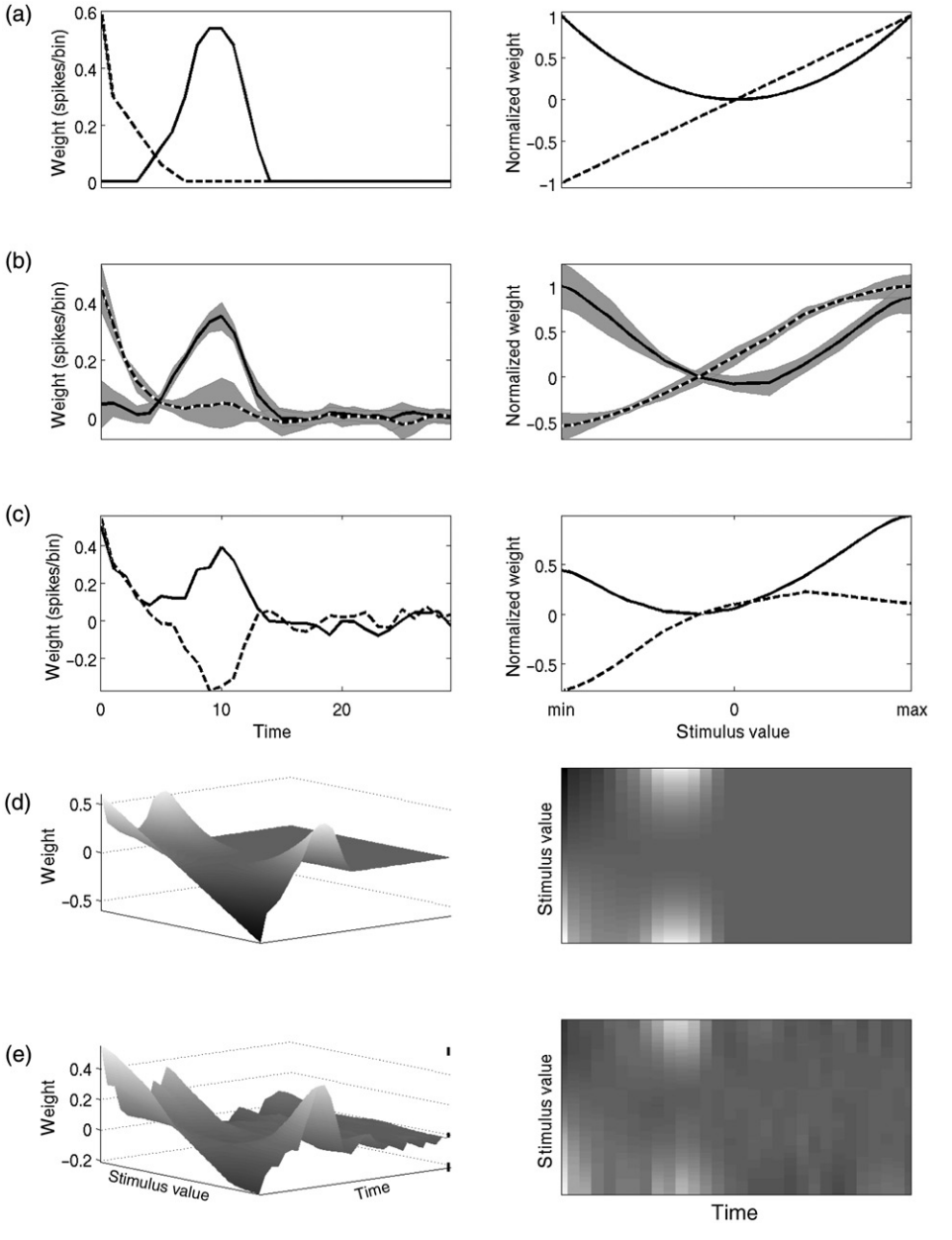


Figure 7. Model fits to data generated from the two-filter process (II). (a): true temporal filters $\mathbf{w}^1$ and $\mathbf{w}^2$ (left) and input nonlinearities f^1 and f^2 (right). (b): estimated bilinear model with two terms. The error bars of one standard error (grey) are calculated by Gibbs sampling. (c): first two SVD terms of the estimated full-rank model. (d): the true full-rank model given by $\mathsf{C} = \mathbf{w}^1\mathbf{b}^{1T} + \mathbf{w}^2\mathbf{b}^{2T}$, shown as a surface (left) and as a matrix (right). (e): the estimated full-rank model. Error bars of one standard error at the maximum and minimum values of the receptive field are shown as lines on the top and bottom right of the surface plot, and the average error in the middle right, and are obtained as in linear regression.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

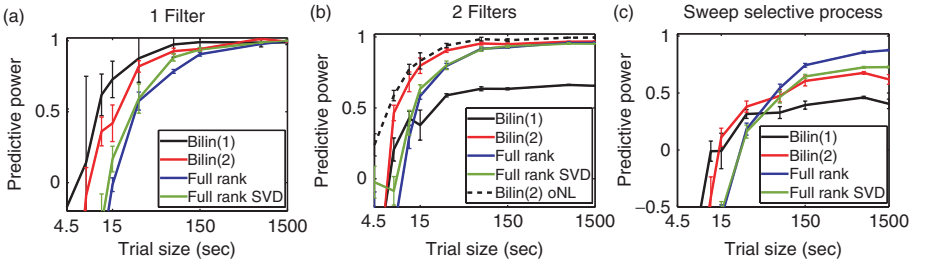


Figure 8. Performance of the models on datasets I–III, averaged over 10 instantiations of the random stimulus. The models were fitted to the firing rate over five trials; thus, the experiment length is five times the trial size. Details and interpretations of the plots can be found in the main text.

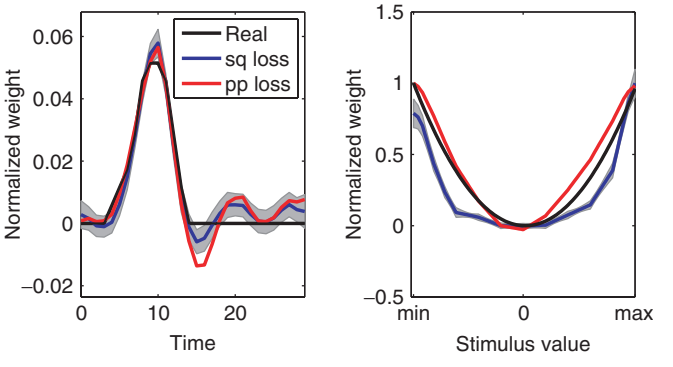


Figure 9. Point process model fits. Black: true filters. Blue: filters found through minimising the squared error, with error bars in grey. Red: filters found by maximising the point-process likelihood, using the exponential output nonlinearity. The normalised shapes of the filters are similar, but the input nonlinearity under the squared error appears to be somewhat biased towards a U-shape.

train was 150 s and the mean firing rate 0.7 Hz, with 106 spikes observed in the simulation. Performing alternating gradient ascent on the objective function, which included regularisation terms, resulted in the parameters shown in Figure 9; the fit is reasonable considering little more than 100 spikes were used for it. Real data will generally be noisier, and not generated by an NLNP system, so that 100 spikes are unlikely to suffice for fitting the model to real systems. For comparison, this figure also shows a fit of a bilinear model (using as its objective function the squared error between the predicted firing rate and the observed binary spike train). The parameters of this bilinear model are similar in shape to the true parameters, though they appear to be biased. In this example, no spike-response current was used. The next section includes an NLNP model fitted to real data.

Demonstration on real data

We estimated the full-rank model (using the velocity signal as the stimulus) and a bilinear model (with two terms, position and velocity) using spike-rate data from

Downloaded By: [Columbia University] At: 22:13 19 June 2008

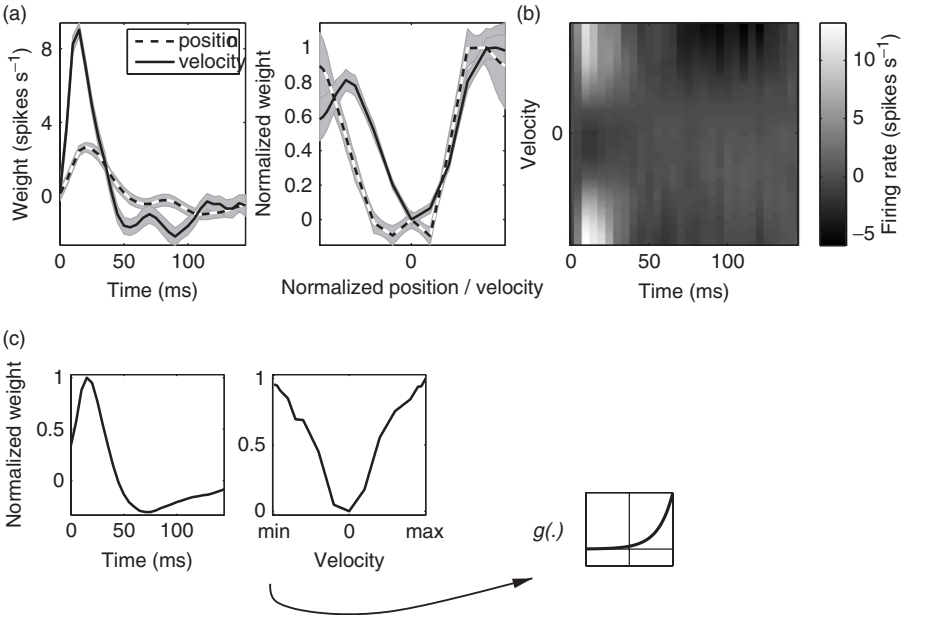


Figure 10. Bilinear and full-rank models applied to responses from a rodent somatosensory cortex neuron. (a): Input nonlinearity model. Left: temporal filters for position and velocity, right: the corresponding input nonlinearities. The grey areas show one standard error, obtained by Gibbs sampling. No output nonlinearity was used as this did not significantly improve the predictions. (b): Full-rank model on velocity. The full-rank model shares most features of the bilinear model, such as direction invariance. (c): The NLNP model, with exponential output nonlinearity. Only the velocity feature of the stimulus was used for this model. The filters have similar shapes to the velocity filters of the bilinear model.

a cell in rodent barrel cortex, stimulated by a white noise whisker displacement stimulus. These models were fitted using a PSTH with 3000 time bins of 5 ms. We also fitted an NLNP model to the spike times, using approximately 12 000 spikes collected over about half an hour. The resulting models are shown in Figure 10. They recover a direction invariant response to velocity. In the bilinear model, this can be seen from the approximately symmetric shape of the input nonlinearities (right panel). In the full-rank model, this is evident from the symmetry of the receptive field about the zero velocity line. Also, the shapes of the temporal filters of the bilinear model (left panel) indicate that this cell is more responsive to the velocity than to the position of the whisker: since both input nonlinearities have been normalized, the size and shape of the temporal filter of a certain feature (position or velocity) is an indication of how much variance in the spike rate that feature predicts. These observations are in agreement with previous results (Pinto et al. 2000; Arabzadeh et al. 2003, 2005). The NLNP model recovers similar filters as the other models, and demonstrates the feasibility of estimating this model on spike time data.

Although the models for the PSTH are structurally only a small departure from the linear model, their predictions are far superior. The predictive power of a linear model is 0.01 on training data and negative on cross-validation data, whereas the predictive powers of the input nonlinearity and full-rank models are 0.6 and 0.54,

Downloaded By: [Columbia University] At: 22:13 19 June 2008

respectively (on cross-validation data). Clearly, the reason for this is the parabolic shape of the input nonlinearity. Note that, even though the full-rank model yields a lower cross-validation predictive power, we cannot say how much its performance suffers from over-fitting and therefore cannot be sure that the underlying system is exactly bilinear. However, of the two sets of parameters fitted to the limited data available, those of the bilinear model appear to be a closer match to the true system.

Discussion

The bilinear model (also called the Hammerstein or NL cascade model) and the full-rank model provide useful nonlinear approaches to describing a neuron's time-varying response to a stimulus. The estimation of parameters in these models is relatively straightforward; the discussions of degeneracies and regularisation methods presented in this article allow for a careful analysis of the model parameters and their error bars. The small number of parameters in the bilinear and NLNP models ($\dim(\mathbf{w}) + \dim(\mathbf{b})$ parameters) makes the data requirements modest, while the full-rank model requires more data for estimation (it has $\dim(\mathbf{w}) \cdot \dim(\mathbf{b})$ parameters). The feasibility of fitting, and potential utility of, the models has been demonstrated on model data and on data from rodent barrel cortex.

Relation to other methods

A successful nonlinear model of similar flexibility to the bilinear model is the Linear-Nonlinear-Poisson model (LNP; e.g., Simoncelli et al. 2004). The LNP model has several variants. For example, the output nonlinearity might be fixed and the temporal filter estimated by gradient ascent on the point-process likelihood (Paninski 2004). Such an LNP model is a special case of the NLNP model, in which the input nonlinearity is the identity function. Other variants of the LNP model incorporate non-parametric output nonlinearities, and may be estimated by Spike-Triggered Covariance analysis (De Ruyter van Steveninck and Bialek 1988; Schwartz et al. 2002), a powerful method for finding relevant directions ($\mathbf{w}$ vectors) in stimulus space. This technique's provable accuracy is limited to the case where the stimulus is Gaussian (Paninski 2003) (though interesting and useful results have been obtained using non-Gaussian stimuli; Touryan et al. 2005), and not very high dimensional (as estimating the spike-triggered covariance involves identifying order $\dim(\mathbf{w})^2$ parameters). STC analysis may automatically find multiple relevant stimulus representations and can also be used to construct models with nonlinear stimulus-stimulus interactions such as divisive normalisation. The special case of an LNP model using just one STC vector is similar to the bilinear model, but with the ordering of linear and nonlinear operations reversed. LNP models, bilinear models and full-rank models are likely to have regions of overlap in terms of the types of neurons that can be successfully modeled. Although each model has its own benefits and disadvantages in terms of data requirements, ease of estimation, etc., in the end, the neuron under investigation determines which model provides the most appropriate description (as measured by the predictive performance on cross-validation data).

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Another useful method is Wiener–Volterra systems identification, a classical non-linear estimation method which has been in use in neuroscience for a long time (e.g., Marmarelis and Naka, 1973). More recently it has found applications in, for example, characterizing subthreshold dynamics in barrel cortex (Webber and Stanley, 2004). Since in theory these expansions span all non-linear models, the models introduced in this article can also be phrased as restricted Volterra–Wiener expansions (e.g., the bilinear model would become $\hat{r}(t) = c + \sum_{\tau j} w_\tau b_j [s(t-\tau)]^j$, in which the input nonlinearity is expressed as a power-series expansion, $f(x) = \sum_j b_j x^j$). In practice, this method is most suitable when an appropriate reduction in the parameter space can be identified (e.g., Young and Calhoun 2005); otherwise, the number of parameters tends to be too large for such models to be practical (see also Juusola et al. 1995 for a comparison between Volterra series and cascade models).

Several estimation methods for Hammerstein cascades and related models have been previously proposed. Narendra and Gallman (1966) use an algorithm similar to alternating least squares, but with an approximation to a more general temporal filtering component based on the *pulse transfer function*. Westwick and Kearney (2001) use a gradient based method with a similar structure to alternating least squares. Bai (1998) estimates a suboptimal bilinear model by taking the first SVD component of the corresponding full-rank model. Correlation-based methods (e.g., Spekreijse and Oosting 1970) rely on Bussgang's theorem (Bussgang 1952) and only provably work if the system really is a Hammerstein system, whereas methods relying on an objective function – such as the squared error – do not make such assumptions about the system and merely try to find the best-fitting Hammerstein approximation. For small systems, with restrictions on the form and number of basis functions, there also exists a method for finding a closed-form solution for the parameters of a bilinear model (Korenberg, 1991). The above methods generally do not incorporate regularisation techniques or probabilistic interpretations of the models.

Probabilistic interpretations

The noise models that were assumed for the bilinear and full-rank models (Gaussian or Poisson noise) gave them a probabilistic interpretation, allowing for principled regularisation techniques and error bar estimation. Finding error bars for the full-rank model requires a single operation (as for linear models); for the bilinear model, error bars may be estimated through Gibbs sampling. Obtaining error bars for the point-process model is slightly more computationally intensive (since we need to employ a Metropolis-Hastings step (MacKay 2004) to sample from the posterior distributions), but this is still tractable (Rigat et al. 2006; Cronin et al. 2006); in addition, bootstrap techniques are available, where error bars are derived from repeated estimation of the models on datasets randomly reselected from the available data (Effron and Tibshirani, 1993).

Bilinear model, full-rank model and SVD components

In the experiments on simulated data, the bilinear model outperformed the SVD decomposition of the full-rank model. This was expected, as SVD minimizes the distance between the SVD terms and the parameters of the full-rank model, while

Downloaded By: [Columbia University] At: 22:13 19 June 2008

the bilinear model directly minimizes distance between the real and predicted firing rate. However, if it is not too computationally expensive to estimate the full-rank model, the first SVD component can serve as a good initialization for the estimation procedure of the bilinear model (or the first k SVD terms can initialize the ALS estimation of a rank-k model). The full-rank model was shown to capture more complex dynamics when there was enough data available for its estimation.

Extensions of the bilinear model

The dimension of lag time, called τ, does not have to range over lag time only, but can also range over other stimulus features. In audition, for example, it might range over lag time and frequency; the input nonlinearity would then apply to sound level (Ahrens et al. 2008). In vision, τ could be used for time and space, and the input nonlinearity for luminance. Another extension of the bilinear model is the *multilinear* model (Ahrens et al. 2008). An example of a multilinear model is a trilinear model, in which the first two components act as a bilinear model as presented in this article, and the third component multiplicatively modulates the predicted firing rate as a function of the stimulus. The extra components of multilinear models can be used to capture further nonlinear phenomena such as short-term stimulus specific adaptation effects, while maintaining a small and tractable number of parameters. Other extensions of the bilinear model involve learning the basis functions, e.g., the position of the nodes of a spline basis, by adding a nonlinear step to the ALS estimation procedure (Westwick and Kearney, 2001). Finally, in this article we assumed discrete basis functions in the τ direction. Other basis sets may also be used, in which case the bilinear model would become $\hat{r}(t) = c + \sum_{ij} b_i d_j \sum_\tau h_j(\tau) f_i(s(t-\tau))$, with $\{h_j\}$ the basis functions in the τ direction. This model is still bilinear and therefore all previously presented estimation techniques go through, noting that now the prior for $\mathbf{d}$ has the same form as the prior for $\mathbf{b}$.

Acknowledgments

We thank Rasmus Petersen for the data used in the example of Figure 10 and for interesting discussions, Zoubin Ghahramani for suggestions, and Quentin Huys for comments on the manuscript. M.B.A. and M.S. were funded by the Gatsby Charitable Foundation and L.P. was funded by NEI grant EY018003 and by a pilot grant from the Gatsby Charitable Foundation.

Appendix

A. Piecewise linear and discrete bases

The piecewise linear basis $\{f_i(x)\}_{i=1}^N$ consists of tent-shaped functions determined by a set of nodes $\{x_q\}_{q=1}^N$:

$$f_i(x) = \begin{cases} (x - x_{i-1})/(x_i - x_{i-1}) & \text{if } i > 1 \text{ and } x_{i-1} \le x < x_i \\ (x_{i+1} - x)/(x_{i+1} - x_i) & \text{if } i < N \text{ and } x_i \le x < x_{i+1} \\ 0 & \text{otherwise.} \end{cases}$$

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Piecewise linear basis functions have the advantage over polynomial basis functions of being local. Polynomial basis functions are global and noise can cause large and uncontrolled fluctuations in parts of the fitted function that are not tightly constrained by the data. A reasonable alternative choice might have been a polynomial spline basis (Hastie et al. 2001).

The discrete basis is also defined by a set of nodes $\{x_q\}_{q=1}^{N+1}$, but now

$$f_i(x) = \begin{cases} 1 & \text{if } x_i \le x < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

If the stimulus takes on discrete values, then the basis can be simply defined as: $f_i(x) = 1$ if x takes on the i^{th} stimulus value, and zero otherwise.

To compute error bars on the parameters of the models, it is necessary to remove one basis function from the basis set in order to avoid degeneracies, as explained in section ''Degeneracies''.

B. Alternative alternating least squares procedure

When $\dim(\mathbf{w})$ and $\dim(\mathbf{b})$ are small or the number of time points T is big (specifically, when $T > (\dim(\mathbf{w})^2 + \dim(\mathbf{b})^2)/2$) then the estimation of the bilinear model can be accelerated through the use of different arrays. Note that at each iteration of the algorithm presented in the main text, the matrix $B_{t\tau} = \sum_i b_i M_{t\tau i}$ is redefined and used to estimate $\mathbf{w}$ through $\mathbf{w} = (\mathsf{B}^\top \mathsf{B})^{-1} \mathsf{B}^\top \mathbf{r}$. That is, it appears as $\mathsf{B}^\top \mathsf{B}$ and as $\mathsf{B}^\top \mathbf{r}$. Both of these terms include a sum over t and this may be expensive (both in terms of computational load and memory storage) for long experiments. Instead of performing the sum over t at every iteration, one can define alternative data arrays Q and Y as follows: $Q_{\tau i \tau' i'} = \sum_t M_{t\tau i} M_{t\tau' i'}$ and $Y_{\tau i} = \sum_t M_{t\tau i} r(t)$. Then $[\mathsf{B}^\top \mathsf{B}]_{\tau\tau'} = \sum_{ii'} Q_{\tau i \tau' i'} b_i b_{i'}$ and $[\mathsf{B}^\top \mathbf{r}]_\tau = \sum_i Y_{\tau i}\, b_i$ in the update for $\mathbf{w}$, i.e., the sum over t is now no longer required. The expressions for the $\mathbf{b}$ update are analogous: $[\mathsf{W}^\top \mathsf{W}]_{ii'} = \sum_{\tau\tau'} Q_{\tau i \tau' i'} w_\tau w_{\tau'}$ and $[\mathsf{W}^\top \mathbf{r}]_i = \sum_\tau Y_{\tau i}\, w_\tau$.

C. Regularisation of $\mathbf{b}$ *in the input nonlinearity model*

Note that the regularisation techniques explained below are applicable to all models that use basis functions. Penalizing the first derivative of $f(x) = \sum_i b_i f_i(x)$ can be written as placing a prior covariance on the vector $\mathbf{b}$, because adding a quadratic term $(1/2)\mathbf{b}^\top \mathsf{D}^b \mathbf{b}$ to the objective function $\mathcal{E}$ is equivalent to placing a Gaussian prior with inverse covariance D^b on $\mathbf{b}$. Thus, we compute D^b so as to penalise the first derivative of f:

$$\lambda \int \left(\frac{\mathrm{d}f(x)}{\mathrm{d}x} \right)^2 \mathrm{d}x = \lambda \int \left(\sum_i b_i \frac{\mathrm{d}f_i(x)}{\mathrm{d}x} \right)^2 \mathrm{d}x = \sum_{ij} b_i b_j D^b_{ij}. \tag{6}$$

This equation holds if we define D^b as

$$\mathsf{D}^b_{ij} = \lambda \int \frac{\mathrm{d}f_i(x)}{\mathrm{d}x} \frac{\mathrm{d}f_j(x)}{\mathrm{d}x} \mathrm{d}x,$$

Downloaded By: [Columbia University] At: 22:13 19 June 2008

where $f_i(\cdot)$ are basis functions. Note that this is an improper prior (does not integrate to 1), since D^b has a zero eigenvalue – but this penalization is nonetheless useful because it corresponds to familiar features of f (the steepness). Instead of the first derivative, the second derivative of f may be a more relevant property to control. The piecewise linear basis was used for all examples in this article, hence we derive an expression for D^b tailored to this basis set. Note that the second derivative of a piecewise linear function is ill-defined: here we define it (up to an arbitrary constant multiplier) to be the difference between the slopes at either side of the nodes. The piecewise linear basis consists of triangles, which start at 0 at a node x_{q-1}, rise linearly to 1 at the neighbouring node x_q, and descend linearly to 0 at the next node x_{q+1}; see Appendix A. Penalizing the curvature now involves only the nodes, because in between the nodes, f is linear and has zero curvature. Using the notation x_q^+ and x_q^- for values just above and below x_q (e.g., $x_q^- = x_q - \delta$, for δ very small), we penalize

$$\lambda \int \left(\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\right)^2 \mathrm{d}x = \lambda \sum_q \left(\frac{\mathrm{d}f}{\mathrm{d}x}\Big|_{x_q^-} - \frac{\mathrm{d}f}{\mathrm{d}x}\Big|_{x_q^+}\right)^2 = \sum_{ij} b_i b_j D^b_{ij}$$

where $D^b_{ij} = \lambda \int f_i''(x) f_j''(x) \mathrm{d}x$ is, for a piecewise linear basis set,

$$D^b_{ij} = \lambda \sum_q f_i'(x_q^-) f_j'(x_q^-) - f_i'(x_q^-) f_j'(x_q^+) - f_i'(x_q^+) f_j'(x_q^-) + f_i'(x_q^+) f_j'(x_q^+).$$

Note that the only nonzero terms in this sum are those for which $|q-i| \le 1$ and $|q-j| \le 1$ because piecewise linear basis functions are nonzero only across three nodes.

Other regularising priors, for example penalizing higher derivatives (or the Laplacian in higher dimensions; see also Poggio et al. 1985), can be derived in a similar way. Replace $\mathrm{d}/\mathrm{d}x$ in Equation 6 by a different linear operation and find the corresponding D^b.

D. Regularisation of C in the full-rank model

Again using the symbol x for the stimulus value, and writing the effective response as $a(\tau,x) = \sum_i C_{\tau i} f_i(x)$, the penalty term is defined to be $\sum_\tau \int_x \alpha(\mathrm{d}a/\mathrm{d}\tau)^2 + \beta(\mathrm{d}^2 a/\mathrm{d}x^2)^2$, penalizing high derivatives in the τ direction and high second derivatives in the x direction. To write this as a quadratic form in C, note that $\mathrm{d}a/\mathrm{d}\tau = \sum_i (C_{\tau i} - C_{\tau-1,i}) f_i(x)$ and $\mathrm{d}^2 a/\mathrm{d}x^2 = \sum_i w_{\tau i} (\mathrm{d}^2 f_i/\mathrm{d}x^2)$. The expression for the penalty term is then

$$\sum_\tau \int_x \alpha \left(\frac{\mathrm{d}a}{\mathrm{d}\tau}\right)^2 + \beta \left(\frac{\mathrm{d}^2 a}{\mathrm{d}x^2}\right)^2 = \sum_{\tau\tau' ij} C_{\tau i} C_{\tau' j} D^C_{\tau i \tau' i'}$$

where D^C is the inverse prior covariance of C,

$$\begin{aligned} D^C_{\tau i \tau' i'} &= \alpha D^1_{\tau i \tau' i'} + \beta D^2_{\tau i \tau' i'} \\ &= \alpha(2\delta_{\tau,\tau'} - \delta_{\tau+1,\tau'} - \delta_{\tau,\tau'+1}) \int_x f_i(x) f_{i'}(x) + \beta \delta_{\tau,\tau'} \int_x f_i''(x) f_{i'}''(x). \end{aligned}$$

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Finally, because we are using a small but discrete resolution δ_x for x, we replace the integrals by sums:

$$\int_x f_i(x) f_{i'}(x) = \delta_x \sum_n f_i(x_n) f_{i'}(x_n),$$
$$\int_x f_i''(x) f_{i'}''(x) = \frac{1}{\delta_x} \sum_q \Big[f_i'(x_q^+) - f_i'(x_q^-) \Big] \Big[f_{i'}'(x_q^+) - f_{i'}'(x_q^-) \Big].$$

Here x_n are the points in the discretised space of stimulus value (such that $x_{n+1} = x_n + \delta_x$) and x_q are the nodes of the piecewise linear basis functions. D^C is re-shaped into $D^C_{mm'}$ when using it in linear regression by vectorising (τ, i): the index m replaces (τ, i) and m' replaces (τ', i'). The resolution δ_x appears in the definition of the prior as $\alpha \cdot \delta_x$ and as β/δ_x; the prior should not depend on the resolution, but this can be countered by absorbing δ_x into α and β, i.e. ignoring δ_x and tuning α and β by cross validation or automatically as in Appendix F.

E. Gibbs sampling in the bilinear model

Gibbs sampling involves fixing $\mathbf{b}$ to $\mathbf{b}^1$, drawing $\mathbf{w}^1$ from a probability distribution dependent on this value, and then fixing $\mathbf{w}$ to $\mathbf{w}^1$, and drawing $\mathbf{b}^2$ from a distribution dependent on $\mathbf{w}^1$, and so on (MacKay 2004). In this way, $\{\mathbf{w}^n, \mathbf{b}^n\}$ will (as n grows) converge to a set of samples from $P(\mathbf{w}, \mathbf{b} | \mathbf{r}, \mathsf{M}, \mathsf{D}^w, \mathsf{D}^b, \hat{\sigma}^2)$. The conditional probability distributions of $\mathbf{w}$ and $\mathbf{b}$ are Gaussians, with means and covariances obtained from the conditionally-linear regression problems (cf. the linear regression example in "Error bars through Gibbs sampling"):

$$P(\mathbf{w} | \mathbf{b}, \mathbf{r}, \mathsf{M}, \hat{\sigma}^2) = \mathcal{N}\big[(\mathsf{B}^\mathsf{T}\mathsf{B} + \hat{\sigma}^2 \mathsf{D}^b)^{-1} \mathsf{B}^\mathsf{T} \mathbf{r}, \hat{\sigma}^2 (\mathsf{B}^\mathsf{T}\mathsf{B} + \hat{\sigma}^2 \mathsf{D}^b)^{-1} \big]$$

and

$$P(\mathbf{b} | \mathbf{w}, \mathbf{r}, \mathsf{M}, \hat{\sigma}^2) = \mathcal{N}\big[(\mathsf{W}^\mathsf{T}\mathsf{W} + \hat{\sigma}^2 \mathsf{D}^w)^{-1} \mathsf{W}^\mathsf{T} \mathbf{r}, \hat{\sigma}^2 (\mathsf{W}^\mathsf{T}\mathsf{W} + \hat{\sigma}^2 \mathsf{D}^w)^{-1} \big]$$

where $B_{t\tau} = \sum_i b_i M_{t\tau i}$ and $W_{ti} = \sum_\tau w_\tau M_{t\tau i}$; since sampling from a multivariate Gaussian distribution requires only the computation of a matrix square root, Gibbs sampling here is quite computationally efficient. The noise parameter $\hat{\sigma}^2$ is normally set to the squared error between the real and predicted firing rates, using the parameters $\mathbf{w}_{\mathrm{MAP}}$ and $\mathbf{b}_{\mathrm{MAP}}$ obtained by the ALS procedure (though we may easily sample from the posterior distribution of σ^2 as well). These MAP parameters also provide good starting points for Gibbs sampling.

F. Adaptive regularisation of bilinear models

In the main text, we described an alternating least squares procedure to estimate the parameter vectors of an input-nonlinearity model. The algorithm made it easy to incorporate prior covariance matrices for the parameter vectors. In many cases, however, it is not clear what a good prior should be – e.g., how much smoothing is

Downloaded By: [Columbia University] At: 22:13 19 June 2008

needed to get sensible results from data with variable noise? Ideally, one should incorporate some flexibility by making the prior matrices depend on one or more hyperparameters θ (e.g., λ, α and β in Appendices C and D). Whilst these parameters may be set by intuitive expectation, or by cross-validation, we have also developed an empirical Bayesian approach that employs a Variational Bayes EM algorithm (Dempster et al. 1977; Beal 2003) to fit this hierarchical model, treating λ, α and β as hyperparameters and approximately integrating over the parameters **b** and **w**. While the ALS procedure only keeps track of the mean of the parameter vectors, the Bayesian approach also keeps track of their covariances; while previously, a new estimate of one parameter was dependent only on the previous estimate of the other parameters, it now also depends on the uncertainty about the other parameters. Here we only present the resulting algorithm. The derivations and variations will be discussed elsewhere.

The algorithm contains several variables which are updated inside a loop. The terms $\mathbf{u}^{w,b}$ and $\Sigma^{w,b}$ represent estimates of the posterior means and covariance matrices of the parameter vectors **w** and **b**, respectively. $\mathsf{S}^w(\theta^w)$ and $\mathsf{S}^b(\theta^b)$ are the prior covariance matrices that are responsible for regularising the estimates, and depend on (possibly multidimensional) parameters $\theta^{w,b}$, which must be learnt. The parameter $\hat{\sigma}^2$ is an estimate of the scale of the Gaussian noise of the spike rate; this estimate is also updated at every iteration of the algorithm. Finally, the function $\mathcal{F}$ is the portion of the free energy, a lower bound on the log-likelihood, that depends on the θs.

The algorithm is initialised with guesses for $\mathbf{u}^w$ and $\mathbf{u}^b$ (e.g., $\mathbf{w}_{\mathrm{MAP}}$ and $\mathbf{b}_{\mathrm{MAP}}$); the posterior covariance matrices $\Sigma^{w,b}$ can be initialised as a multiple of the identity matrix. The variance $\hat{\sigma}^2$ can be initialized by e.g., the squared error coming from the ALS procedure, or by the variance of $r(t)$.

The algorithm is then:

1. Define
   - $\mathsf{C}^w \leftarrow \Sigma^w + \mathbf{u}^w[\mathbf{u}^w]^\top$
   - $Q^w_{ij} \leftarrow \sum_{tkl} C^w_{kl} M_{tik} M_{tjl}$
   - $[\mathbf{v}^w]_i \leftarrow \sum_{tk} [\mathbf{u}^b]_k \, M_{tik} r(t)$.
2. Update Σ^w and $\mathbf{u}^w$ according to

$$\Sigma^w \leftarrow \mathsf{S}^w \left[\frac{\mathsf{Q}^w \mathsf{S}^w}{\hat{\sigma}^2} + \mathsf{I} \right]^{-1}$$

$$\mathbf{u}^w \leftarrow \Sigma^w \frac{\boldsymbol{v}^w}{\hat{\sigma}^2}$$

3. Perform the analogous operations of steps 1 and 2 for Σ^b and $\mathbf{u}^b$.
4. Update $\hat{\sigma}^2$ according to

$$\hat{\sigma}^2 \leftarrow \frac{1}{T} \left(\mathbf{r}^\mathsf{T} \mathbf{r} - 2 \sum_{tik} [\mathbf{u}^w]_i [\mathbf{u}^b]_k r(t) M_{tik} + \sum_{tijkl} C^w_{ij} C^b_{kl} M_{tik} M_{tjl} \right)$$

where $T = \mathrm{length}(\mathbf{r})$, $\mathsf{C}^w = \mathbf{u}^w[\mathbf{u}^w]^\top + \Sigma^w$ and $\mathsf{C}^b = \mathbf{u}^b[\mathbf{u}^b]^\top + \Sigma^b$.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

5. Do gradient ascent on the function

$$\mathcal{F} = -\frac{1}{2}\log|\mathsf{S}^{w}| - \frac{1}{2}\log|\mathsf{S}^{b}| - \frac{1}{2}\mathrm{trace}\left(\mathsf{C}^{w}[\mathsf{S}^{w}]^{-1} + \mathsf{C}^{b}[\mathsf{S}^{b}]^{-1}\right)$$

with respect to θ^{w} and θ^{b} (which may be multidimensional). The gradients are (written both in terms of the prior covariance S^{w} and the inverse prior covariance $\mathsf{D}^{w} = [\mathsf{S}^{w}]^{-1}$, so that the algorithm can be implemented using either form of regularization),

$$\begin{aligned}\frac{\partial}{\partial\theta^{w}}\mathcal{F} &= \frac{1}{2}\mathrm{trace}\left[\left(\mathsf{C}^{w}[\mathsf{S}^{w}]^{-1} - \mathsf{I}\right)\frac{\partial\mathsf{S}^{w}}{\partial\theta^{w}}[\mathsf{S}^{w}]^{-1}\right] \\ &= \frac{1}{2}\mathrm{trace}\left[(\mathsf{I} - \mathsf{C}^{w}\mathsf{D}^{w})[\mathsf{D}^{w}]^{-1}\frac{\partial\mathsf{D}^{w}}{\partial\theta^{w}}\right]\end{aligned}$$

with an analogous expression for the gradient in the θ^{b} direction. One can either take one or a few gradient steps, or continue the gradient ascent until convergence.

6. Continue this loop, i.e. go to step 1, until $\mathbf{u}^{w}$, $\mathbf{u}^{b}$ and the Σ's converge.

Note that step 1 of this algorithm reduces to a step in the ALS procedure if the Σ's are set to zero.

The prior for the full-rank model can also be adaptively tuned, e.g., by using evidence optimization techniques described in MacKay (1994) and Sahani and Linden (2003a). The latter paper also presents further formulations for tunable prior covariance matrices $\mathsf{S}(\theta)$.

References

Ahrens MB, Linden JF, Sahani M. 2008. Nonlinearities and contextual influences in auditory cortical responses modeled with multilinear spectrotemporal methods. Journal of Neuroscience 28(8):1929–1942.

Arabzadeh E, Petersen RS, Diamond ME. 2003. Encoding of whisker vibration by rat barrel cortex neurons: implications for texture discrimination. Journal of Neuroscience 23:9146–9154.

Arabzadeh E, Zorzin E, Diamond ME. 2005. Neuronal encoding of texture in the whisker sensory pathway. PLOS Biology 3:155–165.

Bai EW. 1998. An optimal two-stage identification algorithm for Hammerstein-Wiener nonlinear systems. Automatica 34:333–338.

Beal MJ. 2003. Variational Algorithms for Approximate Bayesian Inference. PhD Thesis, Gatsby Computational Neuroscience Unit : University College London.

Berman M, Turner TR. 1992. Approximating Point Process Likelihoods with GLIM. Applied Statistics 41:31–38.

Breiman L, Friedman JH. 1985. Estimating optimal transformations for multiple regression and correlation. Journal of the American Statistical Association 80(391):580–598.

Brenner N, Bialek W, de Ruyter van Steveninck R. 2000. Adaptive rescaling maximizes information transmission. Neuron 26:695–702.

Bussgang JJ. 1952. Crosscorrelation functions of amplitude-distorted Gaussian signals : Technical Report No. 216. Research Laboratory of Electronics MIT.

Chichilnisky EJ. 2001. A simple white noise analysis of neuronal light responses. Network: Computation in Neural Systems 12:199–213.

Christianson GB, Sahani M, Linden JF. 2008. The Consequences of Response Nonlinearities for Interpretation of Spectrotemporal Receptive Fields. Journal of Neuroscience 28(2):446–455.

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Cronin B, Schummers J, Koerding K, Sur M. 2006. Bayesian sampling methods for the analysis of reverse correlation data. Society for Neuroscience abstract 545.3/T5, San Diego.

de Ruyter van Steveninck R, Bialek W. 1988. Real-time performance of a movement-sensitive neuron in the blowfly visual system: Coding and information transmission in short spike sequences. Proceedings of the Royal Society of London Series B 234:379–414.

DeAngelis GC, Ohzawa I, Freeman RD. 1995. Receptive-field dynamics in the central visual pathways. Trends in Neuroscience 18:451–458.

Dempster AP, Laird NM, Rubin DB. 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society B 39:1–38.

Depireux DA, Simon JZ, Klein DJ, Shamma SA. 2001. Spectro-temporal response field characterization with dynamic ripples in ferret primary auditory cortex. Journal of Neurophysiology 85:1220–1234.

Effron B, Tibshirani RJ. 1993. An introduction to the bootstrap. New York: Chapman & Hall.

Hastie T, Tibshirani R, Friedman J. 2001. The elements of statistical learning: data mining, inference and prediction. New York: Springer-Verlag.

Hastie TJ, Tibshirani RJ. 1999. Generalized additive models. Monographs on Statistics and Applied Probability. Vol. 43. New York: Chapman & Hall/CRC.

Hunter IW, Korenberg MJ. 1986. The Identification of Nonlinear Biological Systems: Wiener and Hammerstein Cascade Models. Biological Cybernetics 55:135–144.

Juusola M, Weckström M, Uusitalo RO, Korenberg MJ, French AS. 1995. Nonlinear models of the first synapse in the light-adapted fly retina. Journal of Neurophysiology 74:2538–2547.

Korenberg MJ. 1991. Recent advances in the identification of nonlinear systems: Minimum-variance approximation by Hammerstein models. Annals of the International Conference of IEEE Engineering in Medicine and Biology Society 13:2258–2259.

Linden JF, Liu RC, Sahani M, Schreiner CE, Merzenich MM. 2003. Spectrotemporal structure of receptive fields in areas AI and AAF of mouse auditory cortex. Journal of Neurophysiology 90:2660–2675.

Luczak A, Hackett TA, Kajikawa Y, Laubach M. 2004. Multivariate receptive field mapping in marmoset auditory cortex. Journal of Neuroscience Methods 136:77–85.

Machens CK, Wehr MS, Zador AM. 2004. Linearity of cortical receptive fields measured with natural sounds. Journal of Neuroscience 24:1089–1100.

MacKay DJC. 1994. Bayesian non-linear modelling for the prediction competition. ASHRAE Transactions V. 100 Pt.2 Atlanta Georgia: ASHRAE, pp. 1053–1062.

MacKay DJC. 2004. Information Theory, Inference, and Learning Algorithms : Cambridge University Press.

Marmarelis PZ, Naka KI. 1973. Nonlinear analysis and synthesis of receptive-field responses in the catfish retina. I. Horizontal cell leads to ganglion cell chain. Journal of Neurophysiology 36:605–618.

McCullagh P, Nelder J. 1989. Generalized linear models. New York: Chapman & Hall.

Narendra KS, Gallman PG. 1966. An Iterative Method for the Identification of Nonlinear Systems Using a Hammerstein Model. IEEE Transactions on Automatic Control AC-11:546–550.

Paninski L. 2003. Convergence properties of three spike-triggered analysis techniques. Network: Computation in Neural Systems 14:877–883.

Paninski L. 2004. Maximum likelihood estimation of cascade point-process neural encoding models. Network: Computation in Neural Systems 15:243–262.

Pillow JW, Paninski J, Uzzell VJ, Simoncelli EP, Chichilnisky EJ. 2005. Prediction and decoding of retinal ganglion cell responses with a probabilistic spiking model. Journal of Neuroscience 25:11003–11013.

Pillow JW, Simoncelli EP. 2006. Dimensionality reduction in neural models: an information-theoretic generalization of spike-triggered average and covariance analysis. Journal of Vision 6:414–428.

Pinto DJ, Brumberg JC, Simons DJ. 2000. Circuit dynamics and coding strategies in rodent somatosensory cortex. Journal of Neurophysiology 83:1158–1166.

Poggio T, Torre V, Koch C. 1985. Computational vision and regularization theory. Nature 317:314–319.

Rigat F, de Gunst M, van Pelt J. 2006. Bayesian modelling and analysis of spatio-temporal neuronal networks (in press).

Sahani M, Linden JF. 2003a. Evidence optimization techniques for estimating stimulus-response functions. In: Becker S, Thrun S, Obermayer K, editors. Advances in Neural Information Processing

Downloaded By: [Columbia University] At: 22:13 19 June 2008

Systems 15. Cambridge, MA: MIT Press. Vol. 15. pp 109–116. Available online via http://books.nips.cc.

Sahani M, Linden JF. 2003b. How linear are auditory cortical responses?. In: Becker S, Thrun S, Obermayer K, editors. Advances in Neural Information Processing Systems 15. Cambridge, MA: MIT Press. pp 109–116. Available online via http://books.nips.cc.

Schwartz O, Chichilnisky EJ, Simoncelli EP. 2002. Characterizing neural gain control using spike-triggered covariance. In: Dietterich TG, Becker S, Gharamani Z, editors. Advances Neural Information Processing Systems. Cambridge, MA: MIT Press Vol. 14. pp 269–276. Available online via http://books.nips.cc.

Sharpee T, Rust NC, Bialek W. 2004. Analyzing neural responses to natural signals: Maximally informative dimensions. Neural Computation 16:223–250.

Simoncelli EP, Pillow J, Paninski L, Schwartz O. 2004. Characterization of neural responses with stochastic stimuli. In: Gazzaniga M, editor. The Cognitive Neurosciences. 3rd ed.. Cambridge, MA: MIT Press. pp 327–338.

Spekreijse H, Oosting H. 1970. A method for analysing and synthesizing nonlinear systems. Kybernetik 7:23–31.

Strang G. 1988. Linear Algebra and its Applications. 3rd ed. Brooks Cole: Boston, MA.

Suits DB, Mason A, Chan L. 1978. Spline functions fitted by standard regression methods. The Review of Economics and Statistics 60:132–139.

Touryan J, Felsen G, Dan Y. 2005. Spatial structure of complex cell receptive fields measured with natural images. Neuron 45:781–791.

Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN. 2005. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. Journal of Neurophysiology 93:1074–1089.

Webber RM, Stanley GB. 2004. Nonlinear encoding of tactile patterns in the barrel cortex. Journal of Neurophysiology 91:2010–2022.

Westwick WT, Kearney RE. 2001. Separable least squares identification of nonlinear Hammerstein models: application to stretch reflex dynamics. Annals of Biomedical Engineering 29:707–718.

Young ED, Calhoun BM. 2005. Nonlinear modeling of auditory-nerve rate responses to wideband stimuli. Journal of Neurophysiology 94:4441–4454.

Young FW, de Leeuw J, Takane Y. 1976. Regression with qualitative and quantitative variables: An alternating least squares method with optimal scaling features. Psychometrika 41:505–529.