Inferring input nonlinearities in neural encoding models

Misha B. Ahrens[1], Liam Paninski[2] and Maneesh Sahani[1]

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

November 22, 2006

Draft - to be submitted

Abstract. We describe a class of models that can be used to predict how the instantaneous firing rate of a neuron varies in response to a dynamic stimulus. These models are based on learned pointwise nonlinear transforms of the stimulus, followed by a temporal linear filtering operation on the transformed inputs. In one case, the transformation is the same for all lag-times. Thus, this "input nonlinearity" converts the initial numerical representation of the stimulus (e.g. air pressure) to a new representation which is optimal as input to the subsequent linear model (e.g. decibels). We present algorithms for estimating both the input nonlinearity and the linear weights, including regularization techniques, and for quantifying the experimental uncertainty in these estimates. In another approach, the model is generalized to allow a potentially different nonlinear transform of the stimulus value at each lag-time. Although more general, this model is algorithmically more straightforward to fit. However, it contains many more degrees of freedom, and thus requires considerably more data for accurate and precise estimation. The feasibility of these new methods is demonstrated both on synthetic data, and on responses recorded 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.

1 Introduction

Neural encoding models predict how the instantaneous firing rate of a neuron varies in response to a dynamic input, such as the speed of a jittering bar in the visual field, a time-varying sound, the spike rates of a collection of upstream neurons, or a combination of such a stimulus and the spike history of the neuron itself. One major reason for interest in such models is that, once fit to neural data, their parameters can be used to investigate the encoding properties of the modelled neuron; which 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 & Linden, 2003b; Machens et al., 2004; Petersen & Montemurro, 2006). 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. This suggests that nonlinear encoding models may be needed to provide an accurate description of the neuron's functional response (e.g. Marmarelis & Naka, 1973; Fishbach et al., 2001; de Ruyter van Steveninck & Bialek, 1988).

Considerable effort has recently been directed toward LNP (Linear-Nonlinear-Poisson) models, where a linear temporal filter acting on a time-varying stimulus signal is followed by an output nonlinearity to predict the firing rate (Brenner et al., 2000; Schwarz et al., 2002; Paninski, 2003, 2004; Simoncelli et al., 2004; Sharpee et al., 2004; Pillow & Simoncelli, 2006). One motivation for such models is to capture the nonlinearity of 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. Suppose, for instance, 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 were roughly equal, the neuron would effectively respond to the absolute value of the sensory inputs. In this case, a linear model fit to a stimulus that contained equal deflections in both directions, could do no better than predict a constant firing rate.

[Figure 1 about here.]

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 & Friedman, 1985; Hastie & 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 in sections 2.1 and 2.2. 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 & Gallman, 1966; Hunter & Korenberg, 1986; Juusola et al., 1995; Bai, 1998; Westwick & Kearney, 2001). Here, we give these models a probabilistic basis, allowing for the construction of principled regularization techniques (section 2.5); estimation of error bars (section 2.6); we draw connections between the full-rank model and the bilinear model (section 2.3); and we extend the formulation of the bilinear model to the framework of predicting point-process spike trains (section 2.9), leading to the Generalized Bilinear Model or the "NLNP model". We also evaluate the models on simulated and real neural data in sections 3 and 4.

2 The models

2.1 The bilinear model

The model. Consider a one dimensional time-varying stimulus $s(t)$. 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 section 2.9 we will consider the case in which $r(t)$ is a 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 more a suitable distance metric. Generalization to higher dimensional stimuli is straightforward: in this case the index τ ranges over time and space, for example, instead of just time.

In this section we introduce an unknown pointwise transformation $f(.)$ of $s(t)$ into the model, and provide an algorithm to estimate the corresponding parameters. This transformation will be referred to as the "input nonlinearity". This 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 is desirable to be estimated from the data. Once f has been estimated, the resulting model is almost as conceptually 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 illustrates the predictive sequences of the model.

[Figure 2 about here.]

Estimation procedure. It is very hard to manually explore the high dimensional space of functions, especially with the possibility that f might be different for each neuron in a population. For fixed f, the bilinear model reduces to a linear model, and $\mathbf{w}$ can be estimated by linear regression. Learning f from the data, instead of assuming it, can be done as follows. First we choose a fixed set of basis functions $\{f_i\}$ and then express f as a linear combination of these:

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

Naturally the basis functions have to be chosen to approximately encompass the possible space of input nonlinearities (in practise, this might involve some manual exploration). In this paper we use a piecewise linear basis for f; 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 write M as a family of matrices indexed by t: $[\mathrm{M}(t)]_{\tau i} = M_{t\tau i}$. Then redefine

$$\mathrm{M}(t) \to \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{M}(t) \end{pmatrix} \tag{3}$$

and $\mathbf{w} \to [w_1 \quad \mathbf{w}]$, $\mathbf{b} \to [b_1 \quad \mathbf{b}]$, so that now $c = w_1 \cdot b_1$ and

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

Here, $\mathbf{w}$ and $\mathbf{b}$ are the parameter vectors, $\mathbf{w}$ describing the response to time, and $\mathbf{b}$ describing the input nonlinearity. M is the data array and is fixed.

To estimate $\mathbf{w}$ and $\mathbf{b}$, we find values for them that minimize the squared distance between the observed and the predicted spike rates, $\mathcal{E} = \sum_t (r(t) - \hat{r}(t))^2 = ||\mathbf{r} - \hat{\mathbf{r}}||^2$. (Alternatively, one can maximize the point-process likelihood; this will be described in section 2.9.) One method for doing this comes from rewriting the model as

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

or $\mathbf{r} = \mathrm{B}\mathbf{w}$, where the matrix $B_{t\tau} = \sum_i b_i M_{t\tau i}$. That is, if $\mathbf{b}$ is kept fixed, it can be combined with the other fixed components of the model, namely the data array M, to produce a pseudo-data matrix B. Now the model is linear in $\mathbf{w}$ and can be easily inverted to obtain the estimate $\mathbf{w} = (\mathrm{B}^\mathrm{T}\mathrm{B})^{-1}\mathrm{B}^\mathrm{T}\mathbf{r}$. This is the unique best estimate of $\mathbf{w}$ given the fixed $\mathbf{b}$ under the square loss $\mathcal{E}$, assuming $\mathrm{B}^\mathrm{T}\mathrm{B}$ is of full rank (if $\mathrm{B}^\mathrm{T}\mathrm{B}$ is of reduced rank, the inverse in the definition of $\mathbf{w}$ is interpreted as a pseudoinverse, and so the linear estimate may be considered unique in general).

On the other hand, if $\mathbf{w}$ is fixed, the model is

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

where $W_{ti} = \sum_\tau w_\tau M_{t\tau i}$. Now the conditional estimate for $\mathbf{b}$ is $(\mathrm{W}^\mathrm{T}\mathrm{W})^{-1}\mathrm{W}^\mathrm{T}\mathbf{r}$.

If $\mathbf{w}$ and $\mathbf{b}$ are alternately updated in this way, at each step holding the other vector fixed at its most recent value, $\mathcal{E}$ decreases or stays constant at each step. This estimation procedure has to be initialized with a guess for the parameter vector that is held fixed during the first step. The iterations should be continued until $\mathcal{E}$ it converges. At that point, $\mathbf{w}$ and $\mathbf{b}$ are noted, and their first elements removed and formed into the baseline firing rate $c = w_1 \cdot b_1$. The model now consists of a temporal filter $\mathbf{w}$, an input nonlinearity $f(x) = \sum_i b_i f_i(x)$ and a baseline firing rate c. This procedure has the structure of alternating least squares (ALS) (e.g. Young et al., 1976). In certain cases, when the time-range of the stimulus is large compared to the size of $\mathbf{w}$ and $\mathbf{b}$, there is an economical and fast alternative to this algorithm (see Appendix B).

Note that, once the input nonlinearity is determined, it may be fixed and used in extensions of the linear model such as the generalized linear model (which includes an output nonlinearity) and 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 can also be fixed (or approximated by a simpler function) so that estimation on the rest of the population can be carried out as in a linear model. It is also possible to estimate spike history terms simultaneously with the input nonlinearity and the temporal filter (see section 2.9).

As discussed above, a linear model is guaranteed to have a unique optimum, but unfortunately we are not able to give a similar uniqueness guarantee for the bilinear model. For each one of the two parameter vectors $\mathbf{w}$ and $\mathbf{b}$ there is a unique optimum *conditioned* on

the other, but since the other vector also changes in the optimization, there is no guaranteed joint uniqueness. Thus one can only be certain of reaching a local minimum of $\mathcal{E}$. Using numerical simulations and random data arrays M and $\mathbf{r}$, we found a few such local minima. In practise, however, these have not caused problems; at most two different local minima were ever observed in the neuronal data discussed in section 4, and they were almost identical in shape. But in larger models initialization of the parameters might be important (e.g. see section 2.3).

2.2 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 a visual spatiotemporal receptive field, for example: this can be written as a matrix $W_{\tau x}$, and a linear spike rate prediction model would be $\hat{r}(t) = \sum_{\tau=0}^{\tau_{\max}} \sum_x W_{\tau x} S(t-\tau, x)$, with S a time-varying movie presented to the retina. In some cases, it is reasonable to approximate the full matrix $W_{\tau x}$ (which may in general be of full rank) with a simpler, rank-1 matrix, $W_{\tau x} = u_\tau v_x$. In this case we say that the receptive field is "separable," because the receptive field may be written as a product of two separate functions of time and space, $\mathbf{u}$ and $\mathbf{v}$, respectively. In the context of visual spatiotemporal fields, this separability concept is beneficial, since we need only estimate $\dim(\mathbf{u}) + \dim(\mathbf{v})$ parameters, instead of the (typically much-larger) $\dim(\mathbf{u}) \times \dim(\mathbf{v})$, in the case of a full-rank matrix W. On the other hand, the class of separable receptive fields is strictly less flexible than the general class of full-rank (inseparable) receptive fields; for example, separable receptive fields are unable to model direction selectivity.

In the present case of the bilinear model, the receptive field $w_\tau b_i$ can be thought of as a separable (rank-1) receptive field in time (τ) and in stimulus value (i), and the data tensor $M_{t\tau i}$ as a two dimensional time varying "stimulus," varying in the τ and i dimensions. This leads to a generalization of the bilinear model in which $w_\tau b_i$, a rank-1 matrix, is replaced by a general matrix $C_{\tau i}$ to form the *full-rank model*. Being a generalization of the bilinear model, it has the potential of capturing more intricate structure of the stimulus response function, but it has more parameters ($\dim(\mathbf{w}) \times \dim(\mathbf{b})$ in the full-rank case instead of $\dim(\mathbf{w}) + \dim(\mathbf{b})$ in the rank-1 case), so that the data requirements increase when the stimulus is high dimensional, to avoid overfitting (explaining noise on the training data and performing poorly on cross-validation data; in such cases the significance of the model parameters is unclear).

The full-rank model uses the same data array M as the bilinear model:

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

One can estimate this model in a single step, by standard linear regression methods to fit the weights $C_{\tau i}$ and the offset c.

The full-rank model has a unique global optimum (in the sense discussed above) because it is linear in the matrix M. In particular, both the objective function $\mathcal{E}$ 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 a quadratic function, and the parameter space is convex because adding two general matrices together will produce another matrix. Therefore fitting the full-rank model is analogous to descending a parabola on the real line: we are guaranteed to reach the optimum.

Now, returning to the bilinear model: this model has the same objective function as the full-rank model, but a different parameter space: we have to restrict our search to the

space of all rank-1 matrices $w_\tau b_i$. This parameter space is not convex because, by definition, the sum of two rank-1 matrices is a rank-2 matrix, exempting special cases. Therefore, the iterative algorithm for the input nonlinearity model is not guaranteed to converge to the global optimum, even though each step of the optimization (fitting $\mathbf{w}$ with $\mathbf{b}$ fixed, and vice versa) does have a unique optimum. However, as we discussed above, in our experience to date this lack of a global optimizer guarantee has had minimal impact on the performance of the model (likely because each step of the algorithm has a guaranteed optimal solution).

2.3 Rank-k models

Since the bilinear model is a rank-1 special case of the full-rank model, one can also consider rank-k approximations to the full-rank model. These may be able to capture more detailed response properties of the neuron, but at the expense of a potential increase in overfitting, compared to the rank-1 model. A common method for finding low rank approximations to a matrix is the singular value decomposition (SVD, Strang, 1988); this is frequently applied to linear auditory receptive fields (Depireux et al., 2001; Linden et al., 2003). Taking the single leading SVD term produces an approximation to the full-rank model, $C_{\tau i} \approx w_\tau b_i$, giving a bilinear input nonlinearity model (Bai, 1998). However, it can be shown that the $(\mathbf{w}, \mathbf{b})$ thus found minimize the squared Euclidean distance between C and $\mathbf{w}\mathbf{b}^{\mathrm{T}}$, rather than the squared distance between the real and the predicted firing rates. Thus, while the SVD method may provide a good initial guess for $\mathbf{w}\mathbf{b}^{\mathrm{T}}$, we need to apply the alternating least squares procedure of section 2.1 to optimize our loss function $\mathcal{E}$.

Approximating the full-rank model by the k leading SVD components of a full-rank model, $C_{\tau i} = w_\tau^1 b_i^1 + w_\tau^2 b_i^2 + ... + w_\tau^k b_i^k$, is equivalent to taking a sum of k bilinear models, so that the firing rate is predicted by $\hat{r}(t) = c + \sum_k \sum_\tau w_\tau^k f^k(s(t-\tau))$. The extra bilinear models may describe finer aspects of the stimulus response functions. Again, the parameters $\mathbf{w}^{1..k}$ and $\mathbf{b}^{1..k}$, obtained by SVD, minimize an inappropriate objective function (the Euclidean distance to the full-rank matrix C). However, the ALS estimation can still be applied to minimize the squared error $||\mathbf{r} - \hat{\mathbf{r}}||^2$, if one redefines the data array M. E.g. if $k = 3$,

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

with "0" indicating blocks of 0's, the model $\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i}$ is exactly a sum of three bilinear models, with $\mathbf{w}^{1,2,3}$ and $\mathbf{b}^{1,2,3}$ appearing as concatenated vectors in its parameters. More specifically, after learning the parameters with the ALS procedure, $\mathbf{w}$ and $\mathbf{b}$ are decomposed into $\mathbf{w} = [w_1 \quad \mathbf{w}^1 \quad \mathbf{w}^2 \quad \mathbf{w}^3]$, $\mathbf{b} = [b_1 \quad \mathbf{b}^1 \quad \mathbf{b}^2 \quad \mathbf{b}^3]$ with $c = w_1 \cdot b_1$, and the three bilinear models are recovered. The ALS procedure can, of course, be initialized with the vectors coming from the SVD approximation to the full-rank model.

2.4 Models with multiple stimulus features

The framework discussed above allows for arbitrary transformations of the stimulus value, once a stimulus feature has been decided upon. For example, in the context of analyses of whisker barrel data, one might decide a priori that the velocity is the key stimulus feature of interest (Pinto et al., 2000), and define $s(t)$ to be the velocity at time t — and now by

fitting the models described above one can infer what function of the velocity, $f(s)$, is most predictive of the firing rate. But how do we know a priori that the velocity is the "correct" feature to analyze (and nonlinearly transform)? Perhaps a better model could be obtained by nonlinearly transforming the position of the whisker, or the acceleration. Different choices of the stimulus feature (i.e., velocity vs. acceleration) may lead to a different performance of the model.[1] Completely different stimuli can also be combined in one model, e.g. the light intensity and sound volume of an audiovisual stimulus. If we seek a single most appropriate stimulus, we can of course sequentially train one model per stimulus and then check which is most predictive. But the stimuli may interact or jointly influence the firing rate, in which case it is 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); here we show how such formulations extend to the data tensor M of 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 M 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. We define a new stimulus tensor containing the three old ones:

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

where e.g. $[\mathrm{M}^1(t)]_{\tau i} = M^1_{t\tau i}$. If we also concatenate the $\mathbf{w}^{1,2,3}$ and the $\mathbf{b}^{1,2,3}$ vectors to form $\mathbf{w} = [w_1 \quad \mathbf{w}^1 \quad \mathbf{w}^2 \quad \mathbf{w}^3]$ and $\mathbf{b} = [b_1 \quad \mathbf{b}^1 \quad \mathbf{b}^2 \quad \mathbf{b}^3]$, then the model takes on the shape of the simple bilinear model: $\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i}$. $\mathbf{b}$ and $\mathbf{w}$ are estimated as before, using the ALS procedure. As before, 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 this uses the same stimulus array. The model is

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

[1] This is not true in linear models if the stimulus features are linear transformations of one another. Here, however, that property disappears due to the input nonlinearity.

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:

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

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.

2.5 Regularization

To cope with limited or noisy data, we now equip the bilinear model with regularization methods, to reduce the chances of overfitting (i.e. using the model parameters to explain noise in the training data). By minimizing the squared error $||\mathbf{r} - \hat{\mathbf{r}}||^2$, 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 now allows for principled regularization techniques and estimates for error bars. If we observe $\mathbf{r}$, then the noise term induces a probability distribution over the parameters of the model; we can now specify prior probability distributions on those parameters that describe our expectations, such as the degree of smoothness of $\mathbf{w}$. Imposing priors is thus a method for regularizing the parameters of the model. Gaussian priors are very convenient choices; e.g. $P^{\mathrm{prior}}(\mathbf{w}) \sim \mathcal{N}(0, \mathrm{S}^w)$ with S^w the prior covariance matrix describing the expected smoothness, size, etc. of $\mathbf{w}$. We will often work with the *inverse* covariance D so that $\mathrm{S}^{\mathrm{prior}} = \mathrm{D}^{-1}$.

Bilinear model. As described in section 2.1, 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 & 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 $\log(P(\mathbf{w}, \mathbf{b}|\mathrm{M}, \mathbf{r})) = -\frac{1}{2\sigma^2}\hat{\mathbf{r}}^{\mathrm{T}}\hat{\mathbf{r}} + \frac{1}{\sigma^2}\mathbf{r}^{\mathrm{T}}\hat{\mathbf{r}} - \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathrm{D}^w\mathbf{w} - \frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathrm{D}^b\mathbf{b} + \mathrm{const}$, where the constant does not depend on $\mathbf{w}$ or $\mathbf{b}$ and $\hat{\mathbf{r}}$ is given in terms of $\mathbf{w}$ and $\mathbf{b}$; we may maximize this expression analytically with respect to $\mathbf{w}$ to obtain the usual regularized least squares solution:

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

where $\hat{\sigma}^2$ is an estimate of the noise scale σ^2. For example, in ridge regression, the matrix D^w is a multiple of the identity, so that the values of $\mathbf{w}$ are encouraged to be small. Another common choice for D^w is a matrix with 2's on the diagonal, -1's on the neighbouring positions,

all scaled by a parameter λ^w which sets the "strength" of the regularization (equivalently, $1/\lambda^w$ sets the reliability of the data). Such a D^w will penalize high derivatives of $\mathbf{w}$; this may be seen easily if we notice that the corresponding Gaussian log-prior on $\mathbf{w}$ is proportional to $-\mathbf{w}^{\mathrm{T}}\mathrm{D}^w\mathbf{w} = -\sum_i[\mathbf{w}(i+1)-\mathbf{w}(i)]^2$. The regularized final estimate of $\mathbf{w}$ is now the Maximum A Posteriori (MAP) estimate, $\mathbf{w}_{\mathrm{MAP}}$.

The case of $\mathbf{b}$ is more complicated, because properties of the input nonlinearity $f(x) = \sum_i b_i f_i(x)$ have to be controlled instead of properties of $\mathbf{b}$: smoothness in $\mathbf{b}$ 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 as a quadratic form, so that e.g. to penalize the first derivative of f we find D^b so that $\mathbf{b}^{\mathrm{T}}\mathrm{D}^b\mathbf{b} = \lambda^b \cdot \int \left(\frac{\partial f(x)}{\partial x}\right)^2 dx$. (This is the continuous analogue of D^w defined above.) Again there is a multiplier λ^b which sets the strength of the regularization. Once the prior has been set, the update for $\mathbf{b}$ becomes $\mathbf{b} = (\mathrm{W}^{\mathrm{T}}\mathrm{W} + \hat{\sigma}^2\mathrm{D}^b)^{-1}\mathrm{W}^{\mathrm{T}}\mathbf{r}$, with $W_{ti} = \sum_\tau w_\tau M_{t\tau i}$. The regularized final estimate is now $\mathbf{b}_{\mathrm{MAP}}$.

So far, we have described priors depending on simple scaling parameters λ^w and λ^b which determine the strength of regularization. There may also be other parameters, which e.g. set the scale of smoothness. That is, the priors are parametrised by one or more parameters $\theta_{w,b}$, which are typically set by hand through cross-validation or by utilizing prior knowledge of the expected scale and smoothness of $\mathbf{w}$ and $\mathbf{b}$; good values for the θ's 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 regularization method described in Appendix F. This method is similar to Evidence Optimization techniques for linear models (Sahani & Linden, 2003a). After implementation, these automatic techniques can produce good results with no time spent on manual intervention. We plan to make these algorithms available on our website.

Full-rank model. Regularizing 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. Here 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 (since the input nonlinearity should not favour flat functions). Again, the regularization is done through a Gaussian prior on the model parameters C, specified as a regularization 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 Optimization techniques can be readily applied here (e.g. using an adaptive prior $\lambda\mathrm{D}^{\mathrm{C}}$ and learning an optimal value for λ, or by using other forms of prior covariance matrices; see Sahani & Linden, 2003a).

2.6 Error bars through Gibbs sampling

Error bars for 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 $\mathbf{y} = \mathrm{X}\mathbf{v} + \sigma\eta(t)$, with $\eta(t)$ 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 $\hat{\sigma}^2(\mathrm{X}^{\mathrm{T}}\mathrm{X} + \hat{\sigma}^2\mathrm{D})^{-1}$, because the diagonal elements of a covariance matrix are the marginal variances of the parameters. In the bilinear case, there is no analytical estimate for the error bars due to the dependencies between $\mathbf{w}$ and $\mathbf{b}$. The error bars are determined by

the spread of the posterior distribution of the parameters, $P(\mathbf{w}, \mathbf{b}|\mathbf{r}, \mathrm{M}, \mathrm{D}^w, \mathrm{D}^b, \hat{\sigma}^2)$, around the estimated parameters, $\mathbf{w}_{\mathrm{MAP}}$ and $\mathbf{b}_{\mathrm{MAP}}$. Fortunately, it is quite easy to sample from this distribution using Gibbs sampling (e.g. MacKay, 2004). This sampling procedure, 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 — by simply 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 $f(.) = \mathrm{F}\mathbf{b} = \sum_i b_i f_i(.)$, where F is the matrix containing the basis elements $f_i(.)$: if Σ 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 usual formula $(\mathrm{F}\Sigma\mathrm{F}^{\mathrm{T}})^{1/2}_{x,x}$.

The error bars for the parameters of the full-rank model can be estimated in a similar 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 $\Sigma = \hat{\sigma}^2(\mathrm{M}^{\mathrm{T}}\mathrm{M} + \hat{\sigma}^2\mathrm{D}^{\mathrm{C}})^{-1}$ (see section 2.5; M is here the data array and D^{C} the regularization array, both with with (τ, i) vectorized so that they become matrices). Defining an appropriate basis matrix F similar to the above, the error bars are now $(\mathrm{F}\Sigma\mathrm{F}^{\mathrm{T}})^{1/2}_{x\tau,x\tau}$.

2.7 Degeneracies

Bilinear model. A model contains degeneracies if different settings of its parameters produce identical input-output relationships. Such a model may also be called "non-identifiable": we cannot uniquely identify the parameters from the observed data. The bilinear model contains such degeneracies: the model does not change if $\mathbf{w} \to \lambda \cdot \mathbf{w}$ and $\mathbf{b} \to \frac{1}{\lambda} \cdot \mathbf{b}$, because the parameters only appear as the product $\mathbf{w}\mathbf{b}^{\mathrm{T}}$. This nonidentifiability leads to overestimation of the error bars. Error bars are estimated according to the principle "How much can this parameter be moved without changing the predicted firing rate too much?"; if any change in a parameter can be countered by changes in other parameters so that $\hat{\mathbf{r}}$ is unchanged, the error bars may potentially be infinitely wide (if there is no prior information to constrain the parameter estimates). Probabilistically, degeneracies induce flat directions in the likelihood. The Gibbs sampling procedure samples from these directions and overestimate the error bars. Figure 3 clarifies this picture by showing four equivalent configurations of the bilinear model.

[Figure 3 about here.]

The bilinear model has a second degerenacy: there is an equivalence between $\mathbf{b}$ and the constant c:

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

Now the last term is a 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, $\sum_i f_i(x) = 1$.[2] Thus, if $\mathbf{b}$ gets changed by adding a constant d to every entry of the vector, this change can be countered by subtracting $d \cdot \sum_\tau w_\tau$ from c. The equivalence of the models thus obtained can lead to seemingly

[2]Most sensible bases have the property of summing to 1, e.g. the piecewise linear, discrete, and spline bases. If they do not sum to 1, but another setting of $\mathbf{b}$ causes $f(x) = 1$, then the degeneracy will be more complicated than just a vertical shift.

large errorbars on $\mathbf{b}$ and c. This additive degeneracy also appears in figure 3, as a vertical shift in $\mathbf{b}$ and as a shift in c.

To remove the additive degeneracy and restore the identifiability of the model, one can change the basis set $\{f_i(.)\}$ by removing one function f_j before defining the data array M: this forces $f(.) = \sum_{i \neq j} b_i f_i(.)$ to be zero at a specific point (specifically, it is zero at the j^{th} node, $f(x_j) = 0$; see Appendix A); equation 5 does not hold anymore because now $\sum_{i \neq j} f_i(.) \neq 1$. Thus the additive degeneracy disappears and all the Gibbs samples will lie at a fixed height. (Note that one could also remove the additive degeneracy in an easier way, by removing the constant offset c from the model, i.e. by not performing the redefinition of M in equation 3 of section 2.1, but although this works for the bilinear model, this does not remove the degeneracy of the full-rank model; see below.) The multiplicative degeneracy can be fixed post-hoc, by rescaling the Gibbs samples to minimise the squared distance between the samples and $\mathbf{b}_{\text{MAP}}$. The inverse rescaling is applied to the corresponding $\mathbf{w}$ samples.

Full-rank model. There is also a degeneracy in the full-rank model: changing $C_{\tau i}$ to $C_{\tau i} + d_\tau$, with $\mathbf{d}$ an arbitrary vector that sums to zero, gives $\hat{r}(t) \rightarrow \hat{r}(t) + \sum_\tau d_\tau \sum_i (f_i(s(t - \tau))) = \hat{r}(t) + \sum_\tau d_\tau \cdot 1 = \hat{r}(t)$. That is, there is no change in the input-output relation of the model, but the receptive field does change: $a(\tau, x) = \sum_i C_{\tau i} f_i(x) \rightarrow a(\tau, x) + d_\tau$. This again leads to overestimation of the error bars. 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 the degeneracy disappears due to $\sum_{i \neq j} f_i(.) \neq 1$, and the estimated errorbars will have the right size.

2.8 Output nonlinearity

It will rarely be the case that the input nonlinearity framework can capture the entire stimulus response function of a neuron. In some cases it may be useful to slightly extend the model so that it also includes an output nonlinearity, which can, for example, prevent the predicted firing rate from becoming negative. The basis function framework can also be used to estimate such an output nonlinearity. That is, once a bilinear or full-rank model is fit to give an estimated firing rate $\hat{r}(t)$, one can apply a pointwise function $g(.)$ to it so that $g(\hat{r}(t))$ is a better estimate of $r(t)$ than $\hat{r}(t)$ alone. For example, g might cut off $\hat{r}(t)$ when it drops below zero. One way of finding g, described by Chichilnisky (2001), is to to find the average number of spikes that are elicited when $\hat{r}$ is in a certain interval; $g(x)$ is now defined by that average, when x is in the interval. This procedure is equivalent to fitting a nonlinear function g, expressed in a discrete basis $g(.) = \sum_j d_j g_j(.)$, to the graph of $(\hat{r}(t), r(t))$. $\mathbf{d}$ is fit by linear regression, just like $\mathbf{b}$ in the bilinear model. Instead of a discrete set of basis functions, one can choose any set; in section 3 we used piecewise linear basis functions with the same type of regularizing prior as 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 input nonlinearity or full-rank models.

2.9 Fitting spike trains: Generalized Bilinear or LNLP Models

So far, we have dealt only with spike rates, i.e. the average time-binned spike counts over multiple repetitions of the same simulus. The squared error $||\mathbf{r} - \hat{\mathbf{r}}||^2$ is an appropriate objective function for spike rates. The squared error is, however, not an appropriate objective function when fitting the models to spike trains, i.e. when $r(t)$ is 1 or 0 at any one time. An appropriate

objective function is the point-process likelihood (i.e., the probability of the measured spike train under the model given by $\mathbf{w}$ and $\mathbf{b}$) — or equivalently, the log-likelihood. Taking $\hat{r}(t) =$ probability of spike in time bin t (so that $\hat{r}(t) = dt \cdot$ (predicted number of spikes per second), with dt the size of a time bin), and $\{t_k\}$ the real spike times (so $r(t_k) = 1$ and $r(t) = 0$ if t is not a spike time), the point-process log-likelihood can be approximated by its limit as the bin size goes to zero (Berman & Turner, 1992):

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

Here $\hat{r}(t)$ is a model of 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 — here g is fixed and not estimated as in the previous section), and the constant offset c is incorporated into M as usual. The argument of g is exactly what we previously called the bilinear or the full-rank model. Commonly, a linear model with output nonlinearity is called a Generalized Linear Model (GLM; McCullagh & Nelder, 1989) or Linear-Nonlinear-Poisson model (LNP; Simoncelli et al., 2004); thus the bilinear version is a Generalized Bilinear Model or NLNP model (the first N standing for nonlinear) as depicted in figure 4, and the full-rank version is still a GLM.

[Figure 4 about here.]

We seek to estimate the same parameters as before, i.e. $\mathbf{w}$ and $\mathbf{b}$, or C, by maximizing $\mathcal{L}$. For the full-rank model, the methods are the same as for linear models, and a unique optimum of $\mathcal{L}$ is guaranteed if g is convex and log-concave (Paninski, 2003, 2004).

There is no analytical solution that maximizes $\mathcal{L}$, but one can use gradient ascent on $\mathcal{L}$ with respect to the model parameters. 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 in the $\mathbf{b}$ direction is

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

with a similar expression for $\partial \mathrm{L}/\partial w_\tau$. Note that the gradient simplifies when g is the exponential function due to $g'(y)/g(y) = 1$. One can still use a Gaussian prior for $\mathbf{b}$ and $\mathbf{w}$, in which case the objective function becomes the log-posterior $\mathcal{L} - \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathrm{D}^w\mathbf{w} - \frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathrm{D}^b\mathbf{b}$, whose gradient with respect to $\mathbf{b}$ is $\partial\mathcal{L}/\partial b_j - \left[\mathrm{D}^b\mathbf{b}\right]_j$ (and a similar expression for $\partial\mathcal{L}/\partial\mathbf{w}$), where D^w and D^b are regularization matrices as described in section 2.5.

Note that if g is convex and log-concave (e.g. $g(y) = \exp(y)$, $g(y) = \log(1 + \exp(y))$, the optimization is separately concave in the parameter vectors just like the Gaussian likelihood — i.e. if $\mathbf{b}$ is fixed, then the optimization is concave in $\mathbf{w}$ and thus has a unique optimum (Paninski, 2004), implying that we may easily fit the model by alternating maximization, just as in the setting of squared-error discussed above — 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 is superior to doing gradient descent on $\mathcal{L}$ with respect to $\mathbf{w}$ and $\mathbf{b}$ jointly. Instead of gradient methods, (alternating) Iteratively Reweighted Least Squares techniques are also available for the regularized estimation of the parameters.

One of the main reasons for using spike timing data rather than spike rate data is that the spike history of the neuron can be incorporated in the model in a natural way (Paninski,

2004). Spike history effects, such as refractory periods and self-excitation, are often modeled as an additive feed-back current: if the model neuron emits a spike, the current $\mathbf{z}$, varying over a small interval of J time bins, is fed back into the neuron. That is, j time steps after the last spike, the probability of spiking is changed by an amount z_j. Thus the model is $\hat{r}(t) = \sum_{\tau i} w_\tau b_i M_{t\tau i} + \sum_{j=1}^{J} z_j \mathrm{sp}(t-j)$, where $\mathrm{sp}(t)$ is either the true spike train $r(t)$ (when doing inference), or the spike ($\mathrm{sp} = 1$) or no-spike ($\mathrm{sp} = 0$) instantiations of the estimated firing rate $\hat{r}(t)$ (when generating sample spike trains). To estimate the current shape $\mathbf{z}$ together with $\mathbf{w}$ and $\mathbf{b}$ from the data, one incorporates the true spike train into the stimulus array M as follows. If the data array M is initially of size $T \times A \times B$, then define $M_{t,A+j,B+1} = r(t-j)$ for $j = 1...J$. The estimation procedure is now the same as before (see section 2.1), but at the end, one defines $z_j = w_{A+j} \cdot b_{B+1}$. That is, the spike history has become part of the data array and is treated as a "stimulus" which is not mapped through an input nonlinearity, unlike the true stimulus $s(t)$ (cf. section 2.4), so that the spike current $\mathbf{z}$ can be estimated together with the other model parameters — $\mathbf{w}$ in this case. In particular, the log-likelihood is concave in $(\mathbf{w}, \mathbf{z})$ and $(\mathbf{b}, \mathbf{z})$, so we may easily estimate $\mathbf{w}$ and $\mathbf{z}$ together, given $\mathbf{b}$, or conversely $\mathbf{b}$ and $\mathbf{z}$, given $\mathbf{w}$.

3 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 $P(t)$ according to three different processes, described below. Five spike trains, corresponding to five repeated "trials" of the experiment, were generated as a Poisson process with rate $P(t)$ (after some uniform noise was added to $P(t)$ before generating each spike train, to simulate unknown non-stimulus-locked "internal processes"). The observed firing rate $r(t)$ was taken to be the mean of these five spike trains. The first half of the spike rate and stimulus was used to train various input nonlinearity and full-rank models, which were tested on the second half; their performance was quantified by means of the predictive power (Sahani & Linden, 2003b). This is a performance measure which takes into account trial-by-trial variability of the response. It has an expected value of 1 for a model that predicts 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 100000 time points, which served to study the overfitting properties of the various models.

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, the stimulus values were squared and linearly filtered by $k(\tau)$.

II. Two filters, additive. $P(t) = \sum_{\tau=0}^{\tau_{\max}} k_1(\tau) s(t-\tau) + \sum_{\tau=0}^{\tau_{\max}} k_2(\tau) s(t-\tau)^2$. 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.

III. Two filters, multiplicative. $P(t) = c_1 + (\sum_{\tau=0}^{\tau_{\max}} k_1(\tau) s(t-\tau)) \, (c_2 + \sum_{\tau=0}^{\tau_{\max}} k_2(\tau) |s(t-\tau)|)$. Here, the firing rate of a linear process is multiplicatively modulated by another process acting on the absolute stimulus values.

The third process is designed to mimic systems with a certain degree of adaptation, or response normalization (Schwarz et al., 2002). The bilinear model and the full-rank model are

explicitly not able to capture the properties of such a model (although there are extensions of the input nonlinearity model that can capture them; see Discussion); it is included to test the behavior of the models when they fail. We trained and tested the following models on the following datasets, using minimal regularization to enable clean comparisons:

1. **bilin(1) and SVD bilin(1).** One-term bilinear models (i.e., $\hat{r}(t) = c + \sum_\tau w_\tau f(s(t-\tau))$) fit through the ALS procedure and through taking the first SVD component of the corresponding full-rank model.

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 2.3).

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; see section 2.2. An SVD term was included when its eigenvalue was larger than 1/4 of the leading eigenvalue.

Figure 5 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 multiplicative 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.

[Figure 5 about here.]

What do the models look like? Figure 6 shows the models for process II. The bilinear model correctly picks out (noisy versions of) the two temporal filters and input nonlinearities. The full-rank model also picks up these terms, but the two leading SVD terms show some undesirable mixing of the linear and quadratic terms; the predictive power is lower than that of the bilinear model.

[Figure 6 about here.]

To compare the various models, their performance is shown against the length of the trial in figure 7. These graphs confirm our intuitions:

A. On dataset II, the one-term bilinear model cannot capture the structure of the spike generating process, which the full-rank and the bilin(2) model do manage. The full-rank model overfits for small datasets; for large datasets, the performances of the bilin(2) and the full-rank models converge.

B. The effect of the output nonlinearity is illustrated using dataset I and the bilin(1) model. The performance is consistently above the performance without output nonlinearity by a small amount.

C. The bilinear model esimated by ALS generally outperforms the same model estimated by taking the first SVD component of the corresponding full-rank model, here shown for dataset III. On this dataset, the full-rank model outperforms the bilinear model, but only for large numbers of data points (because it captures more of the complexity of the spike generating process); for smaller numbers, the bilinear model is superior due to less overfitting.

D. On small datasets, taking the leading SVD terms of a full-rank model improves performance, because it induces some post-hoc smoothing of the receptive field.

[Figure 7 about here.]

Finally, we fitted an NLNP model, or a Generalized Bilinear Model, to a spike train, instead of a spike rate, using the point process likelihood, as described in section 2.9. To generate the spikes, we used a quadratic input nonlinearity $f(s) = s^2$ and an exponential output nonlinearity $g(.) = \exp(.)$, so the model for the underlying spike rate was $r(t) = \exp(c + \sum_\tau w_\tau s(t-\tau)^2)$. Performing alternating gradient ascent on the objective function, which included regularization terms, resulted in the parameters shown in figure 8. 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, although this would be straightforward to implement.

[Figure 8 about here.]

4 Demonstration on real data

We estimated the full-rank model (using the velocity signal as the stimulus) and an input nonlinearity model (with two terms, position and velocity) using spike-rate data from a cell in rodent barrel cortex, stimulated by a white noise whisker displacement stimulus (Petersen & Montemurro, 2006). The models are shown in figure 9. They both 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 bilinear model demonstrates that this cell is more responsive to velocity than to position of the whisker; this is evident from the shape of the temporal filters (left panel). 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 results are in agreement with conculsions drawn from spike-triggered covariance (STC) analysis of the same data in Petersen & Montemurro (2006) and with previous results (Pinto et al., 2000; Arabzadeh et al., 2003, 2005).

Although both models are structurally 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 respectively, both on cross-validation data. Evidently, the reason for this is the parabolic shape of the input nonlinearity.

[Figure 9 about here.]

5 Discussion

The bilinear model (also called the Hammerstein or NL cascade model) and the full-rank model provide useful non-linear 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 regularization methods presented in this paper allow for a careful analysis of the model parameters and their error bars. The small number of paramefters in the bilinear and NLNP models ($\dim(\mathbf{w}) + \dim(\mathbf{b})$ parameters) makes the data requirements modest; the full-rank model requires more data for estimation (it has $\dim(\mathbf{w}) \cdot \dim(\mathbf{b})$ paramters). The feasibility of fitting, and utility of, the models was 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 & Bialek, 1988; Schwarz et al., 2002; Petersen & Montemurro, 2006), 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 too 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 normalization. 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; in the limiting case that $\mathbf{w}$ is a delta-function (consisting of 0's and a single 1), this type of LNP model is mathematically equivalent to a bilinear model. LNP models, bilinear models and full-rank models will have certain overlaps in the types of neurons they can successfully model. 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).

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 & Naka, 1973). Recently it has found applications in e.g. characterizing subthreshold dynamics in barrel cortex (Webber & Stanley, 2004). Since (in theory) these expansions span all non-linear models, the models introduced in this paper can also be phrased as restricted Volterra-Wiener expansions (e.g. the input nonlinearity 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 & 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).

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 regularization techniques and error bar estimation. Error bar estimation 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 under 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.

Bilinear model versus full-rank SVD components. In the model experiments, the bilinear model outperformed the SVD decomposition of the full-rank model, confirming our expectations, since using SVD minimizes the distance between the SVD terms and the full-rank model, whereas minimizing the distance between the real and predicted firing rate is the more appropriate objective. However, if it is not too costly 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).

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., 2006). 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., 2006). 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 & Kearney, 2001). Finally, in this paper we assumed discrete basis functions in the τ direction. Of course, any other basis set can 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 $\mathbf{b}$.

Acknowledgements

We thank Rasmus Petersen for the data used in the example of figure 9 and for interesting discussions, Zoubin Ghahramani for suggestions, and Quentin Huys for comments on the manuscript. M.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 $\{x_q\}_{q=1}^N$:

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

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 2.7.

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 stated in section 2.1, the matrix $B_{t\tau} = \sum_i b_i M_{t\tau i}$ is redefined and used to estimate $\mathbf{w}$ through $\mathbf{w} = (\mathrm{B}^{\mathrm{T}}\mathrm{B})^{-1}\mathrm{B}^{\mathrm{T}}\mathbf{r}$. That is, it appears as $\mathrm{B}^{\mathrm{T}}\mathrm{B}$ and as $\mathrm{B}^{\mathrm{T}}\mathbf{r}$. Both of these terms include a sum over t and this may be big (both in computing time 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 $[\mathrm{B}^{\mathrm{T}}\mathrm{B}]_{\tau\tau'} = \sum_{ii'} Q_{\tau i\tau' i'} b_i b_{i'}$ and $[\mathrm{B}^{\mathrm{T}}\mathbf{r}]_\tau = \sum_i Y_{\tau i} b_i$ 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: $[\mathrm{W}^{\mathrm{T}}\mathrm{W}]_{ii'} = \sum_{\tau\tau'} Q_{\tau i\tau' i'} w_\tau w_{\tau'}$ and $[\mathrm{W}^{\mathrm{T}}\mathbf{r}]_i = \sum_\tau Y_{\tau i} w_\tau$.

C. Regularization of b in the input nonlinearity model

Note that the regularization 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 $\frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathrm{D}^b\mathbf{b}$ 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 penalize the first derivative of f:

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

This equation holds if we define D^b as

$$D^b_{ij} = \lambda \int \frac{df_i(x)}{dx}\frac{df_j(x)}{dx} dx,$$

where $f_i(.)$ 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 paper, hence we derive an expression for D^b taylored 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

$$\begin{aligned}\lambda \int \left(\frac{d^2 f}{dx^2}\right)^2 dx &= \lambda \sum_q (\text{gradient just left of } x_q - \text{gradient just right of } x_q)^2 \\ &= \sum_{ij} b_i b_j D^b_{ij}\end{aligned}$$

where $D^b_{ij} = \lambda \int f_i''(x) f_j''(x) dx$ 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 regularizing priors, that e.g. penalize higher derivatives (or the Laplacian in higher dimensions; see also Poggio et al., 1985), can be derived in the same way (by replacing e.g. $\frac{\partial}{\partial x}$ in equation 6 by a different linear operation and finding the corresponding D^b).

D. Regularization of C in the full-rank model

Again using the symbol x for the stimulus value, and writing the receptive field as $a(\tau, x) = \sum_i C_{\tau i} f_i(x)$, the penalty term is defined to be $\sum_\tau \int_x \alpha \left(\frac{da}{d\tau}\right)^2 + \beta \left(\frac{d^2 a}{dx^2}\right)^2$, 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 $\frac{da}{d\tau} = \sum_i (C_{\tau i} - C_{\tau-1,i}) f_i(x)$ and $\frac{d^2 a}{dx^2} = \sum_i w_{\tau i} \frac{d^2 f_i}{dx^2}$. The expression for the penalty term is

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

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

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

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

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

Here x_n are the points in the discretized 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 vectorizing (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 e.g. 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 $\mathbf{b}$, and then fixing $\mathbf{w}$ to $\mathbf{w}^1$, and drawing $\mathbf{b}^2$ from a distribution dependent on $\mathbf{w}$, etc. In this way, $\{\mathbf{w}^n, \mathbf{b}^n\}$ will form a set of samples from $P(\mathbf{w}, \mathbf{b} | \mathbf{r}, \mathrm{M}, \mathrm{D}^w, \mathrm{D}^b, \hat{\sigma}^2)$. These probability distributions are

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

and

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

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 are also good as starting points for Gibbs sampling.

F. Adaptive regularization of bilinear models

In the main text we have used an alternating least squares procedure to estimate the parameter vectors of an input-nonlinearity model. It was easy to incorporate prior covariance matrices for the parameter vectors. In many cases, however, it is not clear what a good prior is - e.g., how much smoothing is 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 parameters θ (e.g. λ, α and β in Appendices C and D). Whilst one can always use the ad-hoc approach of hand-tuning these parameters (through intuition or cross-validation), we developed a principled Bayesian approach using a Variational Bayes EM algorithm (Dempster et al., 1977; Beal, 2003) to fit this hierarchical model, treating λ, α and β as hyperparameters and using the variational Bayes approach to approximately integrate over the parameters $\mathbf{b}$ and $\mathbf{w}$. 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 $\mathbf{w}$ and $\mathbf{b}$, respectively. $\mathrm{S}^w(\theta^w)$ and $\mathrm{S}^b(\theta^b)$ are the prior covariance matrices that are responsible for regularizing the estimates, and depend on (possibly multidimensional) parameters $\theta^{w,b}$, which have to be learnt. $\hat{\sigma}^2$ 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, a function $\mathcal{F}$ appears in the algorithm; this is the portion of the free energy, a lower bound on the log-likelihood, that depends on the θ's.

We initialize the algorithm 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 initialized 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 now:

1. Define
   - $\mathrm{C}^w \leftarrow \Sigma^w + \mathbf{u}^w[\mathbf{u}^w]^{\mathrm{T}}$
   - $\mathrm{Q}^w_{ij} \leftarrow \sum_{tkl} \mathrm{C}^w_{kl}\mathrm{M}_{tik}\mathrm{M}_{tjl}$
   - $[\mathbf{v}^w]_i \leftarrow \sum_{tk}[\mathbf{u}^b]_k\mathrm{M}_{tik}r(t)$.
2. Update Σ^w and $\mathbf{u}^w$ according to
$$\Sigma^w \leftarrow \mathrm{S}^w\left[\frac{\mathrm{Q}^w\mathrm{S}^w}{\hat{\sigma}^2} + \mathrm{I}\right]^{-1}$$
$$\mathbf{u}^w \leftarrow \Sigma^w\frac{\mathbf{v}^w}{\hat{\sigma}^2}$$
3. Perform the analogous operations of steps 1 and 2 for Σ^b and $\mathbf{u}^b$.
4. Update $\hat{\sigma}^2$ according to
$$\hat{\sigma}^2 \leftarrow \frac{1}{T}\left(\mathbf{r}^t\mathbf{r} - 2\sum_{tik}[\mathbf{u}^w]_i[\mathbf{u}^b]_k r(t)\mathrm{M}_{tik} + \sum_{tijkl} C^w_{ij}C^b_{kl}\mathrm{M}_{tik}\mathrm{M}_{tjl}\right)$$
   where $T = \mathrm{length}(\mathbf{r})$, $\mathrm{C}^w = \mathbf{u}^w[\mathbf{u}^w]^{\mathrm{T}} + \Sigma^w$ and $\mathrm{C}^b = \mathbf{u}^b[\mathbf{u}^b]^{\mathrm{T}} + \Sigma^b$.
5. Do gradient ascent on the function
$$\mathcal{F} = -\frac{1}{2}\log|\mathrm{S}^w| - \frac{1}{2}\log|\mathrm{S}^b| - \frac{1}{2}\mathrm{trace}\left(\mathrm{C}^w[\mathrm{S}^w]^{-1} + \mathrm{C}^b[\mathrm{S}^b]^{-1}\right)$$
   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 $\mathrm{D}^w = [\mathrm{S}^w]^{-1}$,

so that the algorithm can be implemented using either form of regularization),

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

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.

That concludes the algorithm. Note that step 1 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 & Linden (2003a). The latter paper also presents some nice formulations of tunable prior covariance matrices $\mathrm{S}(\theta)$.

References

Ahrens, M. B., Linden, J. F., & Sahani, M. 2006. Multilinear spectrotemporal models for predicting auditory cortical responses. Association for Research in Otolaryngology abstract.

Arabzadeh, E., Petersen, R. S., & Diamond, M. E. 2003. Encoding of whisker vibration by rat barrel cortex neurons: implications for texture discrimination. *J Neurosci 23*, 9146–9154.

Arabzadeh, E., Zorzin, E., & Diamond, M. E. 2005. Neuronal Encoding of Texture in the Whisker Sensory Pathway. *PLoS Biol 3*, 155–165.

Bai, E. W. 1998. An optimal two-stage identification algorithm for Hammerstein-Wiener nonlinear systems. *Automatica 34*, 333–338.

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

Berman, M. & Turner, T. R. 1992. Approximating Point Process Likelihoods with GLIM. *Applied Statistics 41*, 31–38.

Breiman, L. & Friedman, J. H. 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.

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

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

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, G. C., Ohzawa, I., & Freeman, R. D. 1995. Receptive-field dynamics in the central visual pathways. *Trends Neurosci 18*, 451–458.

Dempster, A. P., Laird, N. M., & Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B 39*, 1–38.

Depireux, D. A., Simon, J. Z., Klein, D. J., & Shamma, S. A. 2001. Spectro-Temporal Response Field Characterization With Dynamic Ripples in Ferret Primary Auditory Cortex. *J Neurophysiol 85*, 1220–1234.

Fishbach, A., Nelken, I., & Yeshurun, Y. 2001. Auditory edge detection: a neural model for physiological and psychoacoustical responses to amplitude transients. *J Neurophysiol 85*, 2303 – 2323.

Hastie, T. J. & Tibshirani, R. J. 1999. *Generalized additive models*. Monographs on Statistics and Applied Probability 43, Chapman & Hall/CRC.

Hunter, I. W. & Korenberg, M. J. 1986. The Identification of Nonlinear Biological Systems: Wiener and Hammerstein Cascade Models. *Biological Cybernetics 55*, 135–144.

Juusola, M., Weckström, M., Uusitalo, R. O., Korenberg, M. J., & French, A. S. 1995. Nonlinear Models of the First Synapse in the Light-Adapted Fly Retina. *J Neurophysiol 74*, 2538–2547.

Linden, J. F., Liu, R. C., Sahani, M., Schreiner, C. E., & Merzenich, M. M. 2003. Spectrotemporal structure of receptive fields in areas AI and AAF of mouse auditory cortex. *J Neurophysiol 90*, 2660–2675.

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

Machens, C. K., Wehr, M. S., & Zador, A. M. 2004. Linearity of cortical receptive fields measured with natural sounds. *J Neurosci 24*, 1089–1100.

MacKay, D. J. C. 1994. Bayesian non-linear modelling for the prediction competition. In *ASHRAE Transactions, V.100, Pt.2*, Atlanta Georgia, pp. 1053–1062. ASHRAE.

MacKay, D. J. C. 2004. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press.

Marmarelis, P. Z. & Naka, K. I. 1973. Nonlinear analysis and synthesis of receptive-field responses in the catfish retina. I. Horizontal cell leads to ganglion cell chain. *J Neurophysiol 36*, 605–618.

McCullagh, P. & Nelder, J. 1989. *Generalized Linear Models*. Chapman and Hall.

Narendra, K. S. & Gallman, P. G. 1966. An Iterative Method for the Identification of Nonlinear Systems Using a Hammerstein Model. *IEEE Trans 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.

Petersen, R. S. & Montemurro, M. 2006. Encoding of multiple features of a dynamic tactile stimulus by neurons in primary somatosensory cortex. Submitted.

Pillow, J. W., Paninski, J., Uzzell, V. J., Simoncelli, E. P., & Chichilnisky, E. J. 2005. Prediction and Decoding of Retinal Ganglion Cell Responses with a Probabilistic Spiking Model. *J Neurosci 25*, 11003–11013.

Pillow, J. W. & Simoncelli, E. P. 2006. Dimensionality reduction in neural models: an information-theoretic generalization of spike-triggered average and covariance analysis. *Journal of Vision 6*, 414–428.

Pinto, D. J., Brumberg, J. C., & Simons, D. J. 2000. Circuit Dynamics and Coding Strategies in Rodent Somatosensory Cortex. *J Neurophysiol 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, J. F. 2003a. Evidence optimization techniques for estimating stimulus-response functions. In S. Becker, S. Thrun, & K. Obermayer (Eds.), *Advances in Neural Information Processing Systems 15*, Volume 15, pp. 109–116. Cambridge, MA: MIT Press. Available online via http://www.nips.cc.Proceedings/.

Sahani, M. & Linden, J. F. 2003b. How linear are auditory cortical responses? In S. Becker, S. Thrun, & K. Obermayer (Eds.), *Advances in Neural Information Processing Systems 15*, pp. 109–116. Cambridge, MA: MIT Press. Available online via http://www.nips.cc.Proceedings/.

Schwarz, O., Chichilnisky, E. J., & Simoncelli, E. P. 2002. Characterizing neural gain control using spike-triggered covariance. In T. G. Dietterich, S. Becker, & Z. Gharamani (Eds.), *Adv Neural Information Processing Systems*, Volume 14, pp. 269–276. Cambridge, MA: MIT Press. Available online via http://www.nips.cc.Proceedings/.

Sharpee, T., Rust, N. C., & Bialek, W. 2004. Analyzing Neural Responses to Natural Signals: Maximally Informative Dimensions. *Neural Computation 16*, 223–250.

Simoncelli, E. P., Pillow, J., Paninski, L., & Schwartz, O. 2004. Characterization of neural responses with stochastic stimuli. In M. Gazzaniga (Ed.), *The Cognitive Neurosciences* (3 ed.)., pp. 327–338. Cambridge, MA: MIT Press.

Strang, G. 1988. *Linear Algebra and its Applications* (3 ed.). Brooks Cole.

Suits, D. B., 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, U. T., Fellows, M. R., Donoghue, J. P., & Brown, E. N. 2005. A Point Process Framework for Relating Neural Spiking Activity to Spiking History, Neural Ensemble, and Extrinsic Covariate Effects. *J Neurophysiol 93*, 1074–1089.

Webber, R. M. & Stanley, G. B. 2004. Nonlinear Encoding of Tactile Patterns in the Barrel Cortex. *J Neurophysiol 91*, 2010–2022.

Westwick, W. T. & Kearney, R. E. 2001. Separable Least Squares Identification of Nonlinear Hammerstein Models: Application to Stretch Reflex Dynamics. *Annals of Biomedical Engineering 29*, 707–718.

Young, E. D. & Calhoun, B. M. 2005. Nonlinear Modeling of Auditory-Nerve Rate Responses to Wideband Stimuli. *J Neurophysiol 94*, 4441–4454.

Young, F. W., 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.

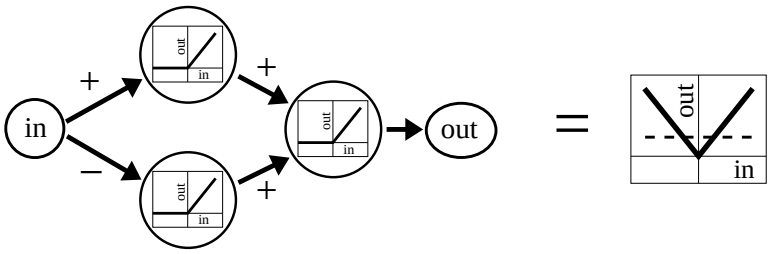

Figure 1: Schematic network with a symmetric input-output relation. −: inhibitory connection. +: excitatory connection. All "neurons" are half-wave rectifiers. The output will be insensitive to the sign of the input; hence a linear fit to the I-O function (dashed line) is constant.

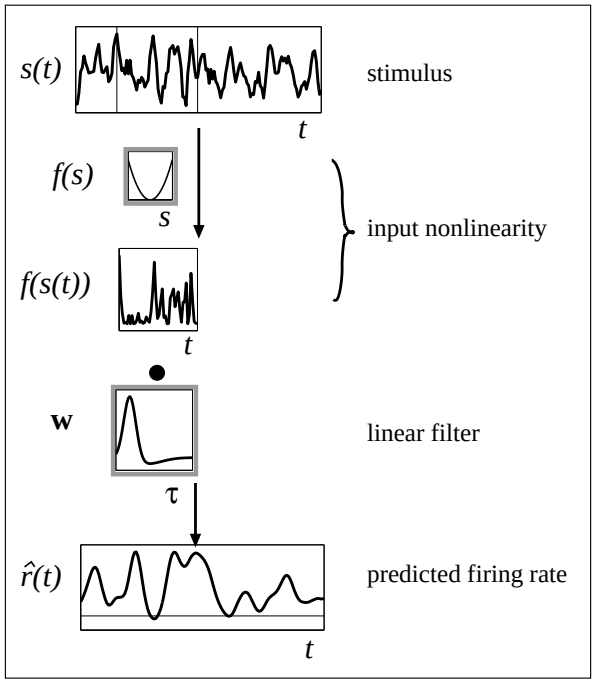

Figure 2: Schematic view of the bilinear model. It consists of two stages, or neural processing operations. First, the stimulus values are transformed by an input nonlinearity $f(.)$, and secondly, a temporal filter $\mathbf{w}$ acts on the transformed stimulus values to form a predicted spike rate. $\mathbf{w}$ and $f(.)$ are both unknown and to be learnt from the data. An output nonlinearity (section 2.8) is optional and not shown in the figure. If the input nonlinearity stage is removed, or, equivalently, $f(.)$ set to the identity, then the model reduces to the linear model. The free parameters of the model are surrounded by gray boxes.

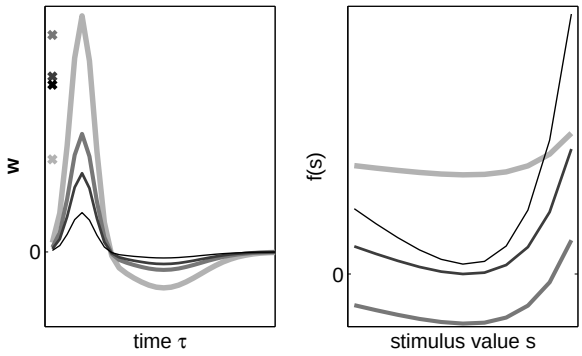

Figure 3: Degeneracies of the bilinear model. *Left:* the temporal filter has a multiplicative degeneracy. The crosses at $\tau = 0$ are the values of the constant terms $c = w_1 \cdot b_1$. *Right:* the input nonlinearity has both a multiplicative and an additive degeneracy. The parameters shown here are equivalent configurations of the bilinear model, i.e. each pair of $\mathbf{w}$ and $f(.)$ shown represent the same input-output relationship of the model. If the degeneracies are not removed, estimated error bars on $\mathbf{w}$ and f will be oversized.

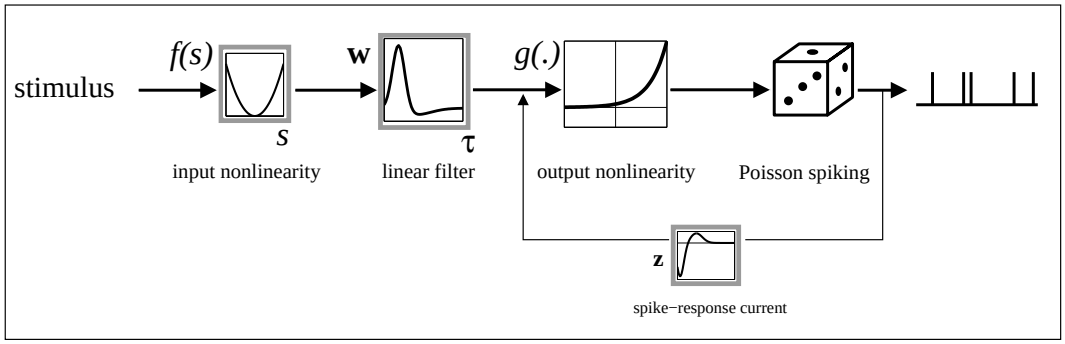

Figure 4: Schematic view of the bilinear model with output nonlinearity. The parameters in the gray boxes are learnt from the data. The spike-response current is optional.

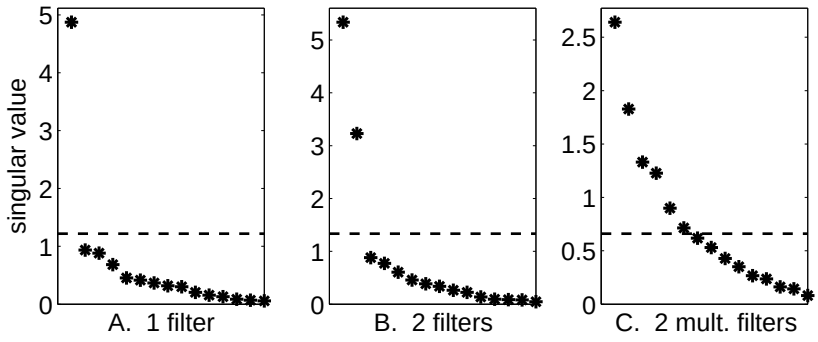

Figure 5: Singular values of the SVD decomposition of the full-rank model trained on model data A-C. Only the first 15 eigenvalues are shown. Note that in these examples, the number of significant singular values are indicative of the complexity of the spike generating process (dashed lines show the theshold for inclusion).

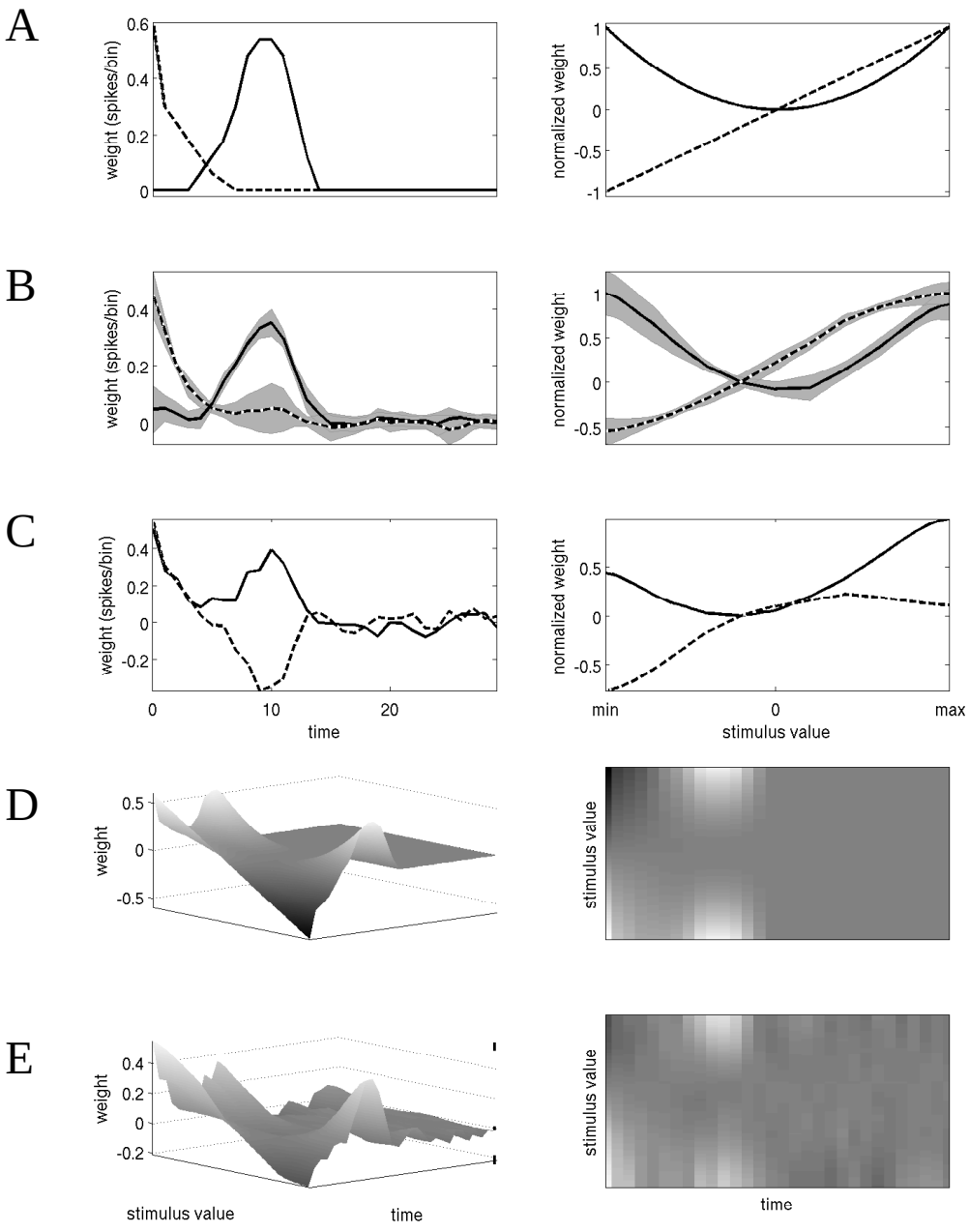

Figure 6: Model fits to data generated from the two-filter process (II). *A:* true temporal filters $\mathbf{w}^1$ and $\mathbf{w}^2$ (*left*) and input nonlinearities f^1 and f^2 (*right*). *B:* estimated bilinear model with two terms. The error bars of one standard error (gray) are calculated by Gibbs sampling. *C:* first two SVD terms of the estimated full-rank model. *D:* the true full-rank model given by $\mathrm{C} = \mathbf{w}^1\mathbf{b}^{1\mathrm{T}} + \mathbf{w}^2\mathbf{b}^{2\mathrm{T}}$, 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.

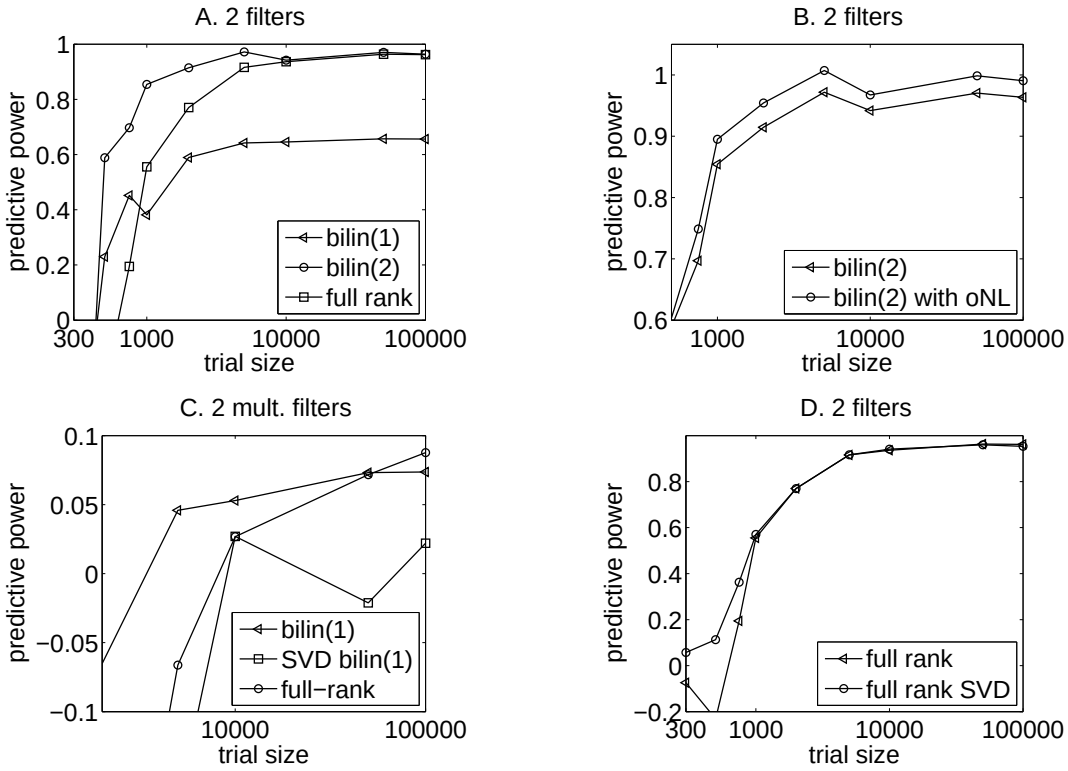

Figure 7: Performance of the models on datasets I-III, averaged over 10 instantiations of the random stimulus. Details can be found in the main text.

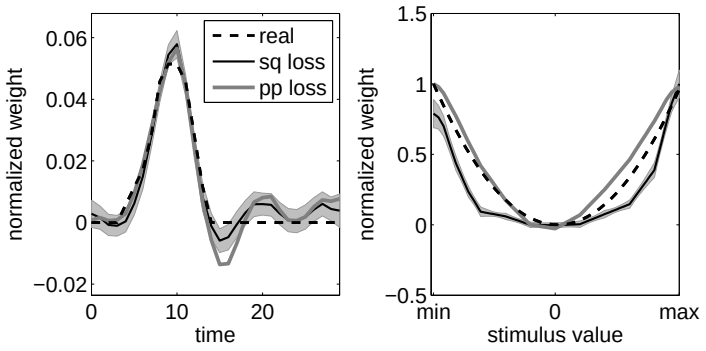

Figure 8: Point process model fits. *Dotted:* true filters. *Black:* filters found through minimizing the squared error. *Gray:* filters found by maximizing the point-process likelihood, using the exponential output nonlinearity. The normalized shapes of the filters are similar, but the input nonlinearity under the squared error appears to be somewhat biased towards an overly-smooth U-shape.

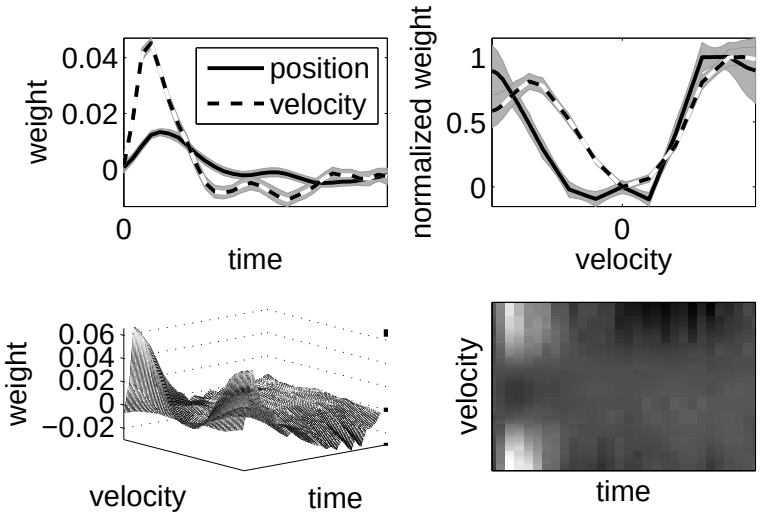

Figure 9: Bilinear and full-rank models applied to real whisker barrel cortex data Petersen & Montemurro (2006). *Top:* input nonlinearity model on position and velocity (*left:* temporal filters, *right:* input nonlinearities). The grey area shows one standard error, obtained by Gibbs sampling. *Bottom:* full-rank model on velocity, shown as a surface (*left*) and a matrix (*right*). 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.