A General Framework for Constructing Large-Scale, Biologically Plausible Simulations

Chris Eliasmith and Charles H. Anderson

21st February 2003

1 A framework

There have recently been calls for the development of a theoretical framework in neuroscience that can help unify the field [1]. Such a framework, it is hoped, can provide structure to the somewhat chaotic landscape of experimental and theoretical results in contemporary neuroscience. In this talk, we discuss our attempt to provide such a framework [2].

Our approach is effectively summarized by the following three principles:

1. Neural representations are defined by the combination of nonlinear encoding (exemplified by neuron tuning curves) and weighted linear decoding.
2. Transformations of neural representations are functions of the variables represented by neural populations. Transformations are determined using an alternately weighted linear decoding.
3. Neural dynamics are characterized by considering neural representations as control theoretic state variables. Thus, the dynamics of neurobiological systems can be analyzed using control theory.

In addition to these main principles, we take the following addendum to be important for analyzing neural systems:

- Neural systems are subject to significant amounts of noise. Therefore, any analysis of such systems must account for the effects of noise.

We do not consider this addendum separately because, unlike the principles, it is a claim about how to *analyze* neural systems, not a claim about their basic *function*. Nevertheless, it is essential for applying and carefully articulating the principles.

1.1 Principle 1 - Representation

We take representation in neural systems to be defined by nonlinear encoding and linear decoding over both time and populations of neurons. Consider a population of rate neurons whose activities, $a_i(\mathbf{x})$, encode some vector, $\mathbf{x}$. The activity of these neurons can be expressed as $a_i(\mathbf{x}) = G_i\left[J_i(\mathbf{x})\right]$, where $G_i\left[\cdot\right]$ is the nonlinear function describing the neuron's response function and $J_i(\mathbf{x})$ is the current entering the soma. The current itself is determined by $J_i(\mathbf{x}) = \alpha_i \left\langle \mathbf{x} \cdot \tilde{\boldsymbol{\phi}}_i \right\rangle + J_i^{bias}$. This last equation expresses that the current in the soma is proportional to some gain, α_i, times the degree to which the vector, $\mathbf{x}$, matches the 'preferred stimulus', $\tilde{\boldsymbol{\phi}}_i$, of the neuron, plus some bias current, J_i^{bias}, that accounts for background activity. The resulting function, $a_i(\mathbf{x})$, is the neuron's tuning curve.

Given this encoding, we can estimate the originally encoded vector by decoding the neural activities, i.e. $\hat{\mathbf{x}} = \sum_i \left[a_i(\mathbf{x}) + \eta_i\right] \boldsymbol{\phi}_i^{\mathbf{x}}$. The decoding vectors, $\boldsymbol{\phi}_i^{\mathbf{x}}$, can be found by a least-squares method, such as singular

value decomposition. The noise term, η_i, is assumed to be independent Gaussian noise with a mean of zero (from now on, we do not explicitly include the noise term for clarity). This completes a characterization of the population representation of the vector $\mathbf{x}$ in the activities, a_i, in a neural population.

To extend this characterization of representation to spiking neurons, we draw on work that has shown that most of the information in neural spike trains can be extracted using a linear decoding [3]. The encoding into spikes is done as before, where $G_i[\cdot]$ now defines a spiking nonlinearity (e.g., a simple leaky integrate-and-fire (LIF) neuron). We can now let the neural activities, $a_i(t)$, in the population representation be the spike trains, i.e. $a_i(t) = \sum_n \delta_i(t - t_n)$. In this equation, $t_n$are the nth spike times for neuron i. These spike trains can then be decoded by a linear decoding function (or filter), $h_i(t)$, giving $\hat{\mathbf{x}}(t) = \sum_n h(t) * \delta(t - t_n)$. Elsewhere we have shown that the post-synaptic currents (PSCs) are usefully understood as playing the role of this filter in real neural systems [2]. Combining this temporal code with the previously defined population code gives the following, unified population-temporal code:

$$\begin{aligned}
\sum_n \delta_i(t - t_n) &= G_i\left[\alpha_i \left\langle \tilde{\boldsymbol{\phi}}_i \cdot \mathbf{x}(t) \right\rangle + J_i^{bias}\right] \qquad \text{Encoding} \\
\hat{\mathbf{x}}(t) &= \sum_{in} \delta_i(t - t_n) * h_i(t) \boldsymbol{\phi}_i^{\mathbf{x}} \qquad \text{Decoding} \\
&= \sum_{in} \delta_i(t - t_n) * \boldsymbol{\phi}_i^{\mathbf{x}}(t) \\
&= \sum_{in} \boldsymbol{\phi}_i^{\mathbf{x}}(t - t_n)
\end{aligned}$$

where $\boldsymbol{\phi}_i^{\mathbf{x}}$ are the optimal population decoders, $\boldsymbol{\phi}_i^{\mathbf{x}}(t)$ are the product of the population decoders and the temporal filter, $h_i(t)$ (i.e., the normalized PSC). As a result, $\boldsymbol{\phi}_i^{\mathbf{x}}(t)$ define combined population-temporal decoders for the spike trains. In this representation, the error of the decreases as $1/Number\,of\,Neurons$ under noise.

1.2 Principle 2 - Transformation

In order to make these representations *useful*, we need to be able to define transformations of these representations (i.e. functions of the encoded variables). To do so, rather than finding the optimal decoders, $\boldsymbol{\phi}_i^{\mathbf{x}}$, to extract the original variable, $\mathbf{x}$, from the information encoded by the neural spikes, we can find decoders, $\boldsymbol{\phi}_i^{f(\mathbf{x})}$, to extract some function $f(\mathbf{x})$ from those spikes:

$$\hat{f}(\mathbf{x}; t) = \sum_i a_i(\mathbf{x}(t)) \boldsymbol{\phi}_i^{f(\mathbf{x})}.$$

Again, these transformational decoders can be found using least squares methods.

Notably, both linear and nonlinear functions of the encoded variable can be computed in this manner. We can use this approach to determine precisely what set of functions can be computed with this linear decoding given any particular population of neurons [2].

1.3 Principle 3 - Dynamics

In order to characterize the dynamic properties of neural systems, we integrate our characterization of representation and transformation with modern control theory. In particular, we suppose that the variables represented by a neural population can be understood as control theoretic state variables. To realize this supposition, we need to describe neural dynamics generally. It can shown that the dynamics of the PSC is likely to dominate the dynamics of the cellular response as a whole [2]. As a result, it is reasonable to characterize the dynamics of neural populations based on their synaptic dynamics.

Recall that the state equation in modern control theory that governs dynamics is $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$. The dynamics matrix, $\mathbf{B}$, and the recurrent matrix, $\mathbf{A}$, completely describe the dynamics of a system,

given the state variables $\mathbf{x}(t)$ and the input $\mathbf{u}(t)$. Taking the Laplace transform of (**??**) gives: $\mathbf{x}(s) = h(s)\left[\mathbf{A}\mathbf{x}(s) + \mathbf{B}\mathbf{u}(s)\right]$, where $h(s) = \frac{1}{s}$. In general, we can write any linear control system in this form.

For a neural system, we can use the first approximation to synaptic dynamics given by $h(t) = \frac{1}{\tau}e^{-t/\tau}$, where τ is the synaptic time constant.[1] The Laplace transform of this filter is $h'(s) = \frac{1}{1+s\tau}$. In order to make the dynamics of a neural population (defined by $\mathbf{A}'$ and $\mathbf{B}'$) equivalent to the dynamics expressed by the standard control theory equation (defined by $\mathbf{A}$ and $\mathbf{B}$), we need to determine the relation between matrices $\mathbf{A}$ and $\mathbf{A}'$ and matrices $\mathbf{B}$ and $\mathbf{B}'$ given the difference between $h(s)$ and $h'(s)$. To do so, we can solve for $\mathbf{x}(s)$ in both cases and equate the resulting expressions for $\mathbf{x}(s)$. This gives

$$\mathbf{A}' = \tau\mathbf{A} + \mathbf{I} \quad (1)$$

$$\mathbf{B}' = \tau\mathbf{B}. \quad (2)$$

Substituting this neural control characterization into the original encoding gives a general form for the spiking responses of neurons participating in the realization of a (linear) neural control structure:

$$\sum_n \delta_i(t - t_n) = G_i\left[\alpha_i\left\langle\tilde{\phi}_i\left(h_i(t) * \left[\mathbf{A}'\mathbf{x}(t) + \mathbf{B}'\mathbf{u}(t)\right]\right)\right\rangle_m + J_i^{bias}\right].$$

The procedures we have used are general enough to allow us to construct a neurobiologically realistic implementation of any dynamical system we can define using the techniques of standard modern control theory. Importantly, this means that, given the general nature of the transformations we can compute using this framework, both time-varying and nonlinear control systems can be implemented in this manner.

1.4 Synthesis - A generic neural subsystem

Combining the preceding characterizations of representation, transformation, and dynamics in neurobiological systems results in the generic neural subsystem shown in figure 1.

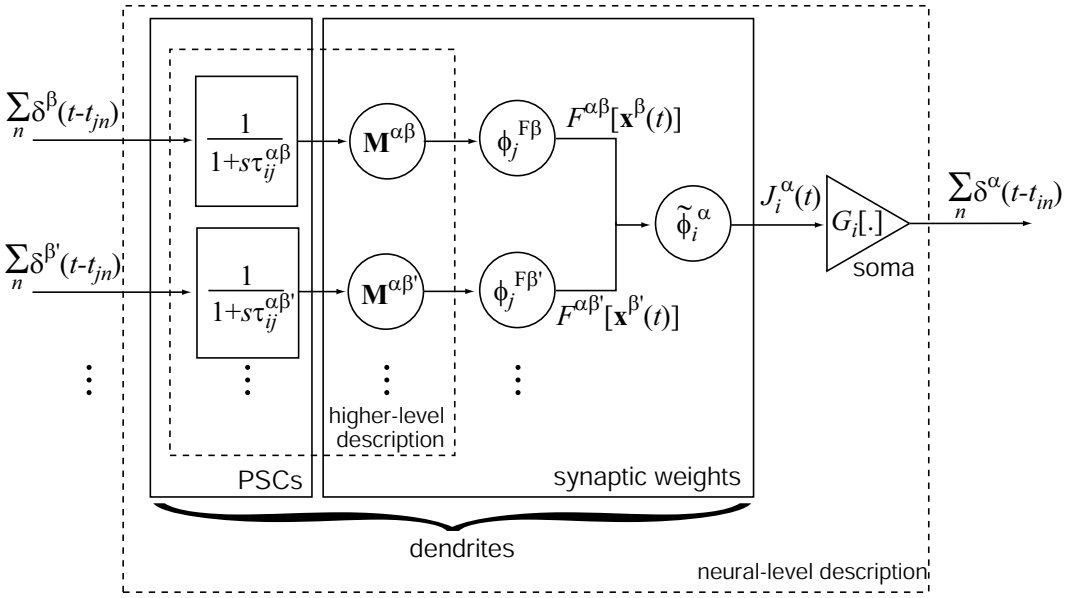

Figure 1: A generic neural subsystem. See text for further discussion.

[1] More complex forms of the PSC can be used, but the increased complexity in the subsequent derivation only serves to obscure the method.

In figure 1, the interior dotted line indicates the higher-level description of the overall neural dynamics. Separating out these elements can be practically useful because it allows for a computationally cheap means of characterizing the system's behavior. The solid boxes highlight the dendritic components, which have been separated into postsynaptic filtering by the PSCs, and the synaptic weights. These components, along with the somatic nonlinearity account for a standard description of neural function: i.e., spikes impact dendrites that give rise to PSCs weighted by a synaptic weight whose effects at the soma result in the generation of outgoing spikes.

The weights themselves are determined by three matrices: 1) the decoding matrix whose elements are the decoders, $\phi_i^{F\beta}$, for some (possibly nonlinear) function, F, of the signal, $\mathbf{x}^\beta(t)$, that comes from a preceding population, β; 2) the encoding matrix whose elements are the encoders, $\tilde{\phi}_i^\alpha$, for this population; and 3) the higher-level transformation matrix, $\mathbf{M}^{\alpha\beta}$, that defines the transformations necessary for implementing the desired control system. In general, the weights are of the form $\omega_{ij}^{\alpha\beta} = \left\langle \tilde{\boldsymbol{\phi}}_i^\alpha \mathbf{M}^{\alpha\beta} \boldsymbol{\phi}_j^{F\beta} \right\rangle$. Given this formulation, the synaptic weights can be computed directly to implement the control system specified over the representations.

1.5 Discussion

Taken together, these three principles can serve to direct the construction of large-scale, biologically plausible simulations. In addition, these principles help to unify several central concepts in computational neuroscience. For instance, principle 1 unifies population and temporal representation in neural systems via the definition of a population-temporal decoder. As well, principles 1 and 2 taken together provide a unified characterization of representation and transformation (or computation) as optimal linear decoding. Considering all three principles together, we can see how this approach can be used to unify top-down and bottom-up evidence on a single neural system. High-level hypotheses, which inform the control theoretic description of the overall system is integrated with the evidence regarding individual neurons, such as tuning curves and response properties.

Furthermore, the principles are general. While we have presented a characterization of the principles in terms of vector representation, there are equivalent characterizations for scalar and function representations, and their combinations (e.g. vector fields). As well, the characterization generalizes over modeling assumptions made regarding individual neuron behavior (e.g., rate, spiking, LIF, full conductance models, etc.). Finally, the principles generalize over transformations (i.e., linear and nonlinear), and dynamics (i.e., time invariant, time varying, linear, nonlinear, or stochastic).

2 Examples

To demonstrate the results of applying these principles, we have chosen three examples of models we have constructed, which represent the tripartite division of neural systems into sensory, cognitive, and motor systems. All of these models have been presented at past CNS conferences.

1. Vestibular system (sensory) [5]: This is a large-scale, spiking neuron model that solves a nonlinear control problem: estimating inertial acceleration given linear acceleration (otoliths) and angular velocity (semi-circular canals). The model employs 6-dimensional vector representation, nonlinear transformations, and feedback.

2. Working memory (cognitive) [4]: This fully recurrent spiking model accounted for two phenomena previously unaccounted for by neural models (parametric variation and multiple target representation in working memory). The model employs function representation.

3. Lamprey locomotion (motor) [6]: This spiking model demonstrates the synthesis of top-down and bottom-up data in a model. The result is a model that guarantees certain high-level behavior (e.g. swimming stability over a range of frequencies), unlike past models. As well this model shows how different levels of description can be concurrently simulated.

3 Benefits

In constructing these and other models, it has become clear to us that there are benefits in adopting this approach for both experimentalists and theoreticians. A number of these benefits affect both experimentalists and theoreticians, but have been included only under one heading.

3.1 For theoreticians

1. Handling heterogeneity: The principles allow for each neuron to have a significantly different response function and tuning curve, as observed in real systems. Nevertheless, we can successfully quantify and analyze the system. We have argued at length that heterogeneity is essential for understanding the high quality of neural representation [2].

2. Predicting function: Given the formulation of transformation as linear decoding, for any neural population, singular value decomposition identifies an ordered orthogonal basis in which transformations of the information encoded by the population must lie. This allows us to put precise contraints on the functions computable by a given neural population.

3. Grounding abstract models: Because the principles support simulating systems at various level of description concurrently, they explicitly relate various levels of abstraction (e.g., spiking neurons, rate neurons, population behavior, etc.). This can fill in (often important) implementational details for high-level descriptions of neural dynamics. More practically, this is useful for fine-tuning a model before introducing the extra computational overhead involved with modeling individual neurons.

4. Negative weights problem: These principles can be used to solve the negative weights problem. That is, the problem of ensuring that there are no negative weights in a model, only inhibitory neurons (as observed in cortical circuits) [2].

5. Generalization of attractor networks: As demonstrated by the working memory and lamprey models, the principles allow for the easy construction of complex, spiking attractor networks. The working memory model shows how such networks can be built for storing large ensembles of functions (e.g., point, line, plane attractors for high dimensional representations). The lamprey model shows how such networks can be built for dynamic attractors (e.g., cyclic attractors).

3.2 For experimentalists

1. Flexible physiological plausibility: Because any neural-level nonlinearity can be used to define the neural encoding (spiking, non-spiking, simple or complex conductance models), as simulation can be made to match what is known about the system under study. As well, this flexibilty can be used to demonstrate what level of organization may be most relevant for the phenomena under study.

2. Generating experiments for completing models: As suggested by the previous benefit, constructing an explicit simulation makes it evident what information is most essential to ensuring the simulation is an accurate reflection of the system being modeled.

3. Limiting hypothesis space: As demonstrated by the vestibular model above, available evidence is not always sufficient to determine how the function understudy is implemented in a neural system. However, the techniques are useful, in such cases, for explicitly outlining the set of alternatives that are consistent with available data.

4. Straightforward application: In our extended discussions of this framework, we outline a simple *methodology* for employing these principles. As well, we have constructed a graphical simulation package in Matlab and Java based on the principles that simplifies the construction of large-scale spiking networks.

5. Generating predictions: Perhaps the central reason to model a system is to learn something new about it. It is difficult to conclusively prove that the principles will help generate predictions generally, so instead we provide examples. In the vestibular system model, the simulations highlighted reasons for specific distributions of NMDA and AMPA receptors in certain subpopulations of the model. In addition, constraints on the tuning curves and their distribution necessary to implement the observed function were evident from the resulting simulation. In the working memory model, there were clear cases where an *increase* in neural activity during a delay period should be expected (i.e., in neurons whose receptive fields are at the periphery of the target; these are seldom recorded from). As well, the model suggests that the kinds of error observed during multi-target delayed saccade or reaching tasks should fall into one of two categories.

4 Conclusion

Given these benefits, we would like to suggest that these principles can provide some of the needed theoretical structure in neuroscience. However, we also suppose that they are, in various ways, only a starting point. Determining the ways in which these principles fail, however, can help guide us to formulating better principles.

References

[1] Nature Neuroscience (2000). 3: Special issue on Computational Neuroscience. (See especially the series of 'Viewpoints' articles.)

[2] Eliasmith, C. and C. H. Anderson (2003). Neural engineering: Computation, representation, and dynamics in neurobiological systems. Cambridge, MA: MIT Press.

[3] Rieke, F., D. Warland, R. R. de Ruyter van Steveninick, and W. Bialek (1997). Spikes: Exploring the neural code. Cambridge, MA: MIT Press.

[4] Eliasmith, C. and C. H. Anderson (2001). Beyond bumps: Spiking networks that store sets of functions. Neurocomputing. 38:581–586.

[5] Eliasmith, C., M. B. Westover, and C. H. Anderson (2002). A general framework for neurobiological modeling: An application to the vestibular system. Neurocomputing. 46: 1071–1076

[6] Eliasmith, C. and C. H. Anderson (2000). Rethinking central pattern generators: A general approach. Neurocomputing 32, 735–740.