Spike-Based Probabilistic Inference in Analog Graphical Models Using Interspike-Interval Coding*

Andreas Steimer
Institute of Neuroinformatics, University of Zürich and ETH Zürich, Zürich, Switzerland
asteimer@ini.phys.ethz.ch

Rodney Douglas
Institute of Neuroinformatics, University of Zürich and ETH Zürich, Zürich, Switzerland
rjd@ini.phys.ethz.ch

March 6, 2013

Abstract

Temporal spike codes play a crucial role in neural information processing. In particular, there is strong experimental evidence that interspike intervals (ISIs) are used for stimulus representation in neural systems. However, there are very few algorithmic principles that exploit the benefits of such temporal codes for probabilistic inference of stimuli or decisions. Here, we describe and rigorously prove the functional properties of a spike-based processor that uses ISI distributions to perform probabilistic inference. The abstract processor architecture serves as a building block for more concrete, neural implementations of the Belief-Propagation (BP) algorithm in arbitrary graphical models (e.g. Bayesian Networks and Factor Graphs). The distributed nature of graphical models matches well with the architectural and functional constraints imposed by biology. In our model, ISI distributions represent the BP-messages exchanged between factor nodes, leading to the interpretation of a single spike as a random sample that follows such a distribution. We verify the abstract processor model by numerical simulation in full graphs, and demonstrate that it can be applied even in presence of analog variables. As a particular example, we also show results of a concrete, neural implementation of the processor, although in principle our approach is more flexible and allows for different neurobiological interpretations. Furthermore, electrophysiological data from area LIP during behavioral experiments is assessed in the light of ISI coding, leading to concrete testable, quantitative predictions and a more accurate description of these data compared to hitherto existing models.

*Author copy of the article 'A. Steimer & R. Douglas. Spike-Based Probabilistic Inference in Analog Graphical Models Using Interspike-Interval Coding, *Neural Computation*, 25:2303-2354, 2013', http://www.mitpressjournals.org/toc/neco/25/9,

1 Introduction

Rapid and unambiguous interpretation of the environment is a necessary condition for survival. These properties are granted already at the neuronal level, since the interspike interval (ISI) distribution distinctly encodes a stimulus and the coding process converges within $\sim 8-12$ spikes after the stimulus has changed (Fairhall, Lewen, Bialek, & de Ruyter van Steveninck, 2001; Lundstrom & Fairhall, 2006). Furthermore, the amount of information about a stimulus that is carried by a spike depends strongly on the spikes preceding ISI (Reich, Mechler, Purpura, & Victor, 2000; Fairhall et al., 2001; Lundstrom & Fairhall, 2006; Shih, Atencio, & Schreiner, 2011). These properties are true across both species and sensory modalities, and are due to synergetic effects between spikes, that cannot be explained by rate modulations alone (Shih et al., 2011).

Taken together these studies raise the question of how stimulus information contained in the ISIs of low-level sensory areas are used by subsequent processing stages to (a) infer the stimulus with higher precision by combining different sensory modalities (cue combination), and (b) induce higher level information processing, leading to e.g. the appropriate selection of motor actions in response to the thus inferred stimulus (decision making). Both of these tasks are crucial to neural systems and are typical examples of probabilistic inference.

In this paper we provide an abstract, mathematical description of a spike-based processor that can perform such inference in very general settings. By combining information processing based on ISIs with message passing schemes in graphical models (Bishop, 2006; MacKay, 2003; Kschischang, Frey, & Löliger, 2001; Löliger, 2004), we are able to implement the Belief Propagation (BP) algorithm (Kschischang et al., 2001; Löliger, 2004; Bishop, 2006), which permits the marginal probabilities of variables to be inferred (e.g. the stimulus or motor commands).

Graphical models such as Factor Graphs (FGs), Bayesian Networks (BNs) or Markov Random Fields (MRFs) constitute a canonical framework for solving probabilistic inference problems. This high level of generality makes this framework very attractive for brain modeling (Deneve, 2007; Rao, 2004; Ott & Stoop, 2006; Steimer, Maass, & Douglas, 2009; Ma, Beck, Latham, & Pouget, 2006). Here we focus exclusively on 'Forney Factor Graphs (FFGs)' (Forney, 2001; Löliger, 2004), which, together with ordinary FGs, allow for sparser graph-topologies compared to BNs/MRFs for the same represented probability distribution (this aspect will be defined more clearly in section 1.1).

Graph-based representations of inference problems give rise to message-passing algorithms such as BP, that allow inference to be performed in a biologically plausible way by purely local computations. That is, information exchange through messages occurs exclusively between neighboring nodes of the graph, thereby dispensing with the need for any global observer. Moreover, when formulated in FFGs, BP leads to a single, stereotyped update rule for computing outgoing messages from incoming ones. This property suggests a canonical strategy that is likely to be utilized by the brain, e.g. in cortex' canonical microcircuitry (Douglas & Martin, 2004).

In the machine learning domain, the messages employed by BP are regarded as probability distributions and so provide an elegant computational meaning for distributions of ISIs in

neural systems. In this view spiking behavior is an ongoing sampling process. That is, a single spike is interpreted as a random sample whose numerical value is determined by its preceding ISI. Since ISIs are analog quantities, our method allows BP even to be applied to graphical models that contain analog variables. This feature is notoriously difficult to achieve in traditional software implementations of BP (Sudderth, Ihler, Freeman, & Willsky, 2003) and hence provides a clear-cut example of the benefits derived from biologically inspired information processing.
Sampling methods can also be used directly for solving inference problems, e.g. in decision making tasks. Most prominently, Markov Chain Monte Carlo(MCMC) methods (MacKay, 2003; Bishop, 2006) such as Gibbs sampling (Geman & Geman, 1984) provide a set of samples from the joint probability distribution, which renders marginalization a simple matter of coordinate projection. Such approaches are complementary to BP since, on the one hand, they generally suffer from both, the curse of dimensionality and poor convergence (MacKay, 2003). On the other hand however these methods can still be applied in cases where BP performs badly, for example in certain graphs with loopy topology.
Furthermore, literal application of the 'Sum Product Rule (SPR)' (Löliger, 2004) (the message update rule used by BP), often leads to intractable integrals in the analog case (Löliger, 2004; Sudderth et al., 2003; Dauwels, Korl, & Löliger, 2006; Isard, 2003). By contrast, the method presented here naturally avoids the explicit computation of these integrals by combining a sample-based MCMC-like message update scheme with the SPR (Dauwels et al., 2006).
Instead of solving a particular cognitive task, we demonstrate our method's high level of generality by statistical analyses that verify the functionality of the processor. Firstly, we show results of literal software implementations of the processor that, apart from being spike-based, disregard the use of neural elements for the underlying computations. The verification is made for two different graph topologies and for various random parameterizations of the nodes (factors) that constitute the FFG in both cases. Consequently we then show how a concrete implementation of the processor based on neural elements can be obtained. The latter implementation however is still abstract enough to be consistent with at least two different neuroanatomical interpretations.

The paper is organized as follows: In the remainder of the introduction we provide some general background information about FFGs and BP. We then lay out in the results section 2.1 the mathematical description of an abstract processor, whose literal implementation in software performs BP in analog FFGs. The abstract processor lends itself also to a neural architecture for its concrete implementation (section 2.2). In both cases statistical analyses evaluate performance by comparing our method to ordinary, discretized BP. Subsequently, ISI coding is applied for the qualitative and quantitative modeling of electrophysiological data, leading to concrete predictions in the respective experimental setup (section 2.3). In the discussion (section 3), this we continue on that issue by presenting experimental evidence for ISI coding in neural systems, along with biophysical mechanisms that are relevant for the functioning of the concrete processor. The Materials and Methods (section 4) then contains some mathematical derivations, whose results are needed in sections 2.1&2.2, along with

more detailed descriptions of the various methods used in the paper.

1.1 Basics of Forney Factor Graphs and Belief-Propagation

Like other graphical models, a FFG is a graph based representation of a factorized joint probability distribution/density of a set of variables. In a FFG, the nodes of the graph correspond to the individual factors of that factorization and the edges to variables. The factor nodes are defined by nonnegative functions of the variables (edges) these nodes are attached to (figure 1). The seemingly severe restriction that variables are represented by edges, and as such may be connected to at most two factor nodes, can be easily circumvented by including 'equality constraint' factor nodes (Löliger, 2004). That is, the factorization is expanded to include additional, deterministic factors, which enforce equality among the variables they are attached to. We will apply our methods to such a factor in section 2.1.2).

Using the structure of a graphical model, the goal of inference by means of BP is to simultaneously compute the marginal probability densities $P(x_i)$ of all n variables X_i in the graph:

$$P(x_i) = \int_{D^{n-1}} P(x_1, ..., x_n)\, \mathrm{d}v_i \tag{1}$$

where D is the domain of each variable (which for simplicity is assumed to be the same for all variables); D^{n-1} is the cartesian product of the domains of all variables $j \neq i$ and $\mathrm{d}v_i := \prod_{j \neq i} \mathrm{d}x_j$. In general, computing these integrals analytically is not possible. Also, approximations using finite sums are computationally prohibitive due to the combinatorial nature of the problem. BP however avoids at least this combinatorial aspect by solving equation 1 through the distribution of messages between neighboring nodes in the graph representing $P(x_1, ..., x_n)$ (see figure 1):
Suppose for the moment that all variables were discrete with finite domain and the integrals in equation 1 were replaced by corresponding sums. In a FFG, a message $m_{f_1 \to f_2}(x)$ sent from a node f_1 to a node f_2 along the variable X provides a biased estimate about the states of X by summarizing the subgraph $\mathcal{G}_1$ 'behind' (with respect to the direction of the message) f_1. This means that in a marginalization task $m_{f_1 \to f_2}(x)$ contains all sums associated with variables in $\mathcal{G}_1$(Löliger, 2004). Since (in a tree-structured graph) these summations do not depend on the variables in $\mathcal{G}_2$, the subgraph 'in front of' f_2, they need not be repeated for each (joint) state of the variables in $\mathcal{G}_2$. This property is a consequence of the distributive law (Aji & McEliece, 2000), and the reason for the much greater efficiency of BP compared to marginalization by direct summation (equation 1).
In general, BP messages following the Sum Product Rule (SPR) in a FFG based on analog variables are computed as follows (in case of discrete variables the integrals have to be replaced by corresponding sums, which has motivated the term 'Sum Product Rule'):

$$um_{f_i \to f_j}(x_{i_j}) := \int_{D^{n-1}} f_i(\boldsymbol{x}_{\mathcal{N}(i)}) \prod_{k \in \mathcal{N}(i)/j} m_{k \to i}(x_{i_k}) \, \mathrm{d}v_{\mathcal{N}(i)/j} \tag{2}$$

$$N_{f_i \to f_j} := \int_D um_{f_i \to f_j}(x_{i_j}) \, \mathrm{d}x_{i_j} \tag{3}$$

$$m_{f_i \to f_j}(x_{i_j}) := \frac{um_{f_i \to f_j}(x_{i_j})}{N_{f_i \to f_j}} \tag{4}$$

where f_i is the function associated with factor node i; $x_{i_j} \in D$ is a value of X_{i_j}, the variable connecting nodes i and j; $\mathcal{N}(i)/j$ indicates the set of neighbors of node i (excluding neighbor j); n the cardinality of $\mathcal{N}(i)$ (= branching degree of i); $\boldsymbol{x}_{\mathcal{N}(i)} \in D^n$ is the vector of values of variables in $\mathcal{N}(i)$; $\mathrm{d}v_{\mathcal{N}(i)/j} := \prod_{k \in \mathcal{N}(i)/j} \mathrm{d}x_{i_k}$; $m_{f_i \to f_j}(x_{i_j})$ is the value the normalized message passed from node f_i to node f_j provides for value x_{i_j} of variable X_{i_j} and $um_{f_i \to f_j}(x_{i_j})$ is the corresponding unnormalized message; $N_{f_i \to f_j}$ is the corresponding normalization constant. Normalization of messages $um_{f_i \to f_j}(x_{i_j})$ is typically needed to avoid exploding and/or diminishing message values, a fact that further justifies their representation as probability distributions.
Using the messages based on equation 4, estimates $B(x_{i_j})$ of the marginals $P(x_{i_j})$ (equation 1) can be computed in the following way:

$$B(x_{i_j}) := \frac{m_{f_i \to f_j}(x_{i_j}) \, m_{f_j \to f_i}(x_{i_j})}{\int_D m_{f_i \to f_j}(x_{i_j}) \, m_{f_j \to f_i}(x_{i_j}) \, \mathrm{d}x_{i_j}} \tag{5}$$

For all tree structured FFGs it is guaranteed that $B(x_{i_j}) = P(x_{i_j})$. Loopy graphs in contrast generally yield $B(x_{i_j})$s that are just approximations to the $P(x_{i_j})$s. However even in the loopy case these approximations are sometimes surprisingly exact (Yedidia, Freeman, & Weiss, 2005). $B(x_{i_j})$ is typically referred to as the *belief* of variable x_{i_j}.

Unlike BNs or MRFs, FFGs belong to the sparsest types of graphical models. That is, all other graphical models can be easily converted to the FG/FFG formulation, without the need to increase the maximum branching degree of the associated nodes (see (Kschischang et al., 2001) for examples). In case of BNs or MRFs however, there are instances of graphs of either type which cannot be translated into the corresponding other one, without the introduction of additional edges (Koller & Friedman, 2009). These edges may not only lead to higher dimensions of the factor functions, which are more difficult to represent neurally, moreover, they may render a tree-shaped graph loopy. Also, the specific version of BP in MRFs is constraint to the representationally restricted case of maximal cliques size 2 (pairwise MRFs), which, in the FFG formulation, corresponds to the case of factor nodes of maximum branching degree 2. In higher-order MRFs however, the BP formulation of FGs/FFGs has to be used anyway. Although in BNs there is no such order restriction for BP, the message updates analogous to 2 and 5 are considerably more complex in this case (Kschischang et al., 2001).

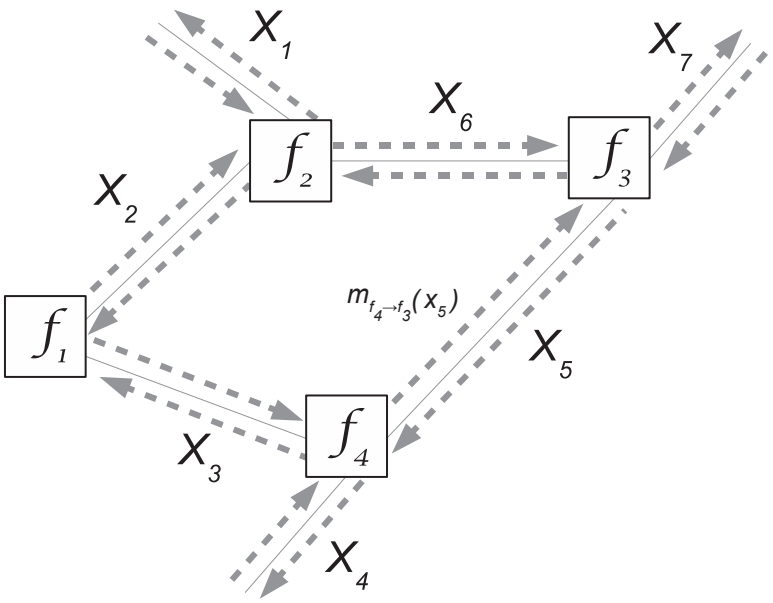

Figure 1: Forney Factor Graphs and Belief-Propagation
Example illustration of BP in a Forney Factor Graph(Löliger, 2004)(nodes represent factors $f_1, ..., f_4$ and edges the variables $X_1, ...X_7$). Each factor f_i gives rise to a nonnegative function, that only depends on the variables f_i is attached to. The product
$P(x_1, ...x_7) \propto f_1(x_2, x_3) \cdot f_2(x_1, x_2, x_6) \cdot f_3(x_5, x_6, x_7) \cdot f_4(x_3, x_4, x_5)$
then provides a factorized representation of the joint probability distribution $P(x_1, ...x_7)$. The BP messages exchanged between neighboring nodes are indicated by arrows along the edge (variable) connecting the two nodes. These messages are functions of the variable they are passed along, e.g. $m_{f_4 \to f_3}(x_5)$.

2 Results

2.1 Abstract Spike-Based Model of Belief-Propagation

In this section, we firstly present the abstract, mathematical description of a spike based processor, whose output spike statistics approximate the SPR. Secondly we show results of direct, literal implementations of that processor in software, disregarding the use of neural elements for the associated computations. We will refer to this version as the black-box implementation of the processor. For the sake of simplicity of exposition, we present the processor description for output message computations of a factor node attached to three variable edges. However the proposed methods can be straightforwardly generalized to factors attached to an arbitrary number of variables.
In the case of a 3-way factor node f attached to variables X, Y, Z (see figure 2b,left), equa-

tions 2-4 for an output message along Z become:

$$uSPR(z) := \int_D \int_D f(x,y,z) \cdot m_X(x) \cdot m_Y(y) \,\mathrm{d}x \,\mathrm{d}y \tag{6}$$

$$N := \int_D uSPR(z) \,\mathrm{d}z \tag{7}$$

$$SPR(z) := \frac{uSPR(z)}{N} \tag{8}$$

where we have replaced $um(z)$ and $m(z)$ by $uSPR(z)$ and $SPR(z)$ respectively for reasons which will become clear below.

2.1.1 Description of an Abstract Fading Memory Processor Computing the Sum Product Rule in a Spike-Based Way

At its core, the proposed processor is based on quantities that approximate scaled versions of $uSPR(z)$ and N in equations 6&7 respectively. Hence we first show how these quantities can be obtained by a simple Monte Carlo method. They are subsequently used to manipulate the firing statistics of an abstract spike-generation mechanism, which conveys $SPR(z)$ to a neighboring factor node.
In the proposed model, BP messages are represented by distributions of ISIs. Hence, the model employs a dedicated spike-generator whose firing statistics are controlled in a manner such that its ISI probability density function follows $SPR(z)$. Therefore, without loss of generality, we here assume all variables in the FFG to have a positive domain $D \subseteq \mathbb{R}^+$, such that ISI distributions can correspond to BP messages. However the presented approach is amendable to the case where the FFG variables are defined on the (potentially negative) codomain C of some invertible function $g : D \to C;\ ISI \to g(ISI)$ (in section 2.3.1 this is laid out in more detail for the concrete example $g(ISI) = \log(ISI)$).

We denote by letter m the spike-based BP messages produced by the processor and the corresponding mathematically exact normalized and unnormalized messages by SPR and $uSPR$ respectively (which fulfill equations 8&6 respectively). The value $m_X(x)$ message m_X assigns to value x of the analog variable X along which m_X is passed is therefore given by the spike-generators probability density of firing two successive spikes with a temporal distance of x. This may be regarded as a firing of labelled spikes, where the analog label attached to each spike is given by the spikes preceding ISI (see figure 2a). In other words, *spikes are interpreted as random samples from their underlying ISI distribution.* This sample-based representation of BP messages offers the possibility of automatically computing those messages according to equation 8, by means of a simple Monte-Carlo method (see figure 2b, right):
Starting from some position (x_0, y_0) on the (X, Y)-plane, each input spike from either $m_X(x)$ or $m_Y(y)$ updates that position according to the label it carries. For example, if the processor receives a spike from $m_X(x)$ labelled by ISI value '3', then the position is updated from

(x_0, y_0) to $(3, y_0)$. Updates along the Y-axis are done in the same manner by spikes from $m_Y(y)$. Assuming $Z = z$ to be fixed for the moment, at each such sampled position (x, y) the factor function $f_z(x, y) := f(x, y, z)$ is evaluated. Therefore one can associate a second kind of label with each spike, which is given by the result of such a function evaluation. In the following, this second label will be referred to as a 'function spike'. Evaluating all function spikes happening within a temporal sliding window $[t - W, t]$ –at any time t during the message passing dynamics– and taking the sum across all of them yields a stochastic process $S(z, t)$ approximately proportional to $uSPR(z)$ in equation 6 (see Materials and Methods section 4.1 for proof). Replacement of f_z with the function $F(x, y) := \int_D f(x, y, z)\, dz$ leads to a third spike label termed an 'integral function spike'. Summing all integral function spikes within W as before, one obtains a second stochastic process $S_F(t)$ approximately proportional to N in equation 7 (section 4.1). $S(z, t)$ and $S_F(t)$ are computed in parallel. Because of the finite window size W both quantities have fading memory. With them all quantities necessary for computing $SPR(z)$ in equation 8 are in place.

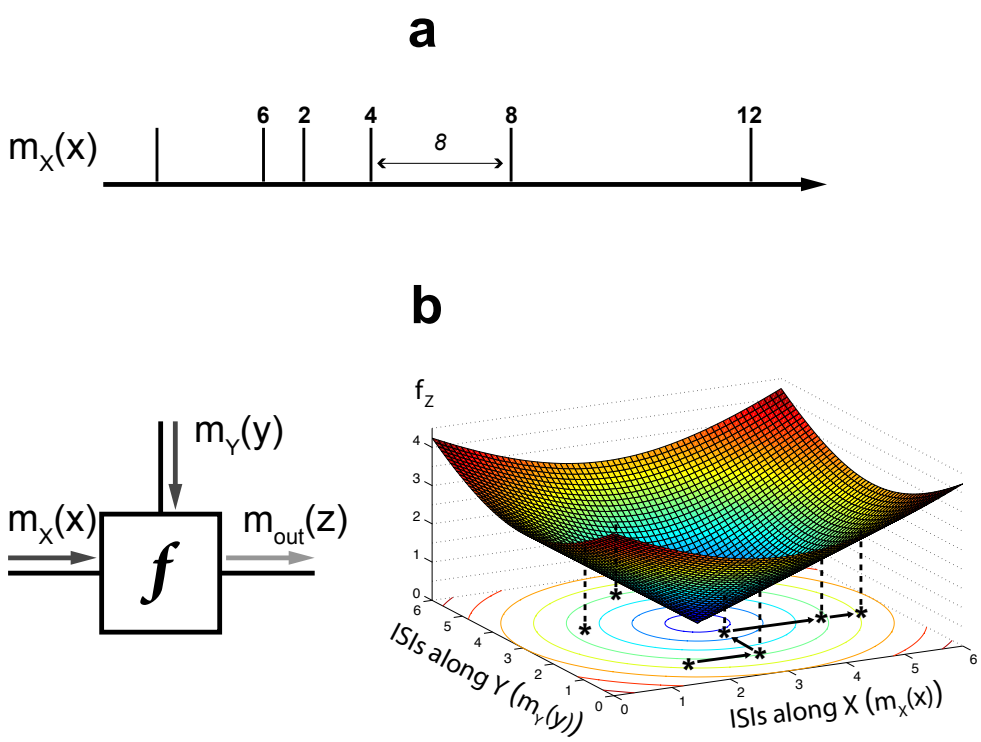

Figure 2: Basic principles of the proposed model
a) Spike encoding of BP messages: Each spike of a train representing a BP message is provided with an analog label, whose value corresponds to the length of the ISI preceding the spike, i.e. to the difference in spike times between the considered spike and its predecessor (see numbers above the spikes in arbitrary units). These analog values are therefore samples of the ISI distribution $p(x)$ underlying the spike train. A BP message $m_X(x)$ along variable X is then defined by $m_X(x) := p(x)$
b,left) 3-way factor node connected to analog variables X, Y, Z and defined by some function $f(x, y, z)$. For all z, output message $m_{out}(z)$ is computed out of input messages $m_X(x)$ and $m_Y(y)$.
b,right) Schematic illustration of the sampling process needed by the abstract processor computing $m_{out}(z)$ on the left by using spike-encoded inputs $m_X(x)$ and $m_Y(y)$ as in a. The shown function $f_z(x, y)$ is given, for fixed z, by $f_z(x, y) := f(x, y, z)$. See text for details.

Technically (but not algorithmically) the proposed sampling scheme bears some similarities to Gibbs Sampling (Geman & Geman, 1984; MacKay, 2003). Its mathematical structure renders the SPR excellently suited for being computed by such a simple Monte Carlo method, which only involves function evaluations of f_z and F. This elegantly avoids all of the issues involved in trying to efficiently implement the tricky SPR components: Products, sums containing many summands or integrals. When analog variables, and hence integrals, are used, in general it might be even impossible to directly compute the SPR. By employing Monte Carlo methods however, the required computation happens automatically, in a similar way

as the dynamics of systems in nature might be considered to be the result of complex computations on natures' mathematically formulated laws.
We show now how $S(z,t)$ and $S_F(t)$, the summed function and integral function spikes respectively, can be used to construct a stochastic spike-generator whose ISI distribution $m_{out}(z)$ is given by $m_{out}(z) \approx SPR(z)$. This step is important for a complete description of the abstract processor, because in general its output $m_{out}(z)$ is used as spike input for a subsequent processor stage formed by a neighboring factor node. The spike-generator constitutes a nonstationary renewal system, i.e. at any time t subsequent spikes are fired randomly with a stochastic intensity, called hazard (Cox & Lewis, 1966; Gerstner & Kistler, 2002), that only depends on t and $\Delta t := t - t_0$, the time elapsed since the time of the last spike t_0. Using $S(z,t)$ and $S_F(t)$ the hazard needs to be computed appropriately, such that the ISI distribution of the spike-generator is guaranteed to follow equation 8. It is because of the finite size W of the sliding summation window that S and S_F depend on t and hence the renewal system is rendered nonstationary. A standard result of renewal theory in the nonstationary case relates the hazard $h(t)$ to the ISI distribution $p(\Delta t, t)$ (Cox & Lewis, 1966; Gerstner & Kistler, 2002):

$$h(t) = \frac{p(t - t_0, t)}{1 - \int_{t_0}^{t} p(t' - t_0, t')\, dt'} \tag{9}$$

If we associate value z of variable Z with Δt, i.e. set $z = \Delta t$, and want to have $p(\Delta t, t)$ to follow the SPR, we have to set

$$p(\Delta t, t) := \frac{S(\Delta t, , t)}{S_F(t)} \approx \frac{r_{tot} \cdot W \cdot uSPR(\Delta t)}{r_{tot} \cdot W \cdot N} = SPR(\Delta t) \tag{10}$$

where r_{tot} is a constant explained in the Materials and Methods section 4.1 together with the approximation and why the latter steadily improves with increasing W. Plugging 10 into 9 yields a closed system for computing spiking BP-messages (see figure 3 for a block diagram of the complete abstract processor).

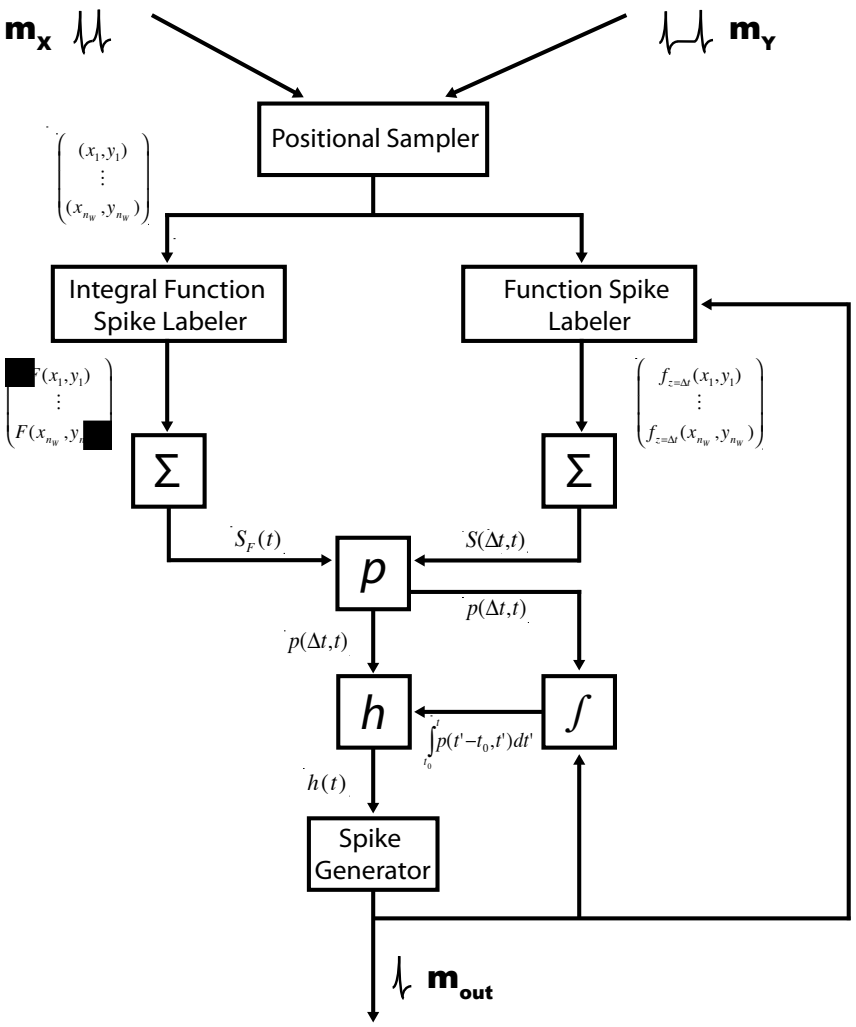

Figure 3: Block diagram of the abstract spiking BP processor

The following operations are executed at any time t during execution of the processor:

1.) Positional Sampler: By means of a sliding summation window of length W, the positional sampler keeps track of a list of $n_W(t)$ samples on the (X,Y)-plane. These samples are caused by input spikes from m_X and m_Y in $[t-W,t]$ and form the basis for computing both, function and integral function spikes.

2.) Integral Function Spike Labeler: The integral function spike labels only depend on X and Y. Hence they are computed only once, as soon as their respective causing (X,Y)-sample comes in.

3.) Function Spike Labeler: In addition to the (X,Y)-samples, the function spike labels also depend on Z. If the spike-generator fires at time t_0 (see (6)), the function spike labeler is 'reset', i.e. the function spikes are labeled according to $f_{z=0}$. Measuring $\Delta t = t - t_0$, the time passed since the last fired spike, these labels continuously change their values according to $f_{z=\Delta t}$.

4.) Σ: The quantities $S(\Delta t,t)$ and $S_F(t)$ are computed by summing all (integral) function spikes in W respectively, leading to

5.) ISI probability density p and cumulative distribution $\int_{t_0}^{t} p(t'-t_0,t')\mathrm{d}t'$: Using $S(\Delta t,t)$ and $S_F(t)$, $p(\Delta t,t)$ is computed from equation 10 and accumulated by the integrator node ($\int$). This integrator must be set to zero after each spike of the spike-generator.

6.) Hazard h: Both $p(\Delta t,t)$ and $\int_{t_0}^{t} p(t')\mathrm{d}t'$ are substituted into eq.9 to update $h(t)$.

7.) Spike Generator: Output spikes are drawn randomly at each time t according to $h(t)$. Because of equation 9 it is assured that the resulting ISI distribution indeed satisfies the SPR (equation 10). To reset the function spike labeler and integrator, feedback of the fired spike is enabled.

2.1.2 Performance Analysis of Black-Box Implementations of the Abstract Processor

In this section we show the performance of direct software implementations of the processing steps described in figure 3, disregarding the use of neural elements. For that, we first evaluated the computational results of a single, isolated factor node attached to three variable edges. Each of the corresponding three output messages was computed by a separate, independent processor, in response to three different, arbitrary ISI distributions representing the input messages. The approximation quality of these messages with respect to results obtained by ordinary, discrete BP has been determined. The processors were necessarily advanced in discrete time steps during each of which spikes were fired randomly according to the actual hazard (see section 2.1.1). All times were measured as multiples of time steps. The input ISI distributions contained probability mass only within the $(0, 100]$ step interval. W was set to 750 steps.

The factor function $f(x, y, z) \approx f_{=}(x, y, z) := \delta(x - y) \cdot \delta(y - z)$ represented an equality constraint factor as described in (Löliger, 2004) –an important type of factor function that provides FFGs with the full representational power of ordinary FGs (see section 1.1). On the other hand, the factor's neurobiological relevance stems from the fact that it can also be used to perform statistically optimal cue combination, since its output messages correspond to the product of its input messages (see below). Interestingly, psychophysical experiments have revealed human subjects to combine sensory information (e.g. visual and haptic cues about the height of an object) in a way that is indeed very close to the statistically optimal estimate (Ernst & Banks, 2002).

Because the support of $f_{=}$ is infinitesimally 'slim' around the line $x = y = z$ (and hence cannot be hit by a sample with nonzero probability), for f we have increased its 'thickness' in our simulations, using a Gaussian whose first principal component was set parallel to that line with infinite standard deviation. The standard deviations of the second and third principal component were set to 2 steps each. Figure 4a shows f together with all involved messages. Application of the SPR to f leads to $SPR(x) \approx m_{in,Y}(x) \cdot m_{in,Z}(x)$, $SPR(y) \approx m_{in,X}(y) \cdot m_{in,Z}(y)$, $SPR(z) \approx m_{in,X}(z) \cdot m_{in,Y}(z)$. These quantities were computed as control by ordinary, discrete BP (see section 1.1), i.e. as a reference for determining the approximation quality. For that, f was discretized and stored in a 3-dimensional tensor of size $100 \times 100 \times 100$, representing all combinations of the first 100 values (time steps) of variables X, Y, Z. Note that this version of BP is very costly: For each of the 100 values of $SPR(z), z \in (0, 100]$ a sum over 100×100 products has to be computed. Additionally, a tensor having 10^6 entries has to be stored, whereas in our model just two function descriptions for computing the (integral) function spikes are needed. Figure 4b shows the ideal ISI distributions $SPR(\cdot)$ and their approximating $m_{out,(\cdot)}$. A close match between these two quantities can be observed.

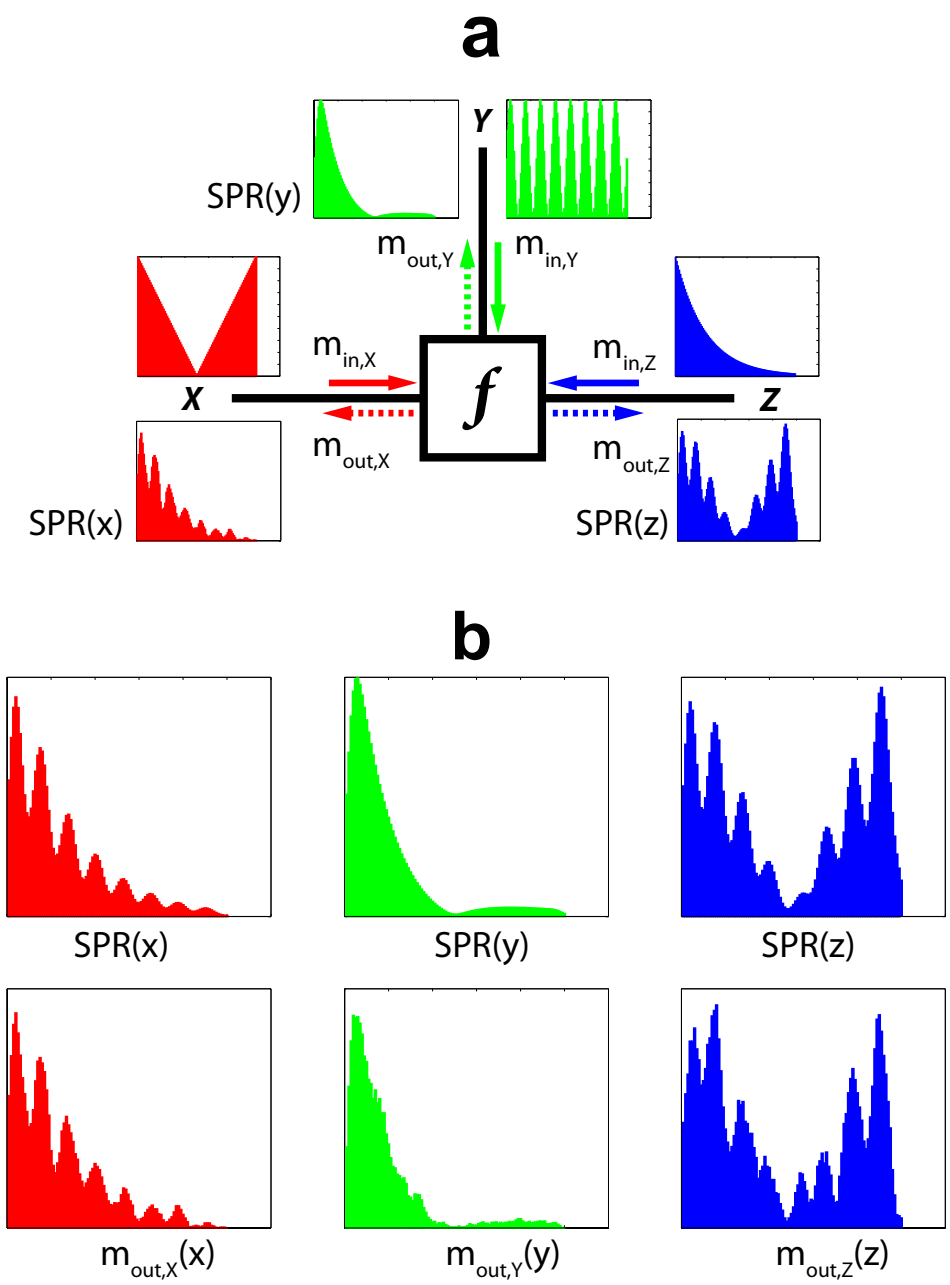

Figure 4: Performance of computing output messages of a single factor node
a) The approximate equality constraint factor $f(x, y, z) \approx f_{=}(x, y, z) := \delta(x - y) \cdot \delta(y - z)$ depending on variables X, Y, Z. The ideal ISI histograms of the input messages $m_{in,X}(x) \propto |x - 50|, m_{in,Y}(y) \propto \sin\left(2\pi \frac{8}{100} \cdot y\right) + 1, m_{in,Z}(z) \propto e^{-\frac{z}{25}}$ are shown together with those of the corresponding ideal output messages $SPR(x) \approx m_{in,Y}(x) \cdot m_{in,Z}(x)$, $SPR(y) \approx m_{in,X}(y) \cdot m_{in,Z}(y)$, $SPR(z) \approx m_{in,X}(z) \cdot m_{in,Y}(z)$, as computed by brute-force, discrete BP. The actual output messages $m_{out,X}, m_{out,Y}, m_{out,Z}$ are computed by three processors following the dynamics described in section 2.1.1.
b) Top row: Magnified ideal output messages $SPR(\cdot)$ as in a.
Bottom row: Actual ISI histograms of the three processors computing the output messages $m_{out,(\cdot)}$. For each such histogram 3000 spike samples have been collected.

We next considered how well a whole FFG based on black-box processors can perform inference, i.e. can compute the beliefs given by equation 5. For that we investigated the two tree structures shown in figures 5a and 6a. In both cases, 30 trials with random instantiations of each involved factor function were conducted, i.e. during any such trial each factor function $f(\boldsymbol{x}) = \sum_i w_i \mathcal{G}_i(\boldsymbol{x} \,|\, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ was given by an unnormalized mixture of Gaussians whose mixture components $\mathcal{G}_i$ had random weights w_i, mean vectors $\boldsymbol{\mu}_i$ and covariance matrices $\boldsymbol{\Sigma}_i$(see Materials and Methods section 4.4 for parameters). Therefore, each trial represented an arbitrary inference problem modeled as FFG. Variables of the FFG could take on integer step values in $(0, 130]$. After having collected 3000 spike-samples for each message in each of the 30 inference runs, we computed the beliefs $B(x_{i_j})$ for some variables x_{i_j} using the message ISI histograms and equation 5 (with the integral replaced by a sum across discrete time steps). Again, as a control, the true marginals $P(x_{i_j})$ were computed by ordinary, discrete BP. Note that in figure 6 we had to coarsen the time resolution from 1 to 4 steps to keep the task tractable for the ordinary BP method. Otherwise tensors of size $130^4 \approx 2.86 \cdot 10^8$ would have had to be stored and 130 sums over 130^3 products processed. In contrast, the black-box processors were still run at a resolution of 1 step, but to keep comparability their ISIs were also binned using the coarser time grid.
To determine the approximation quality, the ordinary($D(B\|P)$) and a normalized$\left(D_{norm}(B\|P) := \frac{D(B\|P)}{H(B,P)}\right)$ Kullback-Leibler (KL) divergence between $B(x_{i_j})$ and $P(x_{i_j})$ were calculated -$H(B, P)$ denotes the cross-entropy $H(B, P) := \sum_{x_{i_j}} B(x_{i_j}) \cdot log(P(x_{i_j}))$ between B and P. Hence $D_{norm}(B\|P)$ is the fraction of additional (wasted) bits used by a code of length $H(B, P)$ constructed from $P(x)$, compared to the minimum code length $H(B)$. Figures 5b,c and 6b,c show the corresponding results. It can be seen that the fraction of wasted bits is at most around 5% and below 1% in most cases, that is, typically very small. To evaluate chance level, in figures 5c and 6c we have compared the values of $D(B\|P)$ to their 3σ-range, when $B(x_{i_j})$ is replaced by a distribution $B^*(x_{i_j})$ that is drawn from a uniform distribution of distributions of x_{i_j}. The 3σ-range is given by those values of $D(B^*\|P)$ that are within three standard deviations $\sigma\left[D(B^*\|P)\right]$ around the expected value $\mathbb{E}\left[D(B^*\|P)\right]$ (see Materials and Methods section 4.2 for how $\mathbb{E}\left[D(B^*\|P)\right]$ and $\sigma\left[D(B^*\|P)\right]$ are calculated). It is apparent from these figures that the performance of $B(x_{i_j})$ as found by the processors is far better than the average performance of randomly drawn distributions $B^*(x_{i_j})$. Due to the coarser time resolution and the corresponding smaller number of states n per variables (see section 4.2), the 3σ-ranges in figure 6c are somewhat larger than in figure 5c. In some cases these ranges almost include $D(B^*\|P) = 0$, which leads to long error bars in log-plots.

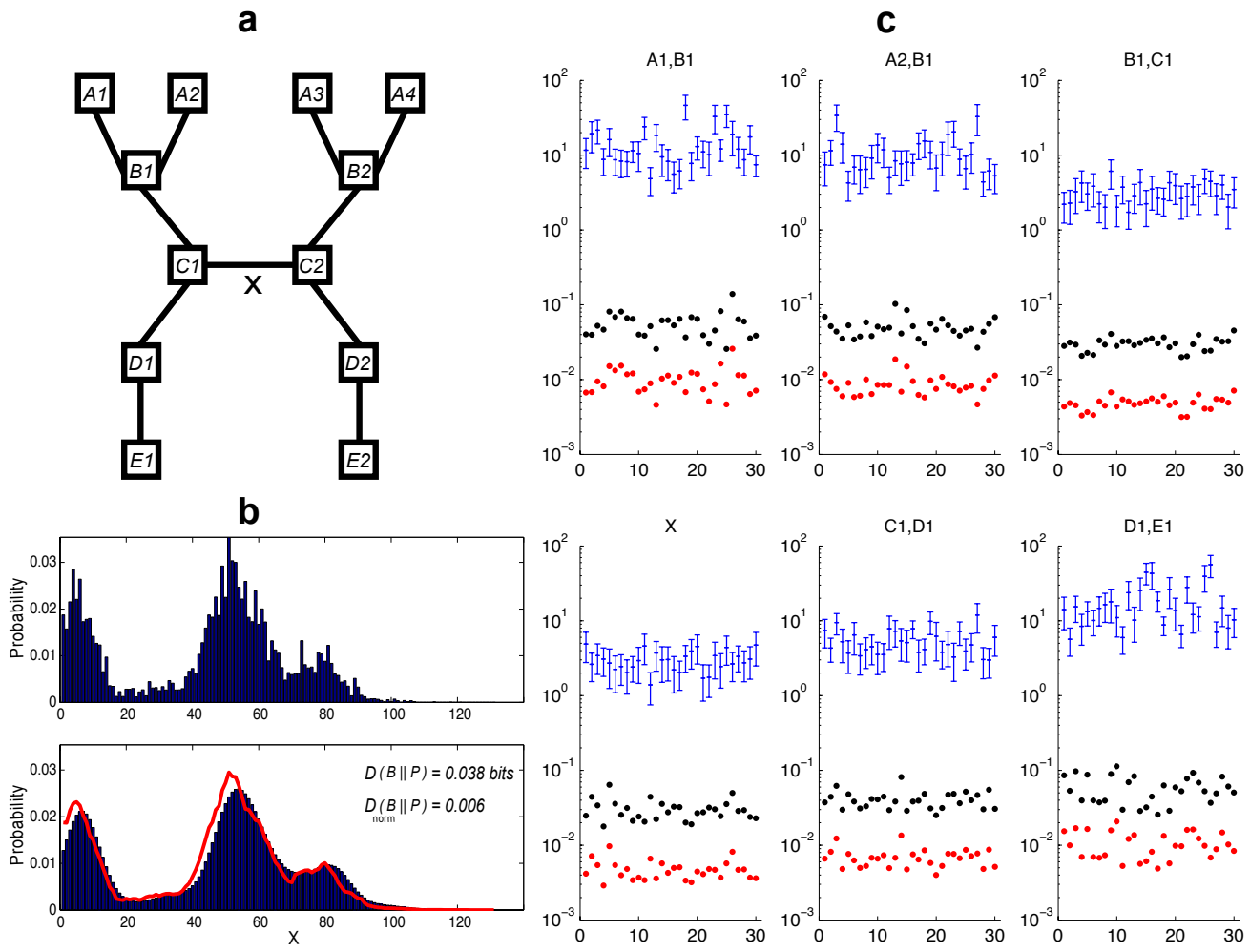

Figure 5: Computing marginals in a randomly parameterized FFG I

a) FFG tree structure used as starting point for 30 inference trials with factor functions A1-E2 randomly defined by gaussian mixtures (see text for details). The maximum branching degree of a factor node is 3.

b) Top: Raw data of the belief of variable X connecting factor nodes $C1$ and $C2$ for trial number 2 as found by the black-box implementation of the abstract processor.

Bottom: Exact marginal distribution $P(x)$ of variable X (blue bars). This is the result of ordinary, discrete BP. The red curve shows the moving averaged version $B(x)$ of the belief in the top row, averaging window size was 5 points.

c) x-axis: Trial number, y-axis: Values of $D(B\|P)$ (black dots, in bits) and $D_{norm}(B\|P)$ (red dots) for representative variables. Blue bars indicate the 3σ-range of $D(B^*\|P)$ if instead of $B(x_{i_j})$ a random distribution $B^*(x_{i_j})$ is taken (see Results and Materials and Methods for details).

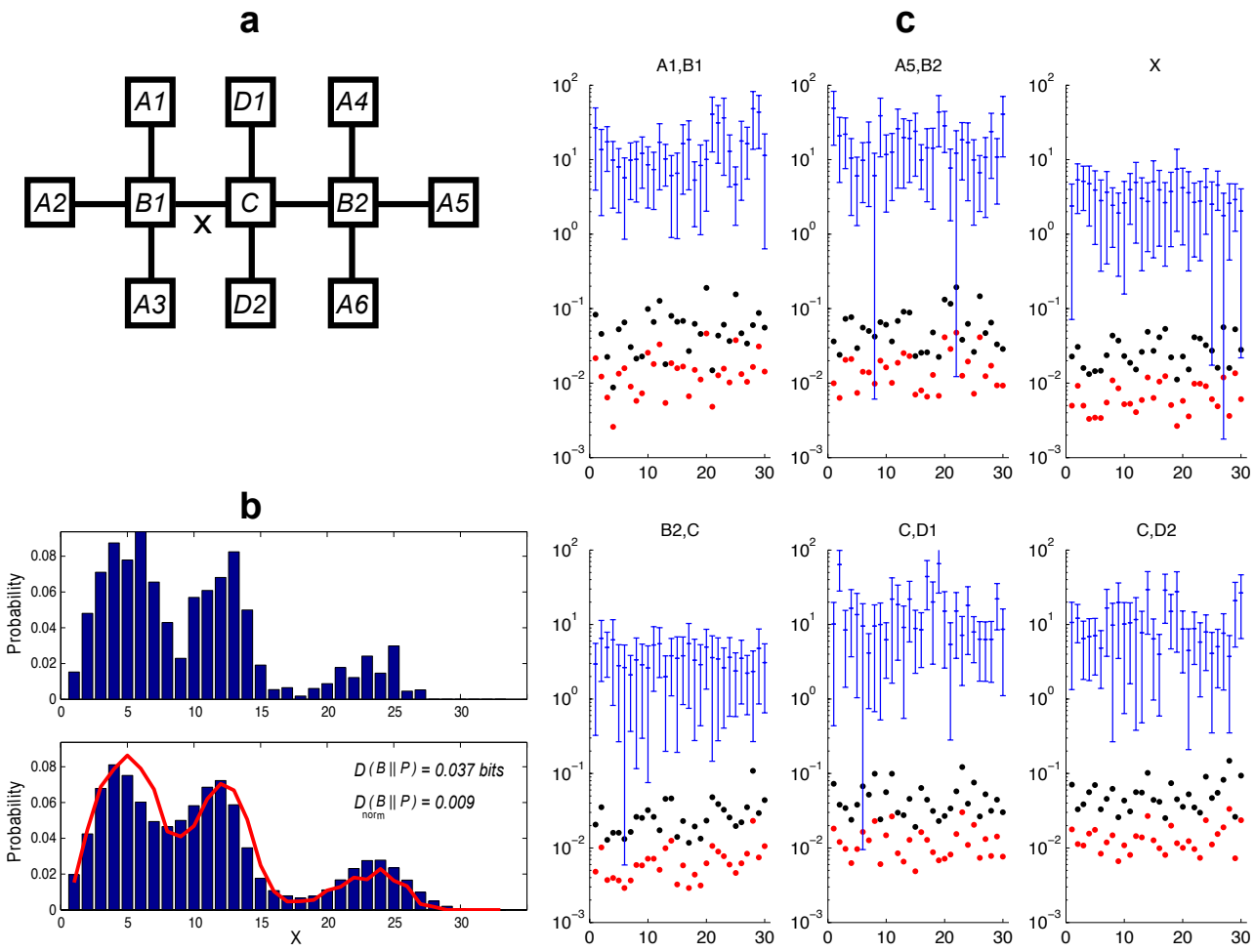

Figure 6: Computing marginals in a randomly parameterized FFG II
a) FFG tree structure used as starting point for 30 inference trials with factor functions A1-D2 randomly defined by gaussian mixtures (see text for details). The maximum branching degree of a factor node is 4.
b) Top: Raw data of the belief of X as found by the black-box processing method.
Bottom: Exact marginal distribution $P(x)$ of variable X connecting factor nodes $B1$ and C for trial number 9 (blue bars). This is the result of ordinary, discrete BP. The red curve shows the moving averaged version $B(x)$ of the belief in the top row, averaging window size was 3 points.
c) x-axis: trial number, y-axis: Values of $D(B\|P)$ (black dots, in bits) and $D_{norm}(B\|P)$ (red dots) for representative variables in each trial. Blue bars indicate the 3σ-range of $D(B^*\|P)$ if instead of $B(x_{i_j})$ a random distribution $B^*(x_{i_j})$ is taken (see Results and Materials and Methods for details).

2.2 Concrete Spike-Based Model of Belief-Propagation Using Neural Elements

Although the abstract, functional description of the BP processor presented in section 2.1.1 is spike-based, it is not obvious, (a) what principal *computational* system/learning machine using which learning algorithm can be used to emulate that processor, and (b) what biophysical, neural mechanisms could possibly be exploited to concretely implement that machinery in *neural* systems. Here we provide a general answer to (a), which is still flexible enough to be consistent with at least two different answers to question (b), as discussed in detail in the discussion (section 3.1).
In the spirit of the previous section we then investigate the performance of such an emulation for the computation of a single output message. Consequently, by simulating a Hidden Markov Model-like chain topology of factor nodes, it is also verified that the proposed model enables inference in larger graphs. In general, the results presented in this chapter are intended to show that our abstract BP processor is not just a mere mathematical abstraction, rather it can be emulated by a concrete computational system. At this point however, we have to emphasize again that the former is by no means confined to an emulation by the latter.

2.2.1 General Neural Architecture Implementing the Abstract BP processor

Figure 7 shows a general computational system answering question (a). It is conceptually divided into two parts. In the first part the incoming spike trains representing the BP-messages (m_X, m_Y) are decoded to time-series of analog ISI values. Using these latter sequences, the second part then performs the actual computations needed by the abstract BP processor (section 2.1.1, figure 3). Building on our previous results in (Steimer et al., 2009), both parts are based on Liquid/Echo-State-Machines (ESNs/LSMs) (Maass, Natschläger, & Markram, 2002; Jäger, 2001). This framework (which is also called 'reservoir computing') utilizes recurrent networks of spiking neurons/sigmoidal neurons respectively ('Liquid pools', illustrated as grey 'L' circles in figure 7), which provide input to distinct 'readout' neurons (indicated by Σ). Only those synaptic weights which correspond to connections from the liquid to the readout neurons are altered during a (usually supervised) training procedure and, depending on the task, the readouts may or may not feed back to the liquid pools. In the absence of such feedback, ESNs/LSMs have been proven capable of approximating any time-invariant filter with fading memory (Maass et al., 2002). Employing feedback however allows these systems to drastically prolong their memorized time span, even in the presence of noise (Maass, Joshi, & Sontag, 2007). In fact, nonfading memory is potentially needed by the ISI decoder readouts (R_X, R_Y), whose task is to put out the analog ISI label associated with the last received spike and to hold that value until the next spike arrives.
For the simulations presented in the next section, we have used sigmoidal neurons for the readout units R_X, R_Y and the liquid pools L_X, L_Y and L_{out}, so technically we are positioned in the framework of Echo-State machines. The readout R_{out} however must necessarily be based on some spiking neuron model, since its output must represent the ISI-endcoded SPR.

Therefore, we have used a leaky-integrate-and-fire neuron with escape noise (Gerstner & Kistler, 2002), whose membrane voltage V_m is given by the weighted sum of activations of the sigmoidal neurons in L_{out}. Subject to escape noise the neuron may fire spikes at any time t, with a firing hazard $h(t)$ that depends exponentially on $V_m(t)$: $h(t) = h(V_m(t)) = h_\Theta e^{\frac{V_m(t)-V_\theta}{\sigma_h}}$ (with h_Θ denoting the hazard at the (soft) threshold value V_θ and σ_h determining the 'softness' of the threshold). This type of hazard activation function for spike firing is often used in the neuroscientific literature (Pfister, Toyoizumi, Barber, & Gerstner, 2006; Pfister, Dayan, & Lengyel, 2010; Rao, 2005; Eggert & van Hemmen, 2001).
In our model, a thus fired spike is fed back into the recurrent network L_{out}. According to the processor description of section 2.1.1 this feedback is necessary to signal the circuit when the function spike labels have to be reset. Spike input to the sigmoidal neurons of L_X, L_Y and L_{out} is provided by low-pass filtering the spikes to mimic postsynaptic potentials. Empirically we have found the approach not to work when the ISI decoder readouts are removed and the low-pass filtered spikes of m_X and m_Y are fed directly into L_{out}.
Training of the weights of R_X and R_Y has been done by ordinary linear regression, i.e. by computing the pseudoinverse. For R_{out} however, we have minimized the summed KL divergences between target and actual ISI distributions using the BFGS quasi-Newton method (Bertsekas, 1999). Alternatively, it is also possible to make use of a Spike Time Dependent Plasticity (STDP)-like learning scheme (Pfister et al., 2006), which amounts to a stochastic gradient descent procedure (Bottou, 1998) applied to the negative log-likelihood of observed spike samples. Interestingly, under rather loose and biologically plausible conditions imposed on $h(V_m)$, this second cost-function is convex (Paninski, 2004). See Materials and Methods section 4.3 for a detailed account of both costs.

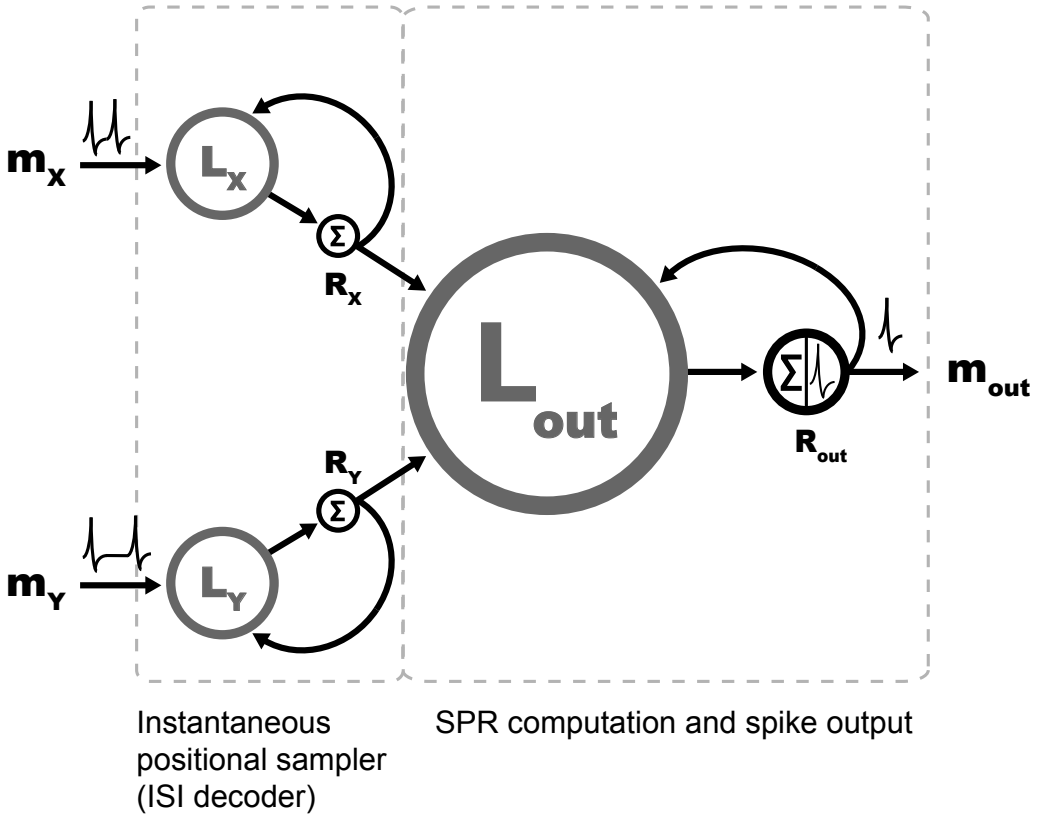

Figure 7: Computational system for a concrete implementation of the abstract BP processor (see main text for details).

2.2.2 Performance Analysis of the Neural BP processor

In the spirit of section 2.1.2 we have conducted a series of simulations evaluating the performance of an ESN based processor as shown in figure 7. For that the ESN software toolbox of Jäger et al. was used (publicly available at `http://www.reservoir-computing.org/`) and customized where needed. Settings of the various reservoir parameters are listed in the appendix (section 4.5).

The processor has been trained to approximate a BP message emitted by a factor node f_1 that consisted of an unnormalized gaussian mixture (four components with equal weight 0.5; random mean vectors whose components were drawn independently and uniformly from $[15, 60]$; circular covariance matrices, $\sigma = 7$). After training, the ESN network was stimulated by a series of spike input messages which were represented by various different ISI distributions. These distributions were also created by random gaussian mixtures, having 1 to 3 mixture components (figure 8). Since only a single, isolated factor node was simulated we could not compute the target belief of the output variable according to equation 5. There-

fore, the normalized output message m_{out} based on equation 4 played the corresponding role. As in section 2.1.2 the approximation quality is apparent.

We have also verified whether a number of serially connected ESN processor stages can compute the correct, final BP output message at the end of the series. This test was mandatory to see if the system of figure 7 allows for a modular composition of larger FFGs, by enabling the desired communication between adjacent factor nodes. Therefore, we have first trained a second ESN processor to approximate an output message of some factor f_2, and then used the output of the f_1 ESN-stage as input to the subsequent f_2 ESN-stage. This way, we have created an alternating sequence of ten f_1/f_2-stages reminiscent of a Markov Chain with external input (figure 9a). To verify proper intercommunication even between factor nodes belonging to different families of functions, factor f_2 was parameterized in a slightly more general way than f_1. For f_2 also the covariance matrices of its four Gaussians were randomized, by drawing random principal component directions whose associated standard deviations were independently and uniformly taken from $[3, 10]$.

The result of such serial assembly of factors can be seen in figure 9b: The whole chain was simulated for ten independent trials, with the external input messages given by single, random gaussians. Although after ESN training the output messages of all stages are subject to unavoidable errors, there are no visible signs of error propagation for later stages of the chain (see also 9d). Average performance stays constant and is comparable to the single-ESN case. The sole exception is stage 8, where a sudden decrease in performance occurs for trial no. 2, 5 and 8. We have carefully verified that this decrease is indeed coincidental and not caused by some artifact. In trial no. 8 we can even see an instance of performance recovery, as performance at stage 8 is worse than chance level, only to regain baseline at stage 10 (figure 9c).

The results indicate that the ESN-based BP processor allows for an assembly of single factor nodes into larger FFGs. However, we do not know to what extent these results depend on the specific identity of the factor functions f_1/f_2, combined with random gaussian input messages. It cannot be ruled out that there exist combinations of factor functions, graph topologies and input messages for which error propagation takes place. In principle, for a given such combination, the methods of (Ihler, Fisher, & Willsky, 2005) can be used to determine the error of the final, steady state messages. Unfortunately however these methods itself are based on a specific form of message-passing, hence, for a given graph the severity of error propagation is hard to determine in advance.

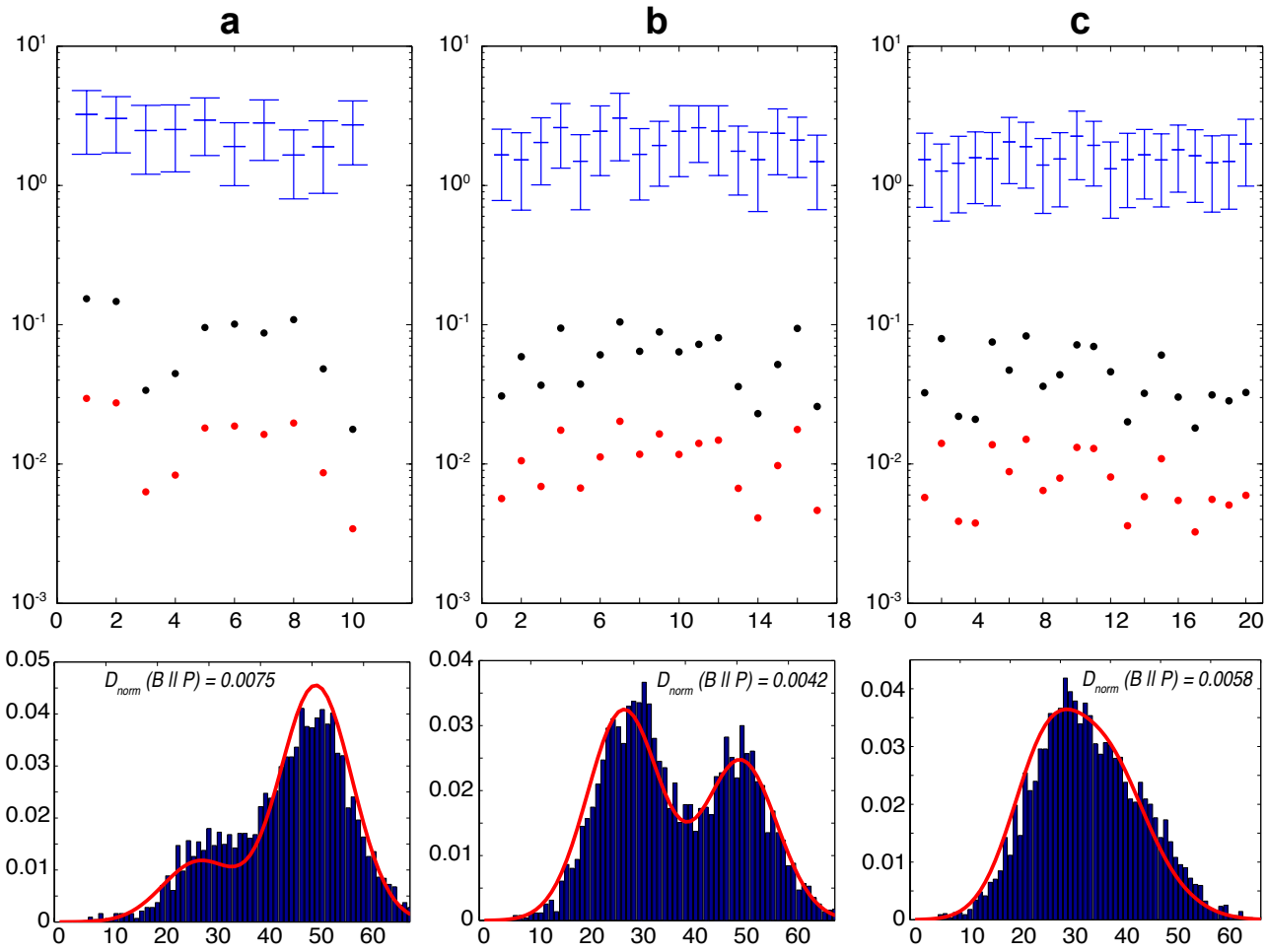

Figure 8: Performance results of a single ESN based BP processor as shown in figure 7
Top row: Approximation quality of the BP output message m_{out} when input messages m_X and m_Y were both created by random gaussian mixtures consisting of (a) 1, (b) 2 and (c) 3 mixture components. x-axis: Trial number, y-axis: Values of $D(m_{out} \,\|\, m_{\substack{out,\\target}})$ (black dots, in bits) and $D_{norm}(m_{out} \,\|\, m_{\substack{out,\\target}})$ (red dots). Blue bars indicate the expected value and 3 times the standard deviation of $D(m_{out} \,\|\, m_{\substack{out,\\target}})$ when m_{out} is replaced by a random distribution. In (a) and (b) all mixture components had equal weight, in (c) weight proportions were drawn randomly from $[0.2, 1]$. The mean of each component was chosen randomly from $[10, 50]$, the components' widths were a) $\sigma = 1.5$, b) $\sigma = 2.5$ and c) $\sigma = 1.5$
Bottom row: Example ISI histograms of m_{out} for individual trials in each of the three cases (blue bars). Red curves show $m_{\substack{out,\\target}}$.

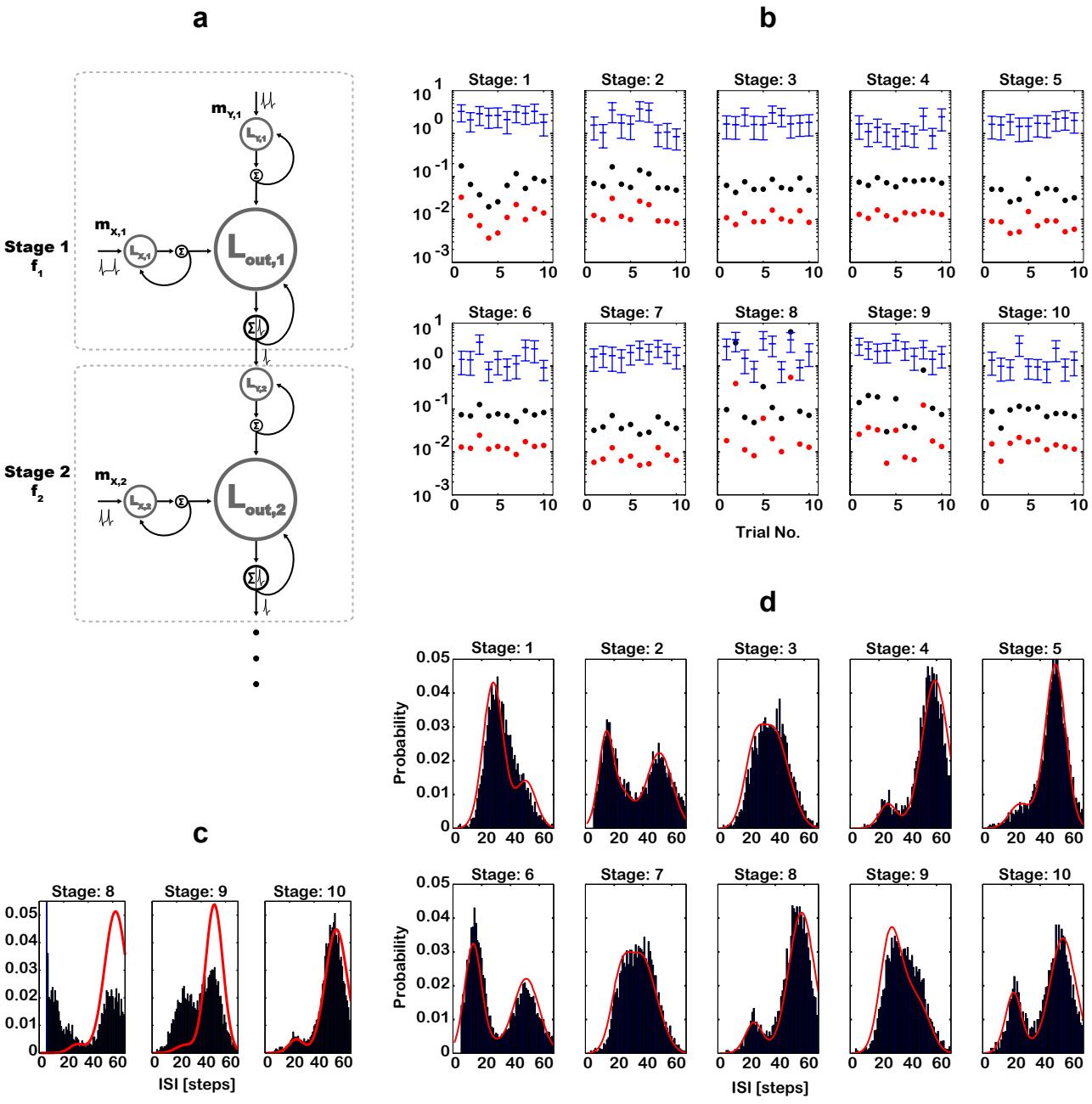

Figure 9: Performance results of serially connected ESN-based BP processors
a) Processor stages are connected in a serial, Markov Chain-like manner. The identity of the corresponding factor functions alternates between f_1 and f_2. External input to the (ten) stages is provided by the messages $m_{Y,1}, m_{X1,}, m_{X,2}, ..., m_{X,10}$.
b) Performance of the output messages of each of the ten stages during ten independent trials (see figure 8 for definitions). In each trial, external input messages were given by single gaussians of constant width $\sigma = 1.5$, with means taken uniformly from $[5, 50]$ (for f_1) and $[5, 40]$ (for f_2).
c) Target (red) and spike-based (blue) output messages of trial no. 8 for the last three stages.
d) Target (red) and spike-based (blue) output messages of trial no. 10 for all stages.

2.3 Interpretation of Experimental Spike-Data in the Light of Information Encoding with ISIs

In this section the ISI-based neural code used by our model is compared to spike data from various experiments. We first show how the general shapes of ISI distributions measured in cortex can be matched to the output of a slightly redefined version of our BP processor. Subsequently, spike data assessed in area LIP during a behavioral task is reinterpreted in the light of the model, leading to a concrete, quantitative prediction for ISI distributions during execution of this task.

2.3.1 Matching the Shape of Experimental ISI distributions

ISI distributions recorded from neural substrates typically show far less variability than those produced by the BP processor architecture of figure 3. For example, such highly structured distributions containing multiple prominent modes as in sections 2.1&2.2 cannot, to our knowledge, be observed in real spike trains. Indeed, a substantial number of studies have reported ISIs in the nervous system to belong to a rather restricted class of distributions, such as quasi-exponential, gamma and log-normal distributions (Noda & Adey, 1970; Burns & Webb, 1976; Softky & Koch, 1993; Bair, Koch, Newsome, & Britten, 1994; Compte et al., 2003). This however stands in contrast to studies which report log-ISI rather than ordinary ISI distributions (Reich et al., 2000; Shih et al., 2011). In this case, histograms are produced by dividing the log-time domain into bins of constant size, a procedure that makes apparent features of the distribution that are barely visible in linearly scaled histograms (Shih et al., 2011).

Our BP processor can accommodate for these findings by slightly changing the sampling process of figure 2. If positional updates are done based on the $\log(ISI)$-label of a spike rather than its ISI-label, the resulting sampling dynamics then correspond to a factor node defined over a set of transformed variables, each of which is the logarithm of one of the original variables. Additionally in equation 10, $p(\Delta t, t) \approx SPR(\Delta t)$ has to be replaced by $p(\Delta t, t) \approx \frac{SPR(\log(\Delta t))}{\Delta t}$ to end up with a spiking system that represents BP messages as $\log(ISI)$-distributions. Note however that the same type of analysis can be performed for any invertible ISI transformation function $g(\Delta t)$, not just the logarithm.

Figure 10 shows simulation results analogous to the single factor results of figure 4, when such a message encoding scheme was used. It is obvious from the figure that a remarkable degree of structure in the $\log(ISI)$-distributions is hidden behind some rather 'boring' looking ISI distributions (e.g. compare $m_{in,X^*}(x^*)$ to $m_{in,X}(x)$ in (b) and $m_{out,Z^*}(z^*)$ to $m_{out,Z}(z)$ in (c)). This is also a feature of some of the experimental data ((Reich et al., 2000), figures 1&2). Qualitatively, the resulting ISI distributions in 10 look quasi-exponential, by consistently displaying a prominent peak at small ISI values.

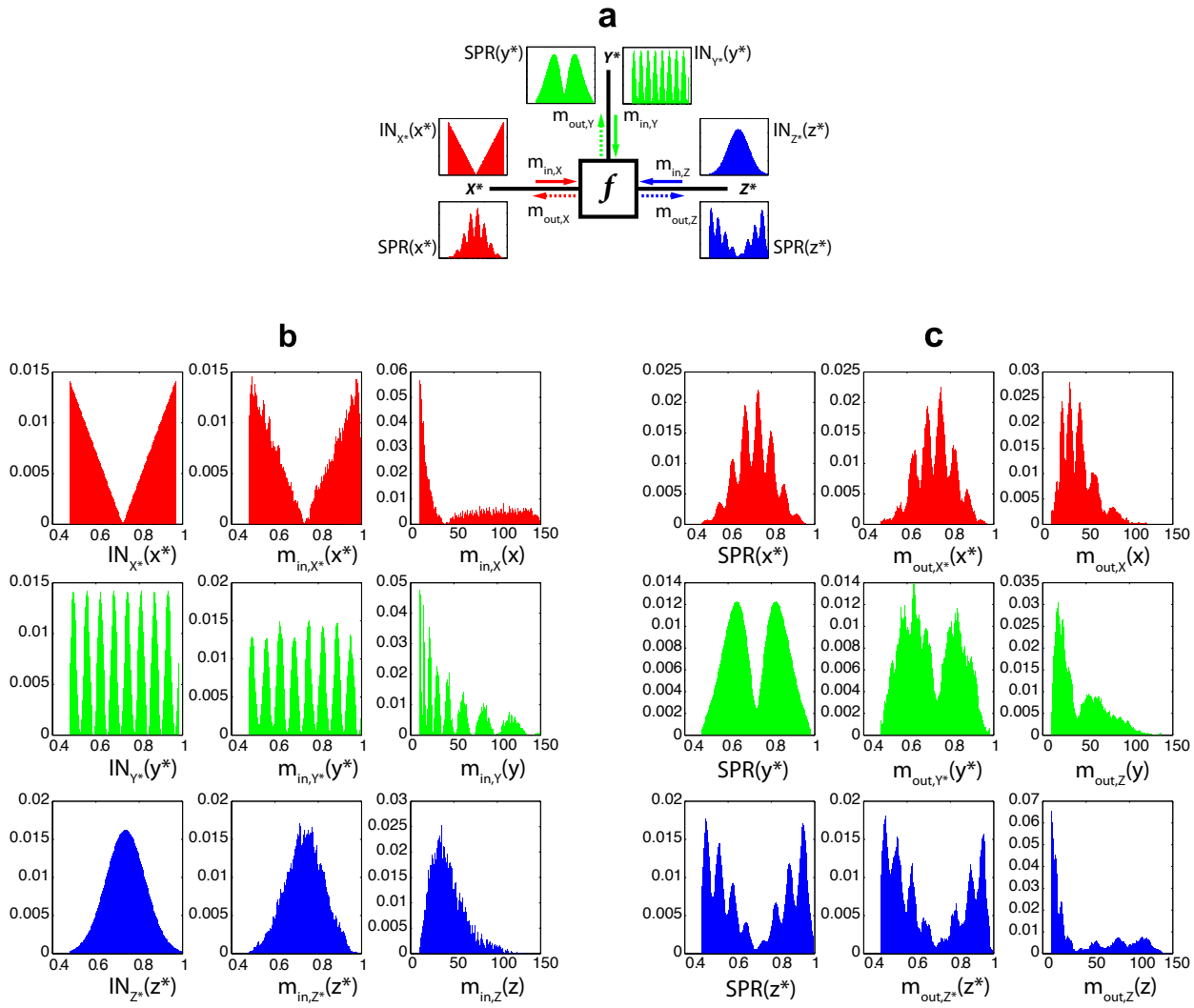

Figure 10: Performance of computing output messages of a single factor node with log(ISI) encoded messages.

a) The approximate equality constraint factor $f(x^*, y^*, z^*) \approx \delta(x^* - y^*)\delta(y^* - z^*)$ depending on the transformed variables $X^* := g(X), Y^* := g(Y), Z^* := g(Z)$, where $g(\Delta t) := \frac{\log(\Delta t)}{b}, b := \log(151)$ is the ISI transformation function used for spike labeling. The distribution of these labels is called the $\log(ISI)$ distribution. Shown are the ideal $\log(ISI)$ distributions of the input messages $IN_{X^*}(x^*) = \left|x^* - \frac{1}{2}(b-a)\right|, IN_{Y^*}(y^*) = \sin\left(2\pi\frac{8}{b-a} \cdot y^*\right)$ and $IN_{Z^*}(z^*) = \exp\left[-\left(\frac{z^* - \frac{1}{2}(b-a)}{\frac{1}{4}(b-a)}\right)^2\right]$, with $a := \log(10)$. The corresponding ideal output messages are $SPR(x^*) \approx IN_{Y^*}(x^*) \cdot IN_{Z^*}(x^*)$, $SPR(y^*) \approx IN_{X^*}(y^*) \cdot IN_{Z^*}(y^*)$, $SPR(z^*) \approx IN_{X^*}(z^*) \cdot IN_{Y^*}(z^*)$ and were computed by brute-force, discrete BP. As before, m denotes actual messages based on spike data.

b) Left column: Magnified histograms of the ideal input messages $IN_{(\cdot)}(\cdot)$ as in (a). Middle column: $\log(ISI)$ histograms of the actual input messages. Right column: ISI histograms of the actual input messages. The $\log(ISI)$ and ISI histograms are both based on the same input spike trains.

c) The same as in (b) for the corresponding output messages computed by abstract processors.

2.3.2 Reinterpreting Firing Rates in Area LIP in the Light of ISI Encoding

In this section we show how the firing rates measured in area LIP in rhesus monkeys during a behavioral experiment can be reconciled with our proposed ISI encoding scheme. From this then follows a reinterpretation of the supposed neural code that underlies these results, along with a quantitative prediction for ISI distributions in LIP during execution of the experiment.

(Yang & Shadlen, 2007) have performed a two-alternative forced-choice task, where different visual stimuli (shapes) shown to the monkey were associated with different evidence of reward administration. Four such shapes were randomly drawn and presented in sequence, one during each part (epoch) of the decision period. The summed evidences provided by the shapes gave the log-odds of reward administration at a red-colored saccade target. At the end of the last epoch, the monkey had to signal its guess of reward-location, by making a saccade to either the red or a green target. During the whole experiment, spiking activity in area LIP was recorded and compared to the probabilistic information the monkey had received so far.

Yang and Shadlen found a quasi-linear relationship between the (shape dependent) momentary reward log-odds l and firing rate r (figure 11a). Likewise, other experimental studies have found LIP firing rates during decision periods to follow linear functions of time. (Shadlen & Newsome, 1996, 2001; Roitman & Shadlen, 2002). Such rate signals have been hypothesized to reflect the accumulated evidence in favor of -or against a particular scene interpretation (left- or rightward dot motion), by representing the log-odds of decision variables (Gold & Shadlen, 2001; Roitman & Shadlen, 2002; Rao, 2004; Beck et al., 2008; Pfeiffer, Nessler, Douglas, & Maass, 2010). This hypothesis is based on the drift-diffusion behavior of the log-odds when, on average, the evidence favors one particular scene interpretation, a fact that poses the defining property of some hallmark models of decision making (Carpenter & Williams, 1995; Gold & Shadlen, 2007).

As we will show in the following, our proposed ISI coding scheme is in accordance with such an interpretation of firing rate and, furthermore, may reproduce the particular shapes of ISI distributions that are observed in LIP in vivo (Maimon & Assad, 2009). The model can also account for the slight deviation from linearity in the Yang&Shadlen data. We first define a binary, ISI-based decision variable $d \in \{0, 1\}$, assuming state '1' when a given ISI, Δt, is below some fixed threshold value θ and assuming state '0' otherwise, i.e. $d(\Delta t) := \{1, \text{if } \Delta t < \theta;\ 0 \text{ otherwise}\}$. In Yang&Shadlen's case for example, state $d = 1$ stands for reward administration at the red target[1]. It follows that for increasing/decreasing the probability mass $P_d(d = 1)$, in general one has to decrease/increase the expected value $\mathbb{E}[\Delta t]$ and therefore to increase/decrease the rate $r = \frac{1}{\mathbb{E}[\Delta t]}$. This way, rate changes are purely epiphenomenal, reflecting redistributions of ISI probability mass, rather than representing changing probabilistic quantities per se.

[1]This is not correct in a strict sense: $d = 1$ stands for reward administration at the target inside the receptive field of a recorded neuron. However, for the sake of simplicity of exposition Yang&Shadlen explained parts of their model using the above definition, to which the exact same concepts apply (see (Yang & Shadlen, 2007), appendix A)

More formally, let $P_d\,(d=1\,|\,r) = \frac{1}{1+\exp[-l(r)]}$ be the probability of reward at the red target, conditioned on rate r through the empirical log-odds-rate relationship $l(r)$ (figure 11a). By assuming $l(r)$ to be monotonic, this implicitly defines a density for r by $p_r(r) := \frac{d}{dr}P_d\,(d=1\,|\,r)$ (we denote probabilities by capitals and densities by lower case letters). We refer to the latter as the 'empirical rate density'.

For deriving the proper conditional ISI distribution $p_{\Delta t}(\Delta t\,|\,r)$ which, according to our hypothesis, gives rise to the experimental data of Yang and Shadlen, we make the assumption that the function class of $p_{\Delta t}(\Delta t\,|\,r)$ is not affected by changes of r and does not depend on epoch number. That is, different r only result in differently scaled ('time-warped') versions of some invariant, template ISI distribution. We have chosen this option, because it leads to rate-independent coefficients of variation (CV), a feature that is also observed for ISI distributions in LIP (Maimon & Assad, 2009). In this context, we define the class template density $p_{\Delta t^*}(\Delta t^*)$, with $\Delta t^* := \frac{\Delta t}{\mathbb{E}[\Delta t]} = r \cdot \Delta t$. It follows $\mathbb{E}\,[\Delta t^*] = 1$ and for any r we can obtain $p_{\Delta t}(\Delta t\,|\,r)$ by time warping the template $p_{\Delta t^*}(\Delta t^*)$. Based on these definitions and given the above threshold model, the following statements must hold true:

$$\begin{aligned} & P_d\,(d=1\,|\,r) = P_{\Delta t}(\Delta t < \theta\,|\,r) = P_{\Delta t^*}(\Delta t^* < r\theta) \\ \Rightarrow \quad & p_r(r) = \theta \cdot p_{\Delta t^*}\,(r\theta) \\ \Rightarrow \quad & \frac{1}{\theta} \cdot p_r\left(\frac{\theta^*}{\theta}\right) = p_{\Delta t^*}\,(\theta^*) \end{aligned} \tag{11}$$

which means that *the desired template ISI density is a time-warped version of the empirical rate density*. Note that this result is unaffected by the assumed ISI transformation function $g(\Delta t)$ (see section 2.3.1). The same analysis can be applied to a threshold model on any such g, resulting in the same equation 11. Moreover, we can deduce the numerical value of θ: Since by construction $\mathbb{E}\,[\Delta t^*] = 1$, it must hold true that $\theta = \frac{1}{\mathbb{E}[r]}$. Applying the time warp to expression 11, the desired ISI density amounts to

$$p_{\Delta t}\,(\Delta t\,|\,r) = \frac{r}{\theta} \cdot p_r\left(\frac{r\Delta t}{\theta}\right) \tag{12}$$

Therefore, we are left with the determination of $p_r(r)$ and consequently the log-odds-rate relationship $l(r)$ from the experimental data. From the data of figure 11a Yang and Shadlen have inferred this relationship to be approximately linear and common to all four epochs (black, dashed lines). In the linear case, the definition of $p_r(r)$ implies the empirical rate density (and consequently the ISI density) to follow a logistic distribution

$$p_r(r) = \frac{m\exp^{-(mr+c)}}{\left(1+\exp^{-(mr+c)}\right)^2} \tag{13}$$

where m, c are the slope and intercept values of $l(r) = mr + c$, determined by inverting the fitted $r(l)$-relationship. Visual inspection of the data indeed suggests a linear model for values around $l = 0$. However, for more extreme values of l, $r(l)$ seems to be of sigmoidal nature and on the extreme positive end, moreover, to be dependent on epoch number. Therefore,

we have also looked at the case when the extreme positive end is neglected and analysis is restricted to the region where a common, epoch-independent $r(l)$-model can be assumed[2]. Using a crossvalidation-like procedure, we determined this region's extent on the positive l-axis (see Materials and Methods section 4.6 for details). In accordance with our subjective impression, the region's positive extents turn out to be 1.11 for the population and 1.16 for the single neuron data. Linear fits to such region-restricted data sets are shown as black, solid lines. Apart from the logistic case, we have also modeled the data by assuming $p_r(r)$ to follow a gamma distribution. Gamma distributions are frequently used for describing recorded ISI histograms in experiments (Maimon & Assad, 2009) and hence are good candidates to be used in equation 12. The fits on the region-restricted data sets in the gamma case are shown as red, solid curves.

Figure 11b suggests that the assumption of a common model is indeed rather justified for the region-restricted than for the complete data sets. In both of these cases, we fitted the data points of each epoch individually and evaluated the training and testing error. Training error was evaluated on the single epoch data that was used for fitting, while testing error was evaluated on all three other epochs. This way, a low testing error indicates a good capability to generalize from one epoch to another. For the single neuron data, the testing error in case of region-restricted data is indeed markedly lower compared to the full data case. This is particularly true for the gamma distribution, which confirms the visibly better fits in (a). The linear model in contrast is unable to account for the sigmoidal curvature at the extreme negative end of l. In case of the population data, there is an apparent discrepancy between the generalization performance of epoch 1 and epochs 2-4. While the former becomes worse during transition from 'linear, complete' to 'gamma, restricted', the opposite holds true for the latter. Most likely, this can be attributed to the necessarily smaller extent of the epoch1-data along the l-axis and its apparent higher residual training error. Collectively, both effects lead to an impaired generalization ability of any model. To verify that the generalization-performance-pattern in the three cases is not tied to the specific, epoch-based choice of training and testing sets, we performed crossvalidation with repeated random subsampling. That is, in each of 900 independent trials we used 10 random and epoch-independent data points for training and the remaining points for testing. The means across trials of the training and testing errors are respectively shown as dashed, horizontal lines. Pairwise differences between those means in the three examined cases were all highly significant ($p << 0.01$, rank-sum test) and show even more clearly that the assumption of a common model is justified rather for the region-restricted than the complete data sets. These differences also imply a $r(l)$-relationship based on gamma $p_r(r)$ to be a better descriptor of the data than a linear $r(l)$.

[2]This is justified because (a) in late epochs, r becomes also dependent on the monkeys choice of eye movement (Yang & Shadlen, 2007), an effect which gains importance for responses at the extreme ends of l, when the animal is more likely to have committed to a choice. (b) there is an asymmetry between choice influence at the extreme positive and the extreme negative end, i.e. when a movement in- or outside the response field of a recorded neuron is more likely[1] (Platt & Glimcher, 1997; Yang & Shadlen, 2007). More specifically, fig. S7 in (Yang & Shadlen, 2007) suggests choice influence as cause of the positive sigmoid, but fails to explain the negative sigmoid, for which we provide an alternative hypothesis in the following.

Figure 11c shows the $p_{\Delta t}(\Delta t \,|\, r)$ corresponding to the various $r(l)$-relationships in (a). A histogram of measured ISIs from a LIP neuron is also shown for comparison. Its unimodal but non-exponential shape is a distinctive feature of LIP/area 5 neurons, compared to visual areas lower in the cortical hierarchy (Maimon & Assad, 2009) and bears similarities to our model distributions. The slightly positive skew of the histogram cannot be reproduced in the (symmetric) logistic case, but can be accounted for by the gamma distribution. As mentioned before, the gamma distribution also leads to better descriptions of the $r(l)$-data. Based on these model distributions we can make concrete and experimentally testable predictions regarding two numerical quantities: a) Following eq.12, for a single measured neuron the CV of the ISI density should match the CV of the empirical rate density. (b) Since $\theta = \frac{1}{\mathbb{E}[r]}$, the threshold parameter θ can also be extracted from the empirical rate density. In the discussion we will elaborate more on how the thus obtained θ could be crosschecked on other neural measurements. Tables 1&2 give the numerical values of both, CV and θ for the various assessments of empirical rate densities in figure 11. In case of prediction (a), the CV values of the ISI distributions should be readily obtainable from the same raw spike data of Yang and Shadlen that has also been used to compute the firing rates.

	Linear Complete	**Linear Restricted**	**Gamma Restricted**
CV	0.11	0.12	0.14
$\boldsymbol{\theta}$ [ms]	54	53	53

Table 1: Numerical values of CV and threshold parameter θ, extracted from the empirical rate densities of the population data

	Linear Complete	**Linear Restricted**	**Gamma Restricted**
CV	0.20	0.27	0.34
$\boldsymbol{\theta}$ [ms]	43	41	39

Table 2: Numerical values of CV and threshold parameter θ, extracted from the empirical rate densities of the single neuron data

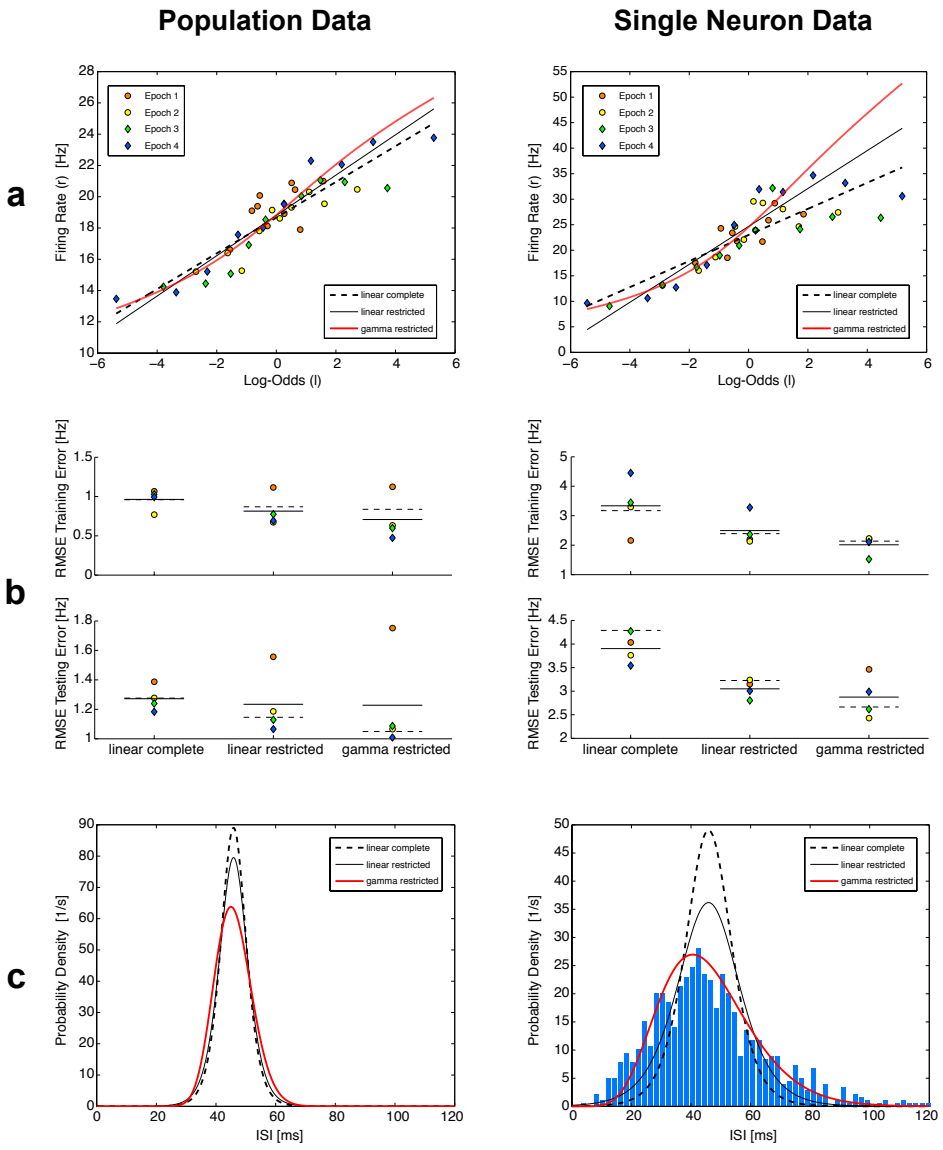

Figure 11: Reinterpretation of the Yang&Shadlen data in the light of ISI coding

Columns: Results with respect to Yang&Shadlen's pooled population and single neuron data.

Rows: a) Raw data (markers) of the $r(l)$ dependency during each epoch (data taken with permission from (Yang & Shadlen, 2007), figures 3b and 2c therein. Here l is based on the natural logarithm). Dashed lines are least-squares regression lines applied to complete data sets, solid lines are such fits when l is region-restricted (see main text). Red/black curves show the case when $p_r(r)$ is assumed to follow a gamma/logistic distribution respectively. Collectively these fits are termed the 'empirical log-odds-rate relationship'.

b) Training and testing error of regression lines fitted exclusively to data points from individual epochs (measured as the root-mean-square error (RMSE)). For the training error, each marker displays the RMSE on the corresponding data from the single epoch that was used for fitting. For the testing error, the same marker shows the RMSE evaluated on the data points from all three other epochs. Marker conventions as in (a), solid horizontal bars denote the average across all markers. Dashed horizontal bars show the mean after crossvalidation (see main text).

c) ISI distributions corresponding to the fitted curves in (a). The blue bars in the right plot form an ISI histogram of a single neuron in LIP (CV = 0.42, reproduced with permission from (Maimon & Assad, 2009), supplementary material). The rates of the fitted distribution were set to 21.8 Hz, the rate of the histogram data.

3 Discussion

We have presented the functional description of a spike-based processor that allows Belief-Propagation to be performed in general (Forney) Factor Graphs, even in presence of analog variables. At its core, the processor is based on a simple Markov Chain Monte Carlo-like sampling method that is able to approximate the otherwise intractable integrals associated with Belief-Propagations Sum-Product-Rule. To achieve this, discrete spike events generated by neurons are interpreted as random samples from their underlying interspike interval distribution, which is directly associated with a BP-message.

The results of direct (black-box) software implementations of the processor were in excellent agreement with those of a discrete approximation by ordinary BP. This has been shown for a single factor node and two different topologies of whole graphs –under random parameterization of the functions defining the involved factors. Despite the highly structured messages produced by BP in general, our ISI coding scheme can be qualitatively reconciled with the comparably restricted class of ISI distributions measured in central neurons. Beyond rate coding, this scheme also offers an alternative explanation for the observed firing rate dynamics in area LIP during behavioral tasks and allows for concrete numerical predictions.

Furthermore, by using the framework of reservoir computing, we have also realized the processor in a network of neural elements. This network is capable of approximating single output messages and it has been verified that an ensemble of such networks allows for inference in Hidden Markov Model-like graph-topologies. By being a functional model however, our abstract BP processor is not confined to a specific implementation strategy. That is, other approaches besides reservoir computing can be considered in this context, e.g. structured approaches that associate specific biophysical mechanisms with the individual functional modules of the processor. In the following section we sketch out some corresponding ideas for future research.

3.1 Mechanistic, Neurobiological Interpretation of the Concrete BP-Processor

The reservoir-based implementation of the abstract BP processor (section 2.2) has a straightforward, if cumbersome, neurobiological interpretation as a network of spiking neurons: Two sets of reservoir networks with dedicated readout neurons serve the purpose of ISI decoding and approximating the Sum-Product Rule respectively. The ISI decoder readouts provide local feedback connections to their respective reservoirs to prolong the memory capacity of the ISI decoder stage. The readout approximating the SPR provides this feedback to inform the reservoir about the time of the last fired spike, as required by the abstract processor description (figure 3). This interpretation is cumbersome, because the whole arrangement consisting of several specifically interconnected networks of neurons has to be realized twice –for forward and backward passing of messages– for each unobserved variable of the underlying FFG. Furthermore, large reservoir networks of neurons have to be 'wasted' just to perform ISI detection.

By abstracting both network tasks (ISI decoding and SPR approximation) however, our processor can be placed on a more general level that offers an additional interpretation: Nonlinear synaptic dynamics and linear summation of dendritic subunit outputs work together to realize the complete BP processor inside the dendritic tree of a single neuron. The resulting voltage trace conveyed to the soma then leads to the appropriate hazard of firing. Using the backpropagating action potential, the soma in turn informs the dendritic tree about the time of the last fired spike. Indeed, by modeling the detailed biophysics of pyramidal neurons, (Poirazi, Brannon, & Mel, 2003a) have suggested terminal dendrites to act as classical quasi-independent sigmoidal units, the workhorse of ANN modeling which has also been used in the present study. The same authors have also extended this model to a two layer network of sigmoidal units describing the global input/output behavior of a pyramidal neuron (Poirazi, Brannon, & Mel, 2003b). In this case, when the synaptic activity vector is replaced by the vector of dendritic subunit outputs, the methods of section 4.3 can still be applied for learning the weights of the output layer. Although in the study of (Poirazi et al., 2003b) the neurons firing rate was taken as the relevant variable, the idea of abstracting single neuron processing to multilayer ANNs has also been examined using the temporal distance between spikes belonging to different inputs (Wang & Liu, 2010). However, the model in (Wang & Liu, 2010) may principally be formulated also in terms of temporal distances between spikes of the same input, leading to ISI decoder units. Therefore, the approach of (Wang & Liu, 2010) opens the possibility for both, ISI decoding and 'dendritic reservoirs', so that our spike-based BP-processor can presumably be realized within that framework. ISI decoding might also be realized, or at least supported, by short term synaptic plasticity mechanisms such as facilitation and depression (Usrey, Reppas, & Reid, 1998; Usrey, Alonso, & Reid, 2000; Reich et al., 2000), features that are present at most central synapses (Usrey et al., 1998). In this context, one may also speculate about a possible role played by spike-frequency adaptation. (Nesse, Maler, & Longtin, 2010) have shown that statistically independent amplitudes of an adaptation current may arise in conjunction with weakly correlated ISIs in spike trains. As an alternative to the renewal ISIs we have considered in this work, such amplitudes could be used to realize the positional sampling process of figure 2, that uses independent updates of each coordinate. However, for this approach to work, a way must be found to convey as input message the information contained in the presynaptic adaptation current to some postsynaptic neuron. (Nesse et al., 2010) have mentioned that synaptic dynamics with a time course similar to the adaptation current can serve this purpose. Since time constants of calcium dependent adaptation currents fall into the $50 - 600$ms range (Y.-H. Liu & Wang, 2001), the time courses of NMDA currents and/or synaptic depression/facilitation could be suitable candidates therefor.

Regardless of the considered neurobiological interpretation however, one disadvantage of the model proposed in figure 3 is the fact that it requires a distributed arrangement of BP processors, one for each outgoing message, which nevertheless are all based on the same underlying factor node. Therefore, the parameterization of the factor function needs to be preserved spatially across all of the processors, a non-trivial requirement in the absence of a very specific supervision signal during learning.

3.2 Experimental Evidence for ISI-Coding in Neural Systems, Model Predictions

The question of neurobiological implementation of the abstract processor relates to the question, whether there exists experimental evidence supporting ISI coding in neural systems. Whereas we have shown in section 2.3 the consistency of ISI coding with some of the electrophysiological data, a substantial number of studies have directly demonstrated the significance of ISIs for processing in neurons of low-level sensory systems across species, sensory modalities and brain areas (Usrey et al., 1998; Cariani, 1999; Usrey et al., 2000; Reich et al., 2000; Fairhall et al., 2001; Lundstrom & Fairhall, 2006; Rathbun, Alitto, Weyand, & Usrey, 2007; Jacobs et al., 2009; Shih et al., 2011). For example, in the motion-sensitive neuron H1 of the fly visual system, changes in stimulus variance can be unambiguously detected using ISI histograms –in marked contrast to information solely provided by instantaneous firing rates (Lundstrom & Fairhall, 2006; Fairhall et al., 2001). This detection happens very fast, allowing perfect stimulus discrimination after $\sim 8-12$ ISIs. Similarly, in a visual 2-alternative forced choice task, a retinal spike code based on ISIs was the only one tested and found able to explain the behavioral performance in a mouse model system (Jacobs et al., 2009). Further upstream the visual hierarchy, it has been shown that the efficiacy of spikes in cat retinothalamic (Usrey et al., 1998) as well as thalamocortical axons (Usrey et al., 2000) for driving a spike in respective postsynaptic neurons smoothly depends on the spikes preceding ISI. Correspondingly, in macaque visual area V1 (Reich et al., 2000) and cat auditory area A1 (Shih et al., 2011), spikes preceded by short ISIs contain a higher amount of information about the stimulus and show greater feature selectivity (Shih et al., 2011). Finally on the psychophysical level, a variety of features of ISI distributions in the auditory nerve and cochlear nucleus can be mapped directly to features of pitch perception (Cariani, 1999).

The ISI encoded stimulus information in low-level areas, e.g. primary sensory cortices, must be picked up by neurons in high-level areas, e.g. secondary sensory and association cortices, in order to infer the identity of the stimulus and/or to select appropriate commands, e.g. in motor tasks. Hence, the sparsest assumption is an uniform coding scheme, such that neurons in high-level areas preserve the ISI coding of neurons in low-level areas. As mentioned in section 2.3.1 however, many studies have reported consistency of in vivo ISI distributions with a class of unimodal distributions, such as quasi-exponential, gamma and log-normal distributions, in area MT (Softky & Koch, 1993; Bair et al., 1994; Maimon & Assad, 2009)), posterior parietal and dorsolateral prefrontal cortex (Compte et al., 2003) and suprasylvian gyrus (Noda & Adey, 1970; Burns & Webb, 1976). These results are usually taken as indications for quasi-poissonian firing and so point to a rather restricted class of ISI distributions present in high-level areas of cortex. We see at least three ways to resolve this issue.

First and foremost, if BP messages are assumed to be encoded as $\log(ISI)$-distributions, section 2.3.1 has shown how rich message information can be concealed behind quasi-exponential ISI distributions. Indeed when ordinary ISI histograms are used, even primary sensory areas display distributions of this type (see (Reich et al., 2000), figure 1). Log-encoding of infor-

mation might be advantageous for the nervous system, e.g. when variables with drastically different dynamic ranges have to be combined. An example of the latter is the equality constraint factor, when used in the context of cue combination as a means of fusing different sensory modalities.

Secondly, the presence of a restricted class of ISI distributions in cortex might be explained by the typically long electrode recording times in the seconds regime (Softky & Koch, 1993; Bair et al., 1994): In general, BP messages in loopy graphs are not stationary from the beginning of an inference trial (in fact message passing may not converge at all (Murphy, Weiss, & Jordan, 1999)), which may lead to smooth ISI histograms that reflect the average of messages computed during the whole trial, rather than the final steady-state messages only. This may explain the lack of features for large ISI values in some experimental histograms, because these features are more susceptible for being averaged out compared to a consistent peak at small ISI values (as it would be present e.g. in case of $\log(ISI)$-coding).

The third objection against poissonian firing is based on a study by Maimon and Assad (Maimon & Assad, 2009). There the authors show that spike time regularity, assessed by the Coefficient of Variation(CV), is larger in high-level association and motor-like parietal regions (area LIP and area 5 respectively) compared to areas lower in the hierarchy (areas MT and MST). If cortex indeed performs computations based on Factor Graphs, unambiguous inference results are expected in high-level areas, where stable interpretations of the world are formed and reliable motor commands have to be read out for execution. In our model, unambiguous marginal probabilities correspond exactly to highly regular spike trains.

This aspect immediately leads to the question, whether the CV values of our ISI distributions modeling the LIP data (figure 11) do match with those measured by Maimon and Assad. Despite qualitative similarities between the distributions, our CV values fall into the range $[0.11, 0.34]$ (tables 1&2) and hence are somewhat smaller than those observed in their study ($CV \in [0.42, 1.50] \,/\, [0.30, 1.70]$, with medians $0.91 \,/\, 0.74$, for neurons in LIP/area 5 respectively). However it must be noted that the behavioral task the monkey had to do in (Maimon & Assad, 2009) is different from the one employed by Yang&Shadlen. In the former case, a ball approaching an obstacle was presented as stimulus and the monkey had to move a lever as soon as it saw a computer controlled reversal of the ball's motion direction. In an alternative set of trials the monkey had to activate the lever just before the ball hit the obstacle. We do not know how such different experimental paradigms affect the $l(r)$ relationship for decision making and consequently the CV of the ISI distributions. Furthermore, this relationship seems to be quite variable across neurons (Yang & Shadlen, 2007) and it is not clear how averaging of neural responses (in case of the population data) affects the resulting CV compared to the CVs of the individual neurons. In the presented case at least, the two were markedly different, with the single neuron data being more close to the data of Maimon and Assad.

Besides the CV, our approach to the LIP data makes quantitative predictions regarding the threshold parameter θ. Even provided the threshold model to be true, it is not clear how this parameter could be measured directly in experiments. However it seems quite reasonable to assume, that in this case the synaptic and/or somatic response of some command executing

neuron to a spike from a probability encoding LIP neuron depends strongly on the spike's preceding ISI Δt. A motor neuron controlling eye movement for example may respond more strongly to spikes with $\Delta t < \theta$ and may initiate action only if such spikes come in at a sufficient rate. Hence, this hypothesis could be tested if in the Yang&Shadlen experiment a LIP neuron and one of its postsynaptic partners were simultaneously recorded. θ could then be determined from the (supposedly different) responses to $\Delta t \gtrless \theta$.

Another prediction of the threshold model is concerned with the accuracy of probability estimation. Because it is the firing rate which determines the number of spike samples that are used for estimation, high probabilities of a particular binary state (which we here called the '1' state) are more accurately represented 'inside' a subject than high probabilities of the corresponding other ('0') state. This in turn is a qualitative prediction that could be tested in behavioral experiments.

Related to this point is the question of the time it takes a biological agent to perform inference, when the posterior distribution is represented by ISIs. That is, how long does the agents' brain need to collect spike samples until the posterior can be reliably read out? In human subjects, psychophysical have revealed reliable inference results to be accessible after rather processing times, that is after less than 150ms (van Rullen & Thorpe, 2001; J. Liu, Harris, & Kanwisher, 2002; Kirchner & Thorpe, 2006), whereas in our model there is obviously a trade-off between processing time and accuracy of the inference procedure. However, this trade-off can be alleviated by considering a population of neurons, whose individual members independently contribute ISIs to the positional sampling process of the BP processor and which all follow the same message ISI distributions. This way, a larger number of samples can be collected on small time scales, e.g. in order to maintain a certain degree of accuracy while parameter W is decreased. In fact, the results presented in section 2.1.2 have been produced by using such a population approach.

3.3 Comparison With Other Spike-Based Inference Approaches

Apart from ISI coding there are several other models using various spike-coding schemes, that propose how Belief-Propagation (Rao, 2004; Ott & Stoop, 2006; Steimer et al., 2009) or other methods of probabilistic inference (Yu & Dayan, 2005; Deneve, Latham, & Pouget, 2001; Deneve, 2007; Börlin & Deneve, 2011; Ma et al., 2006; Litvak & Ullman, 2009) could be performed by spiking neurons. However, these models are limited in that they are confined to a restricted set of graphical models (particular tree-shaped Bayesian networks (Deneve, 2007), naive Bayesian networks (Yu & Dayan, 2005; Ma et al., 2006; Börlin & Deneve, 2011) and pairwise Markov Random Fields (Ott & Stoop, 2006)), or depend on rough biophysical approximations (Rao, 2004). Also, except (Ma et al., 2006; Börlin & Deneve, 2011) all other approaches are based on binary (Ott & Stoop, 2006; Deneve, 2007) or general discrete variables (Yu & Dayan, 2005; Rao, 2004; Steimer et al., 2009; Litvak & Ullman, 2009). In contrast, our model is very general, since it allows for inference in Forney Factor Graphs containing analog variables and hence does not suffer from the aforementioned limitations. Moreover, by being a functional processing model, our approach is not confined to a particular set of neuronal mechanisms for its implementation and therefore very flexible.

This flexibility comes for a price however: In contrast to other authors work (Yu & Dayan, 2005; Ma et al., 2006; Deneve, 2007; Börlin & Deneve, 2011) we can provide less detailed experimental evidence supporting our model. Also, the approach of (Börlin & Deneve, 2011) elegantly combines inference and working memory –a feature we have not considered here. Our focus has mainly been put on algorithmic generality, rather than mimicking the results of psychophysical studies in specific brain areas. The generality of FFGs allows Bayesian inference to be applied in a broader context, e.g. to motor control and face recognition problems (Bessière, Laugier, & Siegwart, 2008; Sudderth et al., 2003), whose corresponding graphical model structures exceed the naive Bayesian networks expressed in pure optimal cue combination. For example think of a human baby trying to imitate the vocal pronunciation of its parents (see (Bessière et al., 2008) last chapter). When observing the parents' lips, while hearing their distinct vocal sound (cue combination), the baby must select appropriate commands for its own lip and tongue muscles in order to properly imitate the vocal (motor action selection).
Given the generality of results presented in section 2.1.2 we have no reason to assume that our BP processor cannot be readily applied to such cognitive modelling. On the contrary a hardware version of the spiking BP processor might even be beneficial, because it abolishes the need for cumbersome discretizations that accompanies software realizations (see (Bessière et al., 2008) last chapter for example).
We have therefore planned to implement the processor as an analog Very Large Scale Integrated (aVLSI) circuit. The inferential results of such a circuit may then be used to learn the parameterization of the underlying factor graph. Indeed samples of the marginal distributions can be applied to a nested EM-procedure (Bishop, 2006), for which we have already derived the corresponding equations.

4 Materials and Methods

4.1 Derivation of $S(z,t)$ and $S_F(t)$

Here we prove that the stochastic processes $S(z,t)$ and $S_F(t)$ approximate scaled versions of $uSPR(z)$ and N respectively. The individual sections thereby cover different aspects of this proof, starting with the main statement and ending with proofs of detailed assertions that are needed in the process of proving the main statement. In section 4.1.1 we derive the expected values of $S(z,t)$ and $S_F(t)$, which turn out to be proportional to $uSPR(z)$ and N respectively. It is then shown in section 4.1.2 that, in relative terms, $S(z,t)$ and $S_F(t)$ converge in probability to these expected values, leading to a steady improvement of the approximation in equation 10 for increasing W. For all derivations we assume that the input ISI distributions $m_X(x)$ and $m_Y(y)$ are both stationary, rendering $S(z,t)$ and $S_F(t)$ stationary as well.

4.1.1 Derivation of $\mathbb{E}[\boldsymbol{S(z)}]$ and $\mathbb{E}[\boldsymbol{S_F}]$

Let $[G(\delta)](t) := H(t) - H(t-W)$ be the impulse-response of filter G constituting the sliding summation window, where $H(t)$ denotes the Heaviside step function. It follows that $\int_0^{T>W} [G(\delta)](t)\,\mathrm{d}t = W$. Let $a_z(t) := \sum_{i=1}^{\infty} f_{z,i}\delta(t-t_i)$ be the function spike train caused by the incoming spiking messages $m_X(x)$ and $m_Y(y)$, where $f_{z,i} := f_z(x_i, y_i)$ represents the i-th function spike. Given the verbal definition of $S(z,t)$ in section 2.1.1, we can see that $S(z,t) := [G(a_z)](t)$. With $\operatorname*{plim}_{T\to\infty}(\cdot)$ denoting convergence in probability of $(\cdot)$ and $\mathbb{E}[n_{tot}]$ denoting the expected total number of function spikes in $[0,T]$, stationarity of $S(z,t)$ leads to the following expression for the expected value $\mathbb{E}[S(z)]$:

$$\begin{aligned}
\mathbb{E}[S(z)] &= \operatorname*{plim}_{T\to\infty} \frac{1}{T}\int_0^T S(z,t)\,\mathrm{d}t \\
&= \operatorname*{plim}_{T\to\infty} \frac{1}{T}\cdot W \cdot \sum_{i:t_i\in[0,T]} f_{z,i} \\
&= W \cdot \operatorname*{plim}_{T\to\infty} \frac{1}{T}\cdot \mathbb{E}[n_{tot}]\cdot\mathbb{E}[f_z]
\end{aligned} \tag{14}$$

The last equation above follows from the convergence behavior of $\sum_{i:t_i\in[0,T]} f_{z,i}$, which is examined in section 4.1.2.
Clearly it holds true that

$$\mathbb{E}[n_{tot}] = r_{tot}\cdot T \tag{15}$$

with $r_{tot} := \frac{1}{\mathbb{E}[X]} + \frac{1}{\mathbb{E}[Y]}$ the total rate of incoming spikes along X and Y.
Since $f_z(x,y)$ is a deterministic function of ISI random variables X and Y, its expected value $\mathbb{E}[f_z]$ is given by (Papoulis & Pillai, 2002):

$$\begin{aligned}
\mathbb{E}[f_z] &= \int_D\int_D f_z(x,y)p_{\boldsymbol{s}}(x,y)\,\mathrm{d}x\,\mathrm{d}y \\
&= \int_D\int_D f_z(x,y)m_X(x)m_Y(y)\,\mathrm{d}x\,\mathrm{d}y \\
&= uSPR(z)
\end{aligned} \tag{16}$$

where $p_{\boldsymbol{s}}$ denotes the density of the positional sample vector $\boldsymbol{s} := (x,y)$. In figure 12 it is shown that, by following the sampling scheme of figure 2b, $p_{\boldsymbol{s}}$ is indeed given by $p_{\boldsymbol{s}}(x,y) = m_X(x)m_Y(y)$. Plugging 15 and 16 into the last expression of 14 one ends up with:

$$\mathbb{E}[S(z)] = r_{tot}\cdot W\cdot uSPR(z) \tag{17}$$

The derivation of $\mathbb{E}[S_F]$ runs along the same lines yielding:

$$\mathbb{E}[S_F] = r_{tot}\cdot W\cdot N \tag{18}$$

□

4.1.2 Convergence Behavior of $S(z)$ and S_F

Here we examine the behavior of sums of (integral) function spikes in the limit of large temporal summation window sizes T. First, it is proven in the main proposition 4.1 that, under weak conditions imposed on m_X, m_Y and f, these sums converge in probability to their respective expected values. Thereby equation 10 and the final step of equation 14 are justified. The proof is based on a decoupling of the limits $T \to \infty$ and $n_f \to \infty$, where n_f is a fixed number of (integral) function spikes disregarding time window $[0, T]$. Accordingly, the claim about limit $n_f \to \infty$ needed by the main proposition is proven in lemma 4.2. In that proof we also specify in detail the above conditions on m_X and m_Y. Finally, lemma 4.3 establishes the link between the two types of limits. To simplify notation we refer in the following to function f as both, f_z in case of function spikes and F in case of integral function spikes.

Proposition 4.1. *Let $A := \{i \in \mathbb{N} \,|\, t_i \in [0, T]\}$ be a sorted set of indices of (integral) function spikes within some time window T, such that $t_i < t_{i+1}\, \forall i \in A$. Let $b := \sum_{i \in A} f_i$ be the corresponding sum of $n_{tot} := |A|$ (integral) function spikes. Then, provided that f is bounded and the ISI distributions m_X and m_Y are well behaved (see proof of lemma 4.2 for detailed conditions),*

$$\operatorname*{plim}_{T \to \infty} \frac{b}{\mathbb{E}[n_{tot}] \cdot \mathbb{E}[f]} = 1 \tag{19}$$

where $\mathbb{E}[f]$ is the expected value of an (integral) function spike when the sampling scheme of figure 2b is followed and $\operatorname{plim}_{T \to \infty} C_T = C$ denotes convergence in probability of C_T to C as $T \to \infty$. In other words, as $T \to \infty$ the relative error between b and $\mathbb{E}[n_{tot}] \cdot \mathbb{E}[f]$ converges to zero in probability.

Proof. The proof is complicated by the fact that n_{tot} and the values of the samples f_i are not independent, since the latter are determined through ISIs in the fixed time window $[0, T]$. Therefore, we first decouple the (integral) function spikes from that window by forming $b^* := \sum_{i=1}^{n_f} f_i$ with a fixed number n_f, for which it is shown in lemma 4.2 that

$$\lim_{n_f \to \infty} \mathrm{CV}^2[b^*] = 0 \tag{20}$$

with $\mathrm{CV}^2[b^*] := \frac{\mathrm{Var}[b^*]}{\mathbb{E}^2[b^*]}$ and $\mathbb{E}[b^*] = n_f \cdot \mathbb{E}[f]$. That is, the limit $\lim_{n_f \to \infty}(\cdot)$ keeps on adding (integral) function spikes to b^*, disregarding whether or not these n_f number of spikes have

occurred in $[0, T]$. It then follows from 20 that $\frac{b^*}{n_f \mathbb{E}[f]}$ converges to 1 in mean-square and hence also in probability (Papoulis & Pillai, 2002), that is

$$\operatorname*{plim}_{n_f \to \infty} \frac{b^*}{n_f \mathbb{E}[f]} = 1 \tag{21}$$

holds true. Finally lemma 4.3 shows that assertion 19 follows from assertion 21, which completes the proof.

□

Lemma 4.2. *Let* $b^* := \sum_{i=1}^{n_f} f_i$ *be a sum of* n_f *(integral) function spikes that are not restricted to fall into any temporal summation window. Define* $\mathrm{CV}^2[b^*] := \frac{\mathrm{Var}[b^*]}{\mathbb{E}^2[b^*]}$ *as the squared coefficient of variation of* b^**. Then, provided that* f *is bounded and the ISI distributions* m_X *and* m_Y *are well behaved,*

$$\lim_{n_f \to \infty} \mathrm{CV}^2[b^*] = 0 \tag{22}$$

and

$$\mathbb{E}[b^*] = n_f \cdot \mathbb{E}[f]. \tag{23}$$

Proof. For the proof we need to derive $\mathbb{E}[b^*] = n_f \cdot \mathbb{E}[f]$ and an upper bound on $\mathrm{Var}[b^*]$ that increases slower than $\mathbb{E}^2[b^*]$ for $n_f \to \infty$. The former is easily achieved by using the result of figure 12, from which it follows that the arguments (x_i, y_i) of each (integral) function spike $f_i(x_i, y_i)$ are identically distributed. Therefore, the expected value of the sum of n_f such spikes is given by $n_f \cdot \mathbb{E}[f]$.

To achieve the latter we first write

$$\mathrm{Var}[b^*] = \sum_{i=1,j=1}^{n_f} \mathrm{Cov}[f_i, f_j] = n_f \cdot \mathrm{Var}[f] + 2 \sum_{i=1}^{n_f} \sum_{i<j \leq n_f} \mathrm{Cov}[f_i, f_j] \tag{24}$$

Note that f_i and f_j are independent, if at least one X- and one Y-spike have occurred between f_i and (including) f_j. This is due to the renewal property and mutual independence of the two spike trains, which assure that both arguments X, Y of f have been independently updated in this case. Accordingly, we define by $\xi_k \in \{0, 1\}$ the binary event that such a two-fold update has happened for two (integral) function spikes with index difference $k = j - i$. Therefore, $\mathrm{Cov}[f_i, f_j \,|\, \xi_{j-i} = 1] = 0$. With $P_{\xi_k}(\xi_k = 1/0)$ denoting the probability of ξ_k, we

may use the law of total covariance to decompose $\mathrm{Cov}\left[f_i, f_j\right]$ for $i < j$:

$$\begin{aligned}
\mathrm{Cov}\left[f_i, f_j\right] &= \mathbb{E}\left[\,\mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i}\right]\right] + \mathrm{Cov}\left[\,\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right], \mathbb{E}\left[f_j \,|\, \xi_{j-i}\right]\right] && (25)\\
\mathbb{E}\left[\,\mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i}\right]\right] &= \mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i} = 0\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0) && (26)\\
&\quad + \mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i} = 1\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 1)\\
&= \mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i} = 0\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0)\\
\mathrm{Cov}\left[\,\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right], \mathbb{E}\left[f_j \,|\, \xi_{j-i}\right]\right] &= \mathbb{E}\left[\,\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right] \cdot \mathbb{E}\left[f_j \,|\, \xi_{j-i}\right]\right] && (27)\\
&\quad - \mathbb{E}\left[\,\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right]\right] \cdot \mathbb{E}\left[\,\mathbb{E}\left[\,f_j \,|\, \xi_{j-i}\right]\right]\\
&= \mathbb{E}\left[f_i \,|\, \xi_{j-i} = 0\right] \cdot \mathbb{E}\left[f_j \,|\, \xi_{j-i} = 0\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0)\\
&\quad + \mathbb{E}\left[f_i \,|\, \xi_{j-i} = 1\right] \cdot \mathbb{E}\left[f_j \,|\, \xi_{j-i} = 1\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 1)\\
&\quad - \mathbb{E}\left[f\right]^2
\end{aligned}$$

We can infer from figure 12 that f_i and ξ_{j-i} are independent, i.e. that $\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right] = \mathbb{E}\left[f\right]$. However, in general the same must not hold for $f_j, i < j$. Hence it follows from 27:

$$\begin{aligned}
\mathrm{Cov}\left[\,\mathbb{E}\left[f_i \,|\, \xi_{j-i}\right], \mathbb{E}\left[f_j \,|\, \xi_{j-i}\right]\right] &= \mathbb{E}\left[f\right]\Big(\mathbb{E}\left[f_j \,|\, \xi_{j-i} = 0\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0) && (28)\\
&\quad + \mathbb{E}\left[f_j \,|\, \xi_{j-i} = 1\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 1)\Big) - \mathbb{E}\left[f\right]^2\\
&= \mathbb{E}\left[f\right]\Big(\mathbb{E}\left[\,\mathbb{E}\left[f_j \,|\, \xi_{j-i}\right]\right]\Big) - \mathbb{E}\left[f\right]^2 = 0
\end{aligned}$$

Since $f \geq 0$, plugging 26 and 28 into 25 yields:

$$\begin{aligned}
\mathrm{Cov}\left[f_i, f_j\right] &= \mathrm{Cov}\left[f_i, f_j \,|\, \xi_{j-i} = 0\right] \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0) && (29)\\
&\leq M^2 \cdot P_{\xi_{j-i}}(\xi_{j-i} = 0) && (30)
\end{aligned}$$

with $M := \sup\left[f\right]$. Therefore,

$$\mathrm{Var}\left[b^*\right] \leq n_f \,\mathrm{Var}\, f + 2M^2 \sum_{i=1}^{n_f} \sum_{i<j\leq n_f} P_{\xi_{j-i}}(\xi_{j-i} = 0) \tag{31}$$

We now assume that $m_X(x), m_Y(y)$ are well behaved, i.e. that $\lim_{n_f\to\infty} \sum_{i<j\leq n_f} P_{\xi_{j-i}}(\xi_{j-i} = 0) = C$, with C some constant. For biologically relevant ISI distributions this condition can always be fulfilled by assuming arbitrarily small/large minimum/maximum ISIs, below/above which the two densities vanish. In this case a two-fold coordinate update ($\xi_k = 1$) will always happen after a finite number K of updates, i.e. $P_{\xi_{j-i}}(\xi_k = 0) = 0$, $\forall k \geq K$. Hence,

$$\mathrm{Var}\left[b^*\right] \leq n_f \left(\mathrm{Var}\left[f\right] + 2M^2 C\right) \tag{32}$$

□

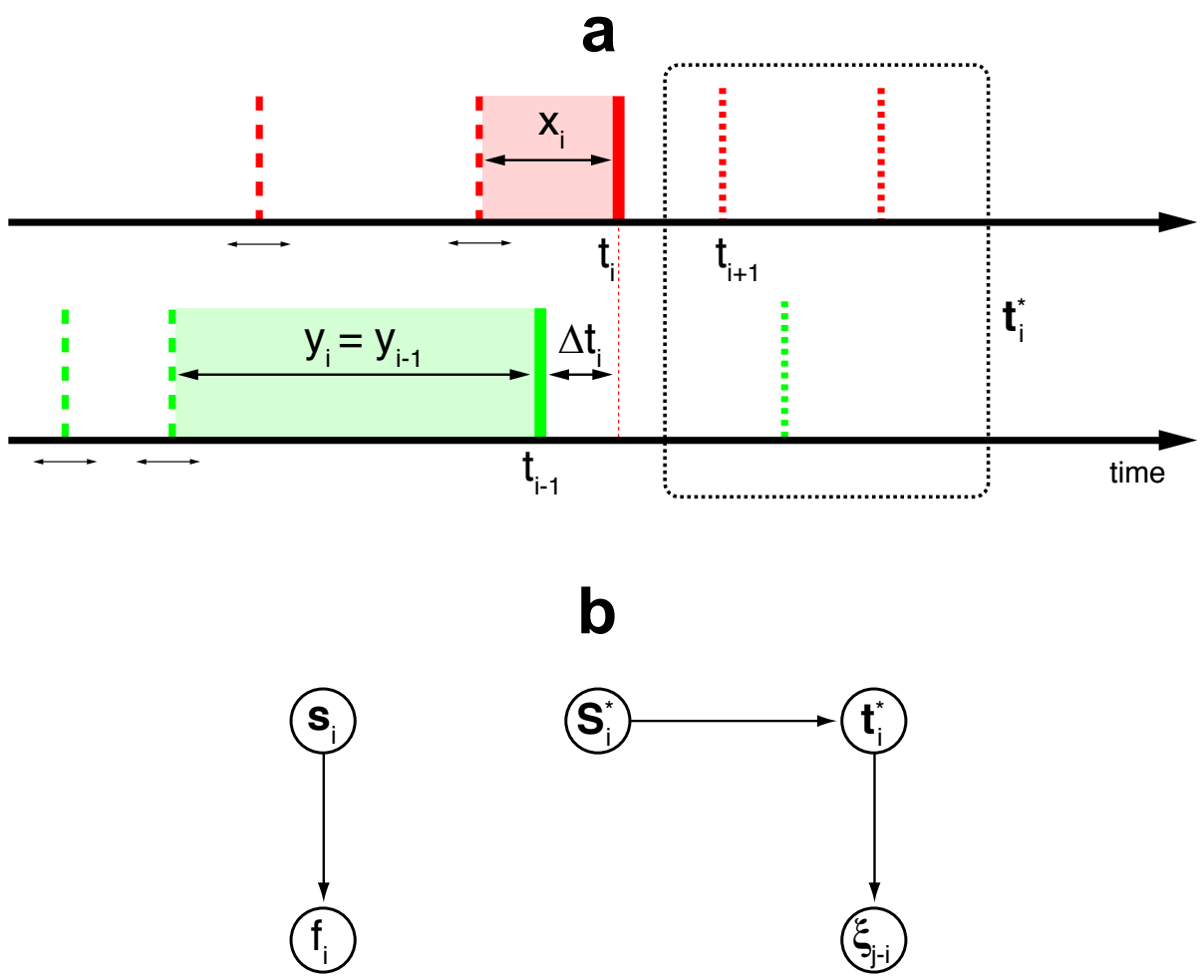

Figure 12: Density of positional sample vectors and (in)dependencies between spike-based quantities

a) Four basic independency statements: 1) Without loss of generality suppose the last (integral) function spike f_i to have happened at time t_i along X. f_i depends exclusively on the positional sample vector $\boldsymbol{s}_i := (x_i, y_i)$, given by the last two respective ISIs along X and Y. In other words, given $\boldsymbol{s}_i$, f_i is conditionally independent of any other variable. 2) Denote by Δt_i the difference between t_i and the time of the last spike along Y before t_i, i.e. $\Delta t_i = t_i - t_{i-1}$ in the illustrated case. Denote by $\boldsymbol{S}_i^*$ the state vector $\boldsymbol{S}_i^* := (t_i, \Delta t_i)$. Because the two spike trains are independent and renewal, $\boldsymbol{S}_i^*$ is not informative about the ISIs x_i, y_i before t_i and t_j respectively. It follows $p_{\boldsymbol{s}}(\boldsymbol{s}_i \,|\, \boldsymbol{S}_i^*) = m_X(x_i) m_Y(y_i) = p_{\boldsymbol{s}}(\boldsymbol{s}_i)$, in particular $\boldsymbol{s}_i$ and $\boldsymbol{S}_i^*$ are independent. 3) With $l_i \in \{0, 1\}$ denoting a binary label indicating whether or not spike i was received along X, it also follows from the renewal property and spike train independence that, given $\boldsymbol{S}_i^*$, future spiking behavior $\boldsymbol{t}_i^* := (t_{i+1}, l_{i+1}, t_{i+2}, l_{i+2}, ...)$ is conditionally independent from any other variable and hence also from past spiking $(t_{i-2}, l_{i-2}, t_{i-3}, l_{i-3}, ...)$. In particular $p_{\boldsymbol{S}}(\boldsymbol{t}_i^* \,|\, \boldsymbol{S}_i^*, \boldsymbol{s}_i) = p_{\boldsymbol{S}}(\boldsymbol{t}_i^* \,|\, \boldsymbol{S}_i^*)$. 4) ξ_{j-i}, $j > i$ is is a function exclusively of $\boldsymbol{t}_i^*$, i.e. ξ_{j-i} is conditionally independent from any other variable.

b) Bayesian network modeling the (in)dependencies between the variables in (a). Any additional edge would violate any of the four independence statements $(1), (2), (3), (4)$ in (a). Clearly, it follows from the graph that f_i and ξ_{j-i} are marginally independent, that is $P_{\xi_{j-i}}(\xi_{j-i} \,|\, f_i) = P_{\xi_{j-i}}(\xi_{j-i})$.

It remains to be shown in proposition 4.1 that assertion 19 follows from assertion 21, which is achieved by the following lemma.

Lemma 4.3. *Let all quantities be as defined in proposition 4.1. If*

$$\operatorname*{plim}_{n_f \to \infty} \frac{b^*}{n_f \cdot \mathbb{E}[f]} = 1 \tag{33}$$

then

$$\operatorname*{plim}_{T \to \infty} \frac{b}{\mathbb{E}[n_{tot}] \cdot \mathbb{E}[f]} = 1 \tag{34}$$

with $\mathbb{E}[n_{tot}] = r_{tot} \cdot T$ *and* $r_{tot} := \frac{1}{\mathbb{E}[X]} + \frac{1}{\mathbb{E}[Y]}$ *denoting the total rate of spikes coming in as messages along* X *and* Y *respectively.*

Proof. Preliminaries: To avoid cluttered notation, define $n := n_f$ and let $p(x)$ be the shorthand for the probability density $p_X(x)$ of some random variable X evaluated at x. Disregarding window $[0, T]$, define by $T_{\epsilon,n}$ the time such that after n (integral) function spikes the probability of $t_n \geq T_{\epsilon,n}$ is ϵ, with $\epsilon > 0$ an arbitrary constant. Denote by b_n^* the random variable that is equal to b^* in case of $t_n < T_{\epsilon,n}$ and hence follows the same distribution as $p(b^* \,|\, t_n < T_{\epsilon,n})$. Analogously, b_n denotes the random variable that is equal to b in case of $t_n < T_{\epsilon,n} \leq T$. The difference between b_n^* and b_n is that the latter might comprise additional (integral) function spikes in $[t_n, T]$, i.e. $n_{tot} > n$ in this case. Let $m(\epsilon, T)$ be the largest (integral) function spike index such that $T_{\epsilon,m(\epsilon,T)} \leq T$ holds for any given T, i.e. $m(\epsilon, T) := \max(n \,|\, T_{\epsilon,n} \leq T)$.

Because the (integral) function spikes are governed by independent renewal processes, we know that $\operatorname*{plim}_{n \to \infty} \frac{n}{r_{tot} \cdot t_n} = 1$. It follows that if 33 holds, then

$$\operatorname*{plim}_{n \to \infty} \frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} = 1 \tag{35}$$

since $\operatorname*{plim}_{n \to \infty} (C_n \cdot D_n) = \operatorname*{plim}_{n \to \infty} C_n \cdot \operatorname*{plim}_{n \to \infty} D_n$ holds for convergent sequences of random variables C_n, D_n. Furthermore, by the above definitions we know that:

$$\begin{aligned} p\left(\frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]}\right) = {} & p\left(\frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} \,\middle|\, t_n < T_{\epsilon,n}\right)(1-\epsilon) \\ & + p\left(\frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} \,\middle|\, t_n \geq T_{\epsilon,n}\right)\epsilon \end{aligned} \tag{36}$$

and

$$p\left(\frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} \,\middle|\, t_n < T_{\epsilon,n}\right) = p\left(\frac{b_n^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]}\right) \tag{37}$$

Hence, if 35 holds then

$$\forall \epsilon_1, \delta > 0 \, \exists N : P\left(\left|\frac{b^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} - 1\right| > \delta\right) < \epsilon_1, \, \forall n \geq N \tag{38}$$

$$\Rightarrow P\left(\left|\frac{b_n^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} - 1\right| > \delta\right) < \frac{\epsilon_1}{1-\epsilon} \tag{39}$$

$$\Rightarrow \operatorname*{plim}_{n\to\infty} \frac{b_n^*}{r_{tot} \cdot t_n \cdot \mathbb{E}[f]} = 1 \tag{40}$$

where $P(\alpha)$ is the probability of event α. Because $m(\epsilon, T)$ is monotonically increasing as a function of T, we can conclude from 40 that

$$\operatorname*{plim}_{T\to\infty} \frac{b_{m(\epsilon,T)}^*}{r_{tot} \cdot t_{m(\epsilon,T)} \cdot \mathbb{E}[f]} = 1 \tag{41}$$

Additionally, since the (integral) function spikes are governed by independent renewal processes, we know that $\operatorname*{plim}_{T\to\infty} \frac{t_{m(\epsilon,T)}}{T} = 1$, from which $\operatorname*{plim}_{T\to\infty} \frac{b_{m(\epsilon,T)}}{b_{m(\epsilon,T)}^*} = 1$ follows, leading to

$$\operatorname*{plim}_{T\to\infty} \frac{b_{m(\epsilon,T)}}{r_{tot} \cdot T \cdot \mathbb{E}[f]} = 1 \tag{42}$$

Using

$$p\left(\frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]}\right) = (1-\epsilon) \cdot p\left(\frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]} \,\middle|\, t_{m(\epsilon,T)} < T_{\epsilon,m(\epsilon,T)}\right) + \epsilon \cdot p\left(\frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]} \,\middle|\, t_{m(\epsilon,T)} \geq T_{\epsilon,m(\epsilon,T)}\right) \tag{43}$$

and

$$p\left(\frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]} \,\middle|\, t_{m(\epsilon,T)} < T_{\epsilon,m(\epsilon,T)}\right) = p\left(\frac{b_{m(\epsilon,T)}}{r_{tot} \cdot T \cdot \mathbb{E}[f]}\right) \tag{44}$$

we can conclude from 42

$$\forall \epsilon_1, \delta > 0 \, \exists T_0 : P\left(\left|\frac{b_{m(\epsilon,T)}}{r_{tot} \cdot T \cdot \mathbb{E}[f]} - 1\right| > \delta\right) < \epsilon_1, \, \forall T \geq T_0 \tag{45}$$

$$\Rightarrow P\left(\left|\frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]} - 1\right| > \delta\right) < (1-\epsilon) \cdot \epsilon_1 + \epsilon \cdot 1 \tag{46}$$

$$\Rightarrow \operatorname*{plim}_{T\to\infty} \frac{b}{r_{tot} \cdot T \cdot \mathbb{E}[f]} = 1 \tag{47}$$

where the last implication follows from our free choice of ϵ, e.g. as $\epsilon < \epsilon_1$.

□

4.2 Calculation of $\mathbb{E}\left[\boldsymbol{D}\left(\boldsymbol{B}^*\|\boldsymbol{P}\right)\right]$ and $\mathrm{Var}\left[\boldsymbol{D}\left(\boldsymbol{B}^*\|\boldsymbol{P}\right)\right]$ Used in Figures 5, 6, 8 and 9

Given a fixed distribution $P(x)$ we are interested in having expressions for $\mathbb{E}\left[D(B^*\|P)\right]$ and $\mathrm{Var}\left[D(B^*\|P)\right]$, when $B^*(x)$ is a random distribution taken from a uniform distribution of distributions of variable X. For that we first require the marginal probability densities $P_{single}(b_x)$ and $P_{pair}(b_x, b_y)$ of the coefficients $b_x := B^*(x); b_y := B^*(y)$. These densities are necessary to compute the various expected values occurring in the expressions for $\mathbb{E}\left[D(B^*\|P)\right]$ and $\mathrm{Var}\left[D(B^*\|P)\right]$. When the distribution $B^*(x)$ is drawn uniformly from the $(n-1)$-dimensional probability simplex –with n denoting the number of states of X– we obtain the following:

$$P_{single}(b_x) = (n-1)(1-b_x)^{n-2} \tag{48}$$

$$P_{pair}(b_x, b_y) = (n-1)(n-2)(1-b_x-b_y)^{n-3} \tag{49}$$

Define $f(b_x, p_x) := b_x \cdot \ln\left(\frac{b_x}{p_x}\right)$ and let $\gamma \approx 0.577$ be the Euler-Mascheroni constant. Denote by H_n the n-th Harmonic Number and by $\mathcal{U}(n)$ the uniform distribution over n states.
It follows:

$$\mathbb{E}\big[D(B^*\|P)\big] = \sum_x \mathbb{E}\big[f(b_x, p_x)\big] \tag{50}$$

$$= \sum_x \int_0^1 P_{single}\left(b_x\right) f\left(b_x, p_x\right)\, \mathrm{d}b_x \tag{51}$$

$$= 1 - H_n - \sum_x \ln p_x \tag{52}$$

$$\approx 1 - \gamma + \sum_x \left(\frac{1}{n}\ln\left(\frac{1}{n}\right) - \frac{1}{n}\ln p_x\right); \qquad (n \to \infty) \tag{53}$$

$$= (1-\gamma) + D\big(\mathcal{U}(n)\|P\big) \tag{54}$$

$$\mathrm{Var}\big[D(B^*\|P)\big] = \sum_x \mathrm{Var}\big[f(b_x, p_x)\big] + \sum_{\substack{x,y \\ x\neq y}} \mathrm{Cov}\big[f(b_x, p_x), f((b_y, p_y)\big] \tag{55}$$

$$= \sum_x \mathbb{E}\big[f^2(b_x, p_x)\big] + \sum_{\substack{x,y \\ x\neq y}} \mathbb{E}\big[f(b_x, p_x)\cdot f(b_y, p_y)\big] - \tag{56}$$

$$\sum_{x,y} \mathbb{E}\big[f(b_x, p_x)\big] \cdot \mathbb{E}\big[f(b_y, p_y)\big]$$

computation of the integrals

$$\mathbb{E}\big[f^2(b_x,p_x)\big] = \int_0^1 P_{single}\,(b_x)\,f^2\,(b_x,p_x)\;\mathrm{d}b_x \tag{57}$$

$$\mathbb{E}\big[f(b_x,p_x)\cdot f(b_y,p_y)\big] = \int_0^1\int_0^{1-b_x} P_{pair}\,(b_x,b_y)\,f\,(b_x,p_x)\,f\,(b_y,p_y)\;\mathrm{d}b_y\,\mathrm{d}b_x \tag{58}$$

and subsequent rearrangement/elimination of terms then yields:

$$\mathrm{Var}\big[D(B^*\|P)\big] = \frac{3n\,\|\boldsymbol{l_P}\|_2^2 + (\pi^2-6)\,n^2 - 3\,\|\boldsymbol{l_P}\|_1^2}{3n^2(n+1)} - \psi^{(1)}(n+1) \tag{59}$$

where $\|\boldsymbol{l_P}\|_1$ and $\|\boldsymbol{l_P}\|_2$ are respectively the 1- and 2-norm of vector $\boldsymbol{l_P} := \{\ln(p_{x_1}),...,\ln(p_{x_n})\}$ and $\psi^{(1)}(\cdot)$ is the polygamma function of order 1.

4.3 Derivation of the Cost Function Used for Training $\boldsymbol{R_{out}}$

Readout R_{out} in figure 7 has been trained using a different cost function than readouts R_X and R_Y. For a spiking neuron subject to escape noise –with hazard activation function $h(t,\boldsymbol{w}) := h(V_m(t,\boldsymbol{w}))$, whose membrane potential V_m is given by a weighted sum $V_m(t,\boldsymbol{w}) = \boldsymbol{w}^T\cdot\boldsymbol{X}(t)$ of its synaptic inputs $\boldsymbol{X}(t)$– we derive here suitable cost functions for learning the neuron to fire with a given ISI distribution $p(\Delta t,t)$. This distribution is explicitly allowed to depend on t (as for example in equation 10). The first learning procedure is based on the log-likelihood of an observed spike train and is semi-supervised and online in nature. In the context of learning precise temporal spike patterns, this method has first and independently been described by (Pfister et al., 2006). These authors also show how it relates to biologically plausible Spike Time Dependent Plasticity (STDP) weight update schemes. We also present a second related cost that is easier to work with in practice and which has been used to produce the results of section 2.2.

Denote by $T_s := \{t_i \,|\, i = 1...N, S(t_i) = 1\}$, the sorted set of spike times of the learning neuron, with $S(t_i) := 1$ iff the neuron has fired a spike at time t, $S(t_i) := 0$ otherwise. As before, let $p(\Delta t,t)$ be the *target unconditioned* ISI distribution at time t, and $q(\Delta t,t,\boldsymbol{w})$ be the corresponding *actual unconditioned* ISI distribution of the neuron. For the online method suppose the ISI distribution of T_s follows $p(\Delta t,t)$ and the neuron is teacher-forced to fire at times T_s. Then our goal is to minimize the negative log-likelihood $C(\boldsymbol{w})$ of the training set T_s:

$$C(\boldsymbol{w}) \;:=\; \sum_{i=2}^{N} -\ln q(t_i - t_{i-1}, t_i, \boldsymbol{w}); \quad t_i \in T_s,\; N := |T_s| \tag{60}$$

Using standard results of renewal theory, $q(\Delta t, t_{i-1} + \Delta t, \boldsymbol{w})$ can be explicitely expressed in terms of the hazard as (Cox & Lewis, 1966; Gerstner & Kistler, 2002):

$$q(\Delta t, t_{i-1} + \Delta t, \boldsymbol{w}) = h(t_{i-1} + \Delta t, \boldsymbol{w}) \cdot \exp\left(- \int_0^{\Delta t} h(t_{i-1} + \Delta t^{'}, \boldsymbol{w}) \, \mathrm{d}\Delta t^{'} \right) \tag{61}$$

yielding

$$C(\boldsymbol{w}) \;\; = \;\; \sum_{i=2}^{N} - \ln h(t_i, \boldsymbol{w}) + \int_{t_{i-1}}^{t_i} h(t^{'}, \boldsymbol{w}) \, \mathrm{d}t^{'} \tag{62}$$

After having computed the gradient of equation 62 (Pfister et al., 2006), one can perform a stochastic gradient descent procedure (Bottou, 1998) based on a biologically plausible, online learning paradigm. Interestingly, on the postsynaptic side learning in this fashion is feasible only at time points when the neuron spikes, i.e. at times $t_i \in T_s$, since expression 62 requires integration from t_{i-1} to t_i (Pfister et al., 2006). This feature however is a necessary requirement in many biological accounts of learning, e.g. in models involving STDP (Pfister et al., 2006; Gerstner & Kistler, 2002). Furthermore, under rather mild and biologically plausible conditions on $h(V_m)$, $C(\boldsymbol{w})$ is a convex function and therefore devoid of local minima (Paninski, 2004). These conditions are also fulfilled by $h(V_m) = h_\Theta e^{\frac{V_m - V_\theta}{\sigma_h}}$, the hazard activation function commonly used in the neuroscientific literature (Pfister et al., 2006, 2010; Rao, 2005; Eggert & van Hemmen, 2001) and in section 2.2.1.

Despite these desirable properties, the presented optimization method is technically impractical for the processor approximation task of section 2.2.1. Since, as stated before, weight updates are restricted to times when the learning neuron spikes, times t' between spikes cannot be used for learning, leading to long learning times until enough training examples have been absorbed for proper learning. Hence, the results presented in section 2.2.1 are based on optimizing a slightly different cost function $C_{batch}(\boldsymbol{w})$ in batch-mode using the BFGS quasi-Newton method (Bertsekas, 1999):

$$C_{batch}(\boldsymbol{w}) \;\; := \;\; \sum_{i=2}^{N} D(p_i^* \| q_i^*) \tag{63}$$

$$p_i^*(\Delta t) \;\; := \;\; \frac{p(\Delta t, t_{i-1} + \Delta t)}{\displaystyle\int_0^{t_i - t_{i-1}} p(\Delta t', t_{i-1} + \Delta t') \, \mathrm{d}\Delta t'} \tag{64}$$

$$q_i^*(\Delta t, \boldsymbol{w}) \;\; := \;\; \frac{q(\Delta t, t_{i-1} + \Delta t, \boldsymbol{w})}{\displaystyle\int_0^{t_i - t_{i-1}} q(\Delta t', t_{i-1} + \Delta t', \boldsymbol{w}) \, \mathrm{d}\Delta t'}; \quad t_i \in T_s \tag{65}$$

where $p_i^*(\Delta t)$ and $q_i^*(\Delta t, \boldsymbol{w})$ are the above target and actual ISI distributions *given* that a spike is fired for sure within the interval $(t_i, t_{i-1}]$. The normalization imposed by such conditioning is necessary because of the neurons' random firing as well as p's explicit dependence

on time, both of which generally prevent $p(\Delta t, t_{i-1} + \Delta t)$ and $q(\Delta t, t_{i-1} + \Delta t)$ from being properly normalized probability densities of Δt, which in turn may render the KL divergence D a negative quantity. Plugging 61 into 63 and rearranging the terms, one obtains:

$$C_{batch}(\boldsymbol{w}) = \sum_{i=2}^{N} \int_0^{\infty} p_i^*(\Delta t) \cdot \left(\int_0^{\Delta t} h(t_{i-1} + \Delta t', \boldsymbol{w}) d\Delta t' - \ln h(t_{i-1} + \Delta t, \boldsymbol{w}) \right) \mathrm{d}\Delta t \quad (66)$$
$$+ \sum_{i=2}^{N} \ln \big(Z_i(\boldsymbol{w}) \big) - \sum_{i=2}^{N} H(p_i^*)$$
$$Z_i(\boldsymbol{w}) := \int_0^{t_i - t_{i-1}} h(t_{i-1} + \Delta t, \boldsymbol{w}) \cdot \exp \left(- \int_0^{\Delta t} h(t_{i-1} + \Delta t', \boldsymbol{w}) \, \mathrm{d}\Delta t' \right) \mathrm{d}\Delta t \quad (67)$$

where $Z_i(\boldsymbol{w})$ is the normalization term of equation 65 and $H(p_i^*)$ the differential Shannon entropy of p_i^*.
In contrast to $C(\boldsymbol{w})$, for $C_{batch}(\boldsymbol{w})$ one can use time points between spikes as training examples by constructing a long time series $p(\Delta t, t)$, which is reset at assumed random spike times T_s, without actually simulating a spiking neuron with escape rate in online mode. This then allows $C_{batch}(\boldsymbol{w})$ to be minimized in batch mode.

4.4 Parameters Used for the Black-Box Implementations of the Abstract Processor

The parameters below were used for the simulations in section 2.1.2 (figures 5 and 6). Quantities based on time are measured in dimensionless steps:

$W = 750$; Domain of variable values (possible ISI values) $= \{i \in \mathbb{N} \,|\, i \leq 100\}$ (if by chance a processor did not fire after at most 200 steps, a spike was forced to keep the inferential dynamics ongoing); Number of gaussian mixture components defining factor nodes of dimension 1/2/3/4: 5/10/20/20 respectively; Range of standard deviations along principal components of the gaussians defining the mixture of 1/2/3/4-dimensional factors: $[3, 7]/[3, 10]/[3, 15]/[3, 15]$ respectively (factors of higher dimension needed larger standard deviations and/or more mixture components to maintain a reasonable filling of their domain space); Range of unnormalized weights of the mixture components $= [0.1, 1]$; Span used for moving average filtering of the raw ISI histograms: 3 bins.

4.5 Parameters Used for the ESN Implementations of the Abstract Processor

The following reservoir parameters have been used for the simulated Echo-State networks of section 2.2.2 (see (Jäger, Lukoševičius, Popovici, & Siewert, 2007) for parameter definitions of the sigmoidal leaky-integrator neurons of the reservoirs):

ISI Decoders (L_X, L_Y, R_X, R_Y):

Number of reservoir neurons = 300; reservoir sparsity (probability of connecting two reservoir neurons) = 0.10; spectral radius $|\lambda_{max}| = 0.90$; leak $a = 1$; time-step $\delta = 1$; time constant $c = 5$; scaling factor of the low-passed filtered spike inputs m_X and m_Y: 0.80; scaling factor of the feedback connections from R_X, R_Y: 0.80

SPR Computer (L_{out}, R_{out}):

Number of reservoir neurons = 1500; reservoir sparsity (probability of connecting two reservoir neurons) = 0.03; spectral radius $|\lambda_{max}| = 0.97$; leak $a = 1$; time-step $\delta = 1$; time constant $c = 2.5$; scaling factor of the inputs from R_X and R_Y to L_{out}: 0.013; scaling factor of the low-passed filtered spike feedback from R_{out}: 0.80

4.6 Crossvalidation-like Procedure for Determining the Range of Validity of a Common Model Underlying the Yang and Shadlen Data

The goal of the following procedure was to find the borders of a region along the l-axis, within which the assumption of a common model is justified. We define a common model to be a distinct $r(l)$ relationship that is followed during each epoch, i.e. one that does not change with epoch number. For determining this region, we first formed all 'tangential' subsets of the (l, r) data set. That is, for each l-datum we grouped the datum itself and its first and second nearest neighbors into one subset. For each such set we then regressed r on l, using three of the five points for fitting and the remaining two points for testing. We conducted this procedure for all $\binom{5}{3}$ combinations of fitting points, each time determining the resulting testing errors. Error was measured as the 'root mean squared error' (RMSE). To avoid artifacts induced by the small extent of epoch 1 along the l-axis, we discarded these data points from the subsets (see also main text).

The approach is based on the assumption that, while moving along the l-axis, a transition from a common model to multiple ones is accompanied by a marked change in the distribution of testing errors. In particular, the error distribution associated with the multiple-model region is expected to have increased mean and variance. Due to the limited size of both of the data sets (population and single neuron data) these distributions cannot be assessed for each tangential subset. However they can be roughly characterized by their respective mean and variance. Hence, we performed k-means clustering on the mean-variance data, estimated for each tangential subset. For both data sets, the algorithm revealed the following three distinct clusters ($k = 3$):

- Cluster 1: Low mean and low variance
- Cluster 2: Moderate mean and low variance

- Cluster 3: Large mean and large variance

Members of cluster 3 were well separated from the other data points and were confined to the region beyond $l = 1.11/l = 1.16$ for the population and single neuron data respectively. For $k = 2$, clusters 1 and 2 were merged into a single cluster, cluster 3 remained constant, apart from 1 out of 8 data points swapping its membership with cluster 2 in case of the single neuron data. For $k = 4$ and $k = 5$ either cluster 1 or 3 was subdivided further depending on the data set. Independent of $k \in \{2, 3, 4, 5\}$ the cluster borders $l = 1.11/l = 1.16$ along the l-axis remained. All results were robust against metric changes, e.g. when the 'mean absolute error' was used in place of the RMSE or when the standard deviation instead of the variance was used as a defining feature of the error distribution.

In summary these results strongly suggest an abrupt change in the $r(l)$ relationship when a threshold at $l \approx 1.1$ is crossed. We take this as an indication for the existence of multiple $r(l)$-models beyond this threshold.

5 Acknowledgments

We thank Michael Pfeiffer and Adrian Whatley for their reviews of the manuscript and Florian Blättler for his corrections of the proofs in sections 4.1 and 4.2. We also thank Tianming Yang and Gaby Maimon for sharing their experimental data and the anonymous reviewers for their work and valuable suggestions.

References

Aji, M., & McEliece, R. (2000, March). The Generalized Distributive Law. *IEEE Transactions on Information Theory*, *46*(2), 325-343.

Bair, W., Koch, C., Newsome, W., & Britten, K. (1994, May). Power Spectrum Analysis of Bursting Cells in Area MT in the Behaving Monkey. *The Journal of Neuroscience*, *14*(5), 2870-2892.

Beck, J., Ma, W., Kiani, R., Hanks, T., Churchland, A., Roitman, J., et al. (2008, December). Probabilistic Population Codes for Bayesian Decision Making. *Neuron*, *60*, 1142-1152.

Bertsekas, D. (1999). *Nonlinear programming* (2nd ed.). P.O. Box 391, Belmont, Mass. 02478-9998 U.S.A.: Athena Scientific.

Bessière, P., Laugier, C., & Siegwart, R. (2008). *Probabilistic reasoning and decision making in sensory-motor systems* (P. Bessière, C. Laugier, & R. Siegwart, Eds.). Springer-Verlag, Berlin Heidelberg.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer Science+Business Media.

Börlin, M., & Deneve, S. (2011, February). Spike-Based Population Coding and Working Memory. *PLoS Computational Biology*, *7*(2), 1-18.

Bottou, L. (1998). *On-line learning in neural networks* (D. Saad, Ed.). The Edinburgh Building, Cambridge CB2 2RU, United Kingdom: Cambridge University Press.

Burns, B., & Webb, A. (1976). The Spontaneous Activity of Neurones in the Cat's Cerebral Cortex. *Proceedings of the Royal Society of London. Series B*, *194*, 211-223.

Cariani, P. (1999). Temporal coding of periodicity pitch in the auditory system: An overview. *Neural Plasticity*, *6*(4), 147-172.

Carpenter, R., & Williams, M. (1995, September). Neural computation of log-likelihood in control of saccadic eye movements. *Nature*, *377*, 59–62.

Compte, A., Constantinidis, C., Tegnér, J., Raghavachari, S., Chafee, M., Goldman-Rakic, P., et al. (2003, May). Temporally Irregular Mnemonic Persistent Activity in Prefrontal Neurons of Monkeys During a Delayed Response Task. *Journal of Neurophysiology*, *90*, 3441-3454.

Cox, D., & Lewis, P. (1966). *The statistical analysis of series of events*. London: Methuen.

Dauwels, J., Korl, S., & Löliger, H. (2006, July). Particle methods as message passing. *IEEE International Symposium on Information Theory*, 2052-2056.

Deneve, S. (2007). Bayesian Spiking Neurons I: Inference. *Neural Computation*, *20*(1), 91-117.

Deneve, S., Latham, P., & Pouget, A. (2001). Efficient Computation and Cue Integration with Noisy Population Codes. *Nature Neuroscience*, *4*(8), 826-831.

Douglas, R., & Martin, K. (2004). Neuronal Circuits of the Neocortex. *Annual Review of Neuroscience*, *27*, 419-451.

Eggert, J., & van Hemmen, J. (2001, September). Modeling Neuronal Assemblies: Theory and Implementation. *Neural Computation*, *13*(9), 1923-1974.

Ernst, M., & Banks, M. (2002). Humans integrate visual and haptic information in a statistically optimal fashion. *Nature*, *415*, 429-433.

Fairhall, A., Lewen, G., Bialek, W., & de Ruyter van Steveninck, R. (2001, August). Efficiency and Ambiguity in an Adaptive Neural Code. *Nature*, *412*, 787-792.

Forney, G. (2001). Codes on Graphs: Normal Realizations. *IEEE Transactions on Information Theory*, *47*(2), 520-548.

Geman, S., & Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*, 721-741.

Gerstner, W., & Kistler, W. (2002). *Spiking Neuron Models*. Cambridge University Press.

Gold, J., & Shadlen, M. (2001, January). Neural computations that underlie decisions about sensory stimuli. *TRENDS in Cognitive Sciences*, *5*(1), 10-16.

Gold, J., & Shadlen, M. (2007, July). The Neural Basis of Decision Making. *Annual Review of Neuroscience*, *30*(1), 535-574.

Ihler, A. T., Fisher, J. W., & Willsky, A. S. (2005, May). Loopy Belief Propagation: Convergence and Effects of Message Errors. *Journal of Machine Learning Research*, *6*, 905-936.

Isard, M. (2003, June). PAMPAS: Real-Valued Graphical Models for Computer Vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 1, p. 613-620).

Jacobs, A., Fridman, G., Douglas, R., Alam, N., Latham, P., Prusky, G., et al. (2009, April). Ruling Out and Ruling In Neural Codes. *Proc. Natl. Acad. Sci. USA*, *106*(14), 5936-5941.

Jäger, H. (2001). *The "Echo State" Approach to Analysing and Training Recurrent Neural Networks* (Tech. Rep. No. 148). German National Research Institute for Computer Science.

Jäger, H., Lukoševičius, M., Popovici, D., & Siewert, U. (2007). Optimization and Applications of Echo State Networks with Leaky-Integrator Neurons. *Neural Networks*, *20*, 335-352.

Kirchner, H., & Thorpe, S. (2006). Ultra-Rapid Object Detection With Saccadic Eye Movements: Visual Processing Speed Revisited. *Vision Research*, *46*, 1762–1776.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models* (T. Dietterich, Ed.). Cambridge Massachusetts: MIT Press.

Kschischang, F., Frey, B., & Löliger, H. (2001, February). Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, *47*(2), 498-519.

Litvak, S., & Ullman, S. (2009, November). Cortical circuitry implementing graphical models. *Neural Computation*, *21*(11), 3010-3056.

Liu, J., Harris, A., & Kanwisher, N. (2002, September). Stages of processing in face perception: An MEG study. *Nature Neuroscience*, *5*(9), 910-916.

Liu, Y.-H., & Wang, X.-J. (2001). Spike-Frequency Adaptation of a Generalized Leaky Integrate-and-Fire Model Neuron. *Journal of Computational Neuroscience*, *10*, 25-45.

Löliger, H. (2004, January). An Introduction to Factor Graphs. *IEEE Signal Proc. Mag.*, *21*, 28-41.

Lundstrom, B., & Fairhall, A. (2006, August). Decoding Stimulus Variance from a Distributional Neural Code of Interspike Intervals. *Journal of Neuroscience*, *26*(35), 9030-9037.

Ma, W., Beck, J., Latham, P., & Pouget, A. (2006). Bayesian Inference with Probabilistic Population Codes. *Nature Neuroscience*, *9*(11), 1432-1438.

Maass, W., Joshi, P., & Sontag, E. (2007, January). Computational Aspects of Feedback in Neural Circuits. *PLoS Computational Biology*, *3*(1), 1-20.

Maass, W., Natschläger, T., & Markram, H. (2002). Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computation*, *14*, 2531-2560.

MacKay, D. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge: Cambridge University Press.

Maimon, G., & Assad, J. (2009, May). Beyond poisson: Increased spike-time regularity across primate parietal cortex. *Neuron*, *62*, 426-440.

Murphy, A., Weiss, Y., & Jordan, M. (1999, July). Loopy Belief Propagation For Approximate Inference: An Empirical Study. *Proc. 15th Conf. Uncertainty in Artificial Intelligence*, 467-475.

Nesse, W., Maler, L., & Longtin, A. (2010, December). Biophysical Information Representation in Temporally Correlated Spike Trains. *Proc. Natl. Acad. Sci. USA*, *107*(51), 21973–21978.

Noda, H., & Adey, W. (1970). Firing Variability in Cat Association Cortex during Sleep and Wakefulness. *Brain Research*, *18*, 513-526.

Ott, T., & Stoop, R. (2006). The Neurodynamics of Belief-Propagation on Binary Markov Random Fields. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems* (Vol. 18, p. 1057-1064). Cambridge MA: MIT Press.

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

Papoulis, A., & Pillai, S. (2002). *Probability, Random Variables and Stochastic Processes* (4th ed.). 7 West Patel Nagar, New Delhi 110 008: Tata McGraw-Hill Publishing Company Limited.

Pfeiffer, M., Nessler, B., Douglas, R., & Maass, W. (2010). Reward-Modulated Hebbian Learning of Decision Making. *Neural Computation*, *22*, 1399-1444.

Pfister, J., Dayan, P., & Lengyel, M. (2010, October). Synapses with Short-Term Plasticity are Optimal Estimators of Presynaptic Membrane Potentials. *Nature Neuroscience*, *13*(10), 1271-1275.

Pfister, J., Toyoizumi, T., Barber, D., & Gerstner, W. (2006). Optimal Spike-Timing-Dependent Plasticity for Precise Action Potential Firing in Supervised Learning. *Neural Computation*, *18*, 1318-1348.

Platt, M., & Glimcher, P. (1997). Responses of Intraparietal Neurons to Saccadic Targets and Visual Distractors. *Journal of Neurophysiology*, *78*, 1574-1589.

Poirazi, P., Brannon, T., & Mel, B. (2003a, March). Arithmetic of Subthreshold Synaptic Summation in a Model CA1 Pyramidal Cell. *Neuron*, *37*, 977-987.

Poirazi, P., Brannon, T., & Mel, B. (2003b, March). Pyramidal Neuron as Two-Layer Neural

Network. *Neuron*, *37*, 989-999.

Rao, R. (2004). Bayesian Computation in Recurrent Neural Circuits. *Neural Computation*, *16*(1), 1-38.

Rao, R. (2005). Hierarchical Bayesian Inference in Networks of Spiking Neurons. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in Neural Information Processing Systems* (Vol. 17). Cambridge MA: MIT Press.

Rathbun, D., Alitto, H., Weyand, T., & Usrey, W. (2007, May). Interspike interval analysis of retinal ganglion cell receptive fields. *Journal of Neurophysiology*, *98*, 911-919.

Reich, D., Mechler, F., Purpura, K., & Victor, J. (2000, March). Interspike intervals, receptive fields and information encoding in primary visual cortex. *Journal of Neuroscience*, *20*(5), 1964-1974.

Roitman, J., & Shadlen, M. (2002, November). Response of Neurons in the Lateral Intraparietal Area during a Combined Visual Discrimination Reaction Time Task. *Journal of Neuroscience*, *22*(21), 9475-9489.

Shadlen, M., & Newsome, W. (1996). Motion Perception: Seeing and Deciding. *Proc. Natl. Acad. Sci. USA*, *93*, 628-633.

Shadlen, M., & Newsome, W. (2001). Neural Basis of a Perceptual Decision in the Parietal Cortex (Area LIP) of the Rhesus Monkey. *Journal of Neurophysiology*, *86*, 1916-1936.

Shih, J., Atencio, C., & Schreiner, C. (2011, February). Improved stimulus representation by short interspike intervals in primary auditory cortex. *Journal of Neurophysiology*, *105*, 1908-1917.

Softky, W., & Koch, C. (1993). The Highly Irregular Firing of Cortical Cells Is Inconsistent with Temporal Integration of Random EPSPs. *The Journal of Neuroscience*, *13*(1), 334-350.

Steimer, A., Maass, W., & Douglas, R. (2009, September). Belief Propagation in Networks of Spiking Neurons. *Neural Computation*, *21*(9), 2502-2523.

Sudderth, E., Ihler, A., Freeman, W., & Willsky, A. (2003, June). Nonparametric Belief Propagation. In *Proceedings of the 2003 IEEE Conference on Computer Vision and Pattern Recognition* (Vol. 1, p. 605-612).

Usrey, W., Alonso, J.-M., & Reid, R. (2000). Synaptic interactions between thalamic inputs to simple cells in cat visual cortex. *Journal of Neuroscience*, *20*(14), 5461-5467.

Usrey, W., Reppas, J., & Reid, R. (1998, September). Paired-spike interactions and synaptic efficacy of retinal inputs to the thalamus. *Nature*, *395*, 384-387.

van Rullen, R., & Thorpe, S. (2001). Is it a bird? is it a plane? ultra-rapid visual categorization of natural and artificial objects. *Perception*, *30*, 655-668.

Wang, Y., & Liu, S. (2010). Multilayer processing of spatiotemporal spike patterns in a neuron with active dendrites. *Neural Computation*, *22*, 2086-2112.

Yang, T., & Shadlen, M. (2007, June). Probabilistic Reasoning By Neurons. *Nature*, *447*, 1075-1082.

Yedidia, J., Freeman, W., & Weiss, Y. (2005, July). Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms. *IEEE Transactions on Information Theory*, *51*(7), 2282-2312.

Yu, A., & Dayan, P. (2005). Inference, Attention, and Decision in a Bayesian Neural Architecture. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems* (Vol. 17, p. 1577-1584). Cambridge MA: MIT Press.