Biological Cybernetics (2019) 113:83–92
https://doi.org/10.1007/s00422-018-0777-7

Biological Cybernetics

ORIGINAL ARTICLE

A normative approach to neuromotor control

Max Berniker[1] · **Steven Penny**[1]

Received: 1 February 2018 / Accepted: 20 August 2018 / Published online: 3 September 2018

Abstract
While we can readily observe and model the dynamics of our limbs, analyzing the neurons that drive movement is not nearly as straightforward. As a result, their role in motor behavior (e.g., forward models, state estimators, controllers, etc.) remains elusive. Computational explanations of electrophysiological data often rely on firing rate models or deterministic spiking models. Yet neither can accurately describe the interactions of neurons that issue spikes, probabilistically. Here we take a normative approach by designing a probabilistic spiking network to implement LQR control for a limb model. We find typical results: cosine tuning curves, population vectors that correlate with reaching directions, low-dimensional oscillatory activity for reaches that have no oscillatory movement, and changes in neuron's tuning curves after force field adaptation. Importantly, while the model is consistent with these empirically derived correlations, we can also analyze it in terms of the known causal mechanism: an LQR controller and the probability distributions of the neurons that encode it. Redesigning the system under a different set of assumptions (e.g. a different controller, or network architecture) would yield a new set of testable predictions. We suggest this normative approach can be a framework for examining the motor system, providing testable links between observed neural activity and motor behavior.

Keywords Spiking neural network · Normative modeling · Motor control · Tuning curves

1 Introduction

Motor control is fundamentally about the interaction of our neurons and our body. That is, how do the coupled dynamics of our neurons and musculo-skeletal apparatus give rise to intelligent motor behavior? Unfortunately, a detailed analysis of these interactions is currently intractable—while relatively simple models of our limbs and muscles are adequate for describing reaching movements, the collective behavior of neurons is far more complex. The number of neurons involved in motor control and the complexity associated with describing their nonlinear stochastic interactions renders their analysis particularly difficult. In part because of this difficulty, a holistic neuromotor approach has been neglected in favor of research that examines motor behavior and neural activity largely independently. This parallel approach has been productive for decades, but lacking a precise description for how neural activity and motor behaviors are coupled, the results are often necessarily correlational.

A computational description of the causal relationship between neurons and the motor apparatus could verify and validate our current best theories. Recent work in this direction has described how neural activity may give rise to motor behaviors. Rate-based models, which substitute a time averaged value, or rate of firing, for actual spikes, have been used to describe a number of empirical and theoretical findings (Laje and Buonomano 2013; Sussillo and Abbott 2009; Mante et al. 2013; Sussillo and Barak 2013). A model that describes how neurons interact when exchanging continuously varying information, however, may not correctly describe the interactions of neurons that share binary information. This is true both for a deterministic setting, and for more realistic stochastic settings. Models of deterministically spiking neurons, on the other hand, can precisely describe the nonlinear interactions of a large number of neurons (Bourdoukan et al. 2012; Denève et al. 2017; Thalmeier et al. 2016). Yet, the validity and use of this approach under noisy settings is also uncertain. Ultimately, to examine the

Communicated by Manoj Srinivasan.

This article belongs to the Special Issue on *Control Theory in Biology and Medicine*. It derived from a workshop at the Mathematical Biosciences Institute, Ohio State University, Columbus, OH, USA.

✉ Max Berniker
mbernike@uic.edu

[1] University of Illinois at Chicago, Chicago, USA

relationship between neurons and movement, we will require models that couple the two through stochastic spiking events (e.g. DeWolf 2016).

Here we take a normative approach, and ask what kind of behavior might we expect from noisy spiking neurons that move a limb. By making some standard assumptions, we create a simple neuromotor model and examine its behavior under a common hypothesis: linear–quadratic regulator (LQR) control. Not surprisingly the resulting motor behavior is consistent with experimental findings, e.g. straight trajectories with positively skewed bell-shaped velocity profiles. More interesting however is the accompanying neural activity. Spike activity mirrors that of experimental features often found. Importantly, however, using our model, we can offer a normative interpretation of individual neural firing patterns in terms of each neuron's probability distribution, and the model's underlying controller hypothesis. That is, we can uncover the neural networks "black box" and explain why a given neuron fires. We suggest that a normative neuromotor approach to examining the motor system is advantageous, especially as more data and accurate models of the motor system emerge, allowing research to focus on high-level explanations for neural processes while remaining faithful to physiological constraints.

2 Methods

Rather than attempting to precisely model neurons and the motor apparatus, with a normative approach, our focus is on organizing hypotheses that can explain how these two systems interact and what testable predictions can be made. Therefore, we model the salient features that describe how the motor apparatus is coupled to neurons. We suggest the motor apparatus is approximately a point mass, driven by muscles that are activated through a network of neurons, which spike stochastically. Changes in the limb's state are then fed back to these neurons in the form of an error signal so that the network can update its commands to the muscles (Fig. 1). Our hypothesis is that the network of neurons is implementing an LQR controller, a commonly supported proposition for both upper and lower limbs (Kuo 1995; Scott 2004; Todorov 2004).

2.1 Musculo-skeletal model and control

To examine point-to-point reaches, the limb is modeled as a point mass that moves in a plane. For the x-axis, we have

$$m\dot{v}_x = -b\dot{x} + f_x^m \tag{1}$$

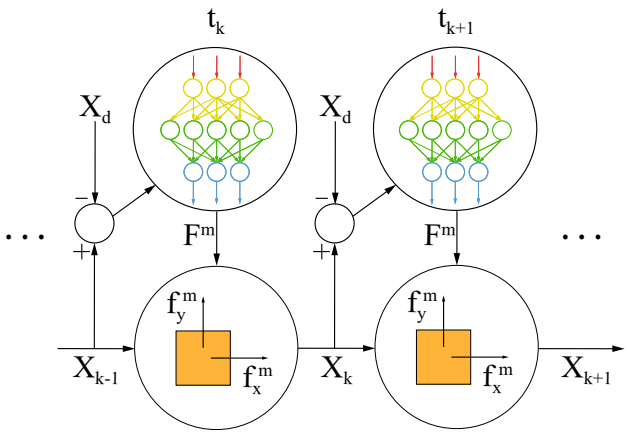

Fig. 1 The generative model for our neuromotor system. A feed-forward network of probabilistically firing neurons is used to generate muscle forces. These forces accelerate the limb model, changing its current state. The updated state error is fed back to the network, and a new muscle force is sampled

The y-component is modeled identically, where $m = 1$ kg, and $b = 0.1$ Ns/m. Analogous to real muscles, muscle force, f_x^m, is modeled as a linear sum of activated motor units,

$$f_x^m = \sum_i^n \alpha_i o_i^x \tag{2}$$

where α_i is the force of the ith motor unit and o_i is the corresponding drive to that unit, modeled as a spike (zero or one) from the network. In order to "pull" in both the positive and negative directions, half the α_i's were positive, and half negative (see below for more details).

In state space form, the limb dynamics are $\dot{X} = AX + BU$, where $X = [x, y, v_x, v_y]^T$ and $U = F^m = [f_x^m, f_y^m]^T$. Our hypothesis is that control of the limb is approximating a linear–quadratic regulator, driving the limb to a desired location (X_d) in the workspace, by minimizing the following,

$$L_\mathrm{lqr} = \int_0^\infty \frac{1}{2}(\Delta X^T Q \Delta X + U^T R U)\mathrm{d}t \tag{3}$$

where $\Delta X = X - X_\mathrm{d}$. The resulting optimal control signal can be found from the algebraic Riccati equation, $U_\mathrm{lqr} = -K\Delta X$. For the results shown here, $Q = \mathrm{diag}([100, 100, 1, 1])$, and $R = \mathrm{diag}([0.1, 0.1])$. By examining a large number of random reaches over a workspace 1×1 m, we defined the following range of states, $-.5\,\mathrm{m} \le x, y \le .5\,\mathrm{m}$, and $-2.5\,\mathrm{m/s} \le v_x, v_y \le 2.5\,\mathrm{m/s}$, and commands $-30\,\mathrm{N} \le f_x^m, f_y^m \le 30\,\mathrm{N}$. We note that this optimal feedback control law is smooth, continuous, and deterministic, whereas our network can only output discrete force values (according to the sum in Eq. 2), and does so stochastically.

2.2 Spiking network

In rough approximation, neurons appear to transmit information through discrete spikes, in a probabilistic fashion, based on the spikes they receive from other neurons. These discrete events render the relationship between incoming and outgoing spikes nonlinear, and an accompanying computational description of large numbers of neurons particularly challenging. However, with two important simplifications, we can obtain a model with an analytic solution for each neuron's probability distribution.

First, we assume each neuron's probability of emitting a spike is only a function of the current incoming spikes, and not the history of incoming or outgoing spikes; that is, our neurons have no dynamics. Second, we assume the network only has feed-forward connections. While this limits the potential distributions the network can represent, this significantly simplifies the required calculations (Jordan et al. 1999). With these two assumptions, our network of spiking neurons can be modeled as a sigmoidal belief network (Neal 1990) with a single hidden layer of neurons, $H = [h_1, h_2, ..., h_m]$ and an output layer, $O = [o_1, o_2, ..., o_n]$. Parameterized by the hidden and output layer weights, $\{W^1, B^1\}$ and $\{W^2, B^2\}$, respectively, the probability of any neuron firing is given by

$$p(h_i = 1|\Delta X) = \sigma\left(\sum_j w_{ij}^1 \Delta x_i + b_i^1\right) \tag{4}$$

$$p(o_i = 1|H) = \sigma\left(\sum_j w_{ij}^2 h_j + b_i^2\right) \tag{5}$$

where $\sigma(\cdot)$ is the standard sigmoidal activation function. We can compute the exact probability that any output neuron in the network spikes by marginalizing over all the $2^m = M$ permutations of the hidden neurons.

$$p(o_i = 1|\Delta X) = \sum_v^M p(o_i = 1|H^v)p(H^v|\Delta X) \tag{6}$$

where

$$p(H|\Delta X) = \prod_i^m p(h_i|\Delta X) \tag{7}$$

For all the results shown here, the network had 12 output units, 6 for producing forces in the x-axis (3 positive and 3 negative units), and 6 for the y-axis. The α values were obtained by numerically minimizing the error between the discrete values F^m could sum to, and the continuous force values desired. The rule, $\alpha_{j+1} = 2\alpha_j$, minimized these errors and α was set to $[-17.1, -8.6, -4.3, 4.3, 8.6, 17.1]$, allowing for 15 possible unique force values, and a maximum of 30 Newtons in either axis.

Since the optimal LQR muscle forces are only a function of the input error signal, ΔX, and conditionally independent of each other, networks for x- and y-component muscle forces were trained separately and then combined to perform the simulations. The resulting network had a total of 40 neurons (see Sect. 3).

To build the LQR controller with our spiking network, we designed the initial weights to achieve an approximate solution (see Sect. 3), and then fine-tuned the network by optimizing a cost function that penalizes the network's expected deviations from the desired probability. To do so, we note that for any state input, ΔX, there is a corresponding muscle force, F^m, that most closely approximates $-K\Delta X$. This F^m is obtained with a unique set of output neuron spikes, O. We defined a desired probability, $\mu(\Delta X)$ such that $\mu_i(\Delta X) = .99$ if $o_i = 1$ and $\mu_i(\Delta X) = .01$ if $o_i = 0$. Then we defined the following cost,

$$L_k = \frac{1}{2n}(\mu(\Delta X^k) - \bar{O}^k)^T(\mu(\Delta X^k) - \bar{O}^k) \tag{8}$$

$$L = \sum_k^N L_k + \lambda \sum_{i,j,p,q} \left[(w_{ij}^1)^2 + (w_{pq}^2)^2\right] \tag{9}$$

where $\bar{O}^k$ is the expected output conditioned on the state, ΔX^k, according to Eq. 6, and λ = 1E–6, is a regularizing term and the sum is over training data $\{\Delta X^k, \mu(\Delta X^k)\}_{k=1}^N$. Since the exact gradients for our network are readily computable (although numerically expensive for large networks, see "Appendix"), fine-tuning the network with this cost was straightforward.

2.3 Simulated experiments and data analysis

We can sample muscle forces from the network by providing an input, ΔX, then drawing spikes from the hidden layer according to its probabilities (Eq. 4), and then drawing spikes from the output layer according to its probabilities (Eq. 5) and computing F^m (Eq. 2). With a sample force, the limb's dynamics could be discretely updated to the next time step ($\Delta t = 5$ ms), and the process repeated. Point-to-point reaches were simulated by holding $X_d = X(t = 0)$ for 250 milliseconds, and then switching X_d to the desired state value and integrating the equations forward for another second. This resulted in time-varying trajectories for the limb, along with a spike train history for each neuron.

With our simulated data, we performed conventional statistical analyses, including computing peristimulus time

histograms (PSTH), cosine tuning curves, and population vectors. PSTHs were created using a 25 millisecond time bin, aligning spike data at time t=0, and averaging across trials. Cosine tuning curves were found using a least squares fit of the parameters A, B and C from the equation $y = A + B\sin(\theta) + C\cos(\theta)$, where y is the firing rate of a neuron for a specific reach angle, θ, and the resulting preferred direction $\theta_p = \tan^{-1}(B/C)$. Using these tuning parameters, we computed the population vectors as the vector sum of each neuron's preferred direction scaled by its firing rate. In addition, we performed a dynamical systems analysis as described in Churchland et al. (2012), which searches the neural firing rate data for a low-dimensional subspace that exhibits oscillatory behavior.

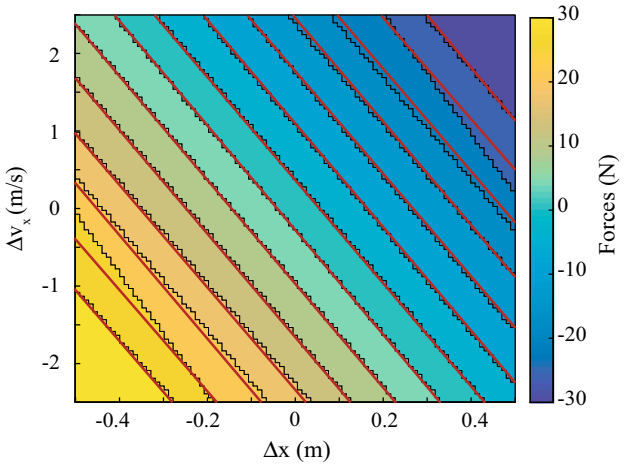

Fig. 2 To design the network weights the input space (shown here as x and v_x) is broken into linearly separable regions. The ideal decision boundary for the 15 force levels of an LQR controller is shown in red. These boundaries are then used to initialize the hidden neuron's weights such that the boundaries align with the hidden neuron's probability of $p = 0.5$. After training, these boundaries are altered (black lines)

3 Results

3.1 A spiking network controller

We designed a network to accurately encode the linear LQR control law, $U = -K\Delta X$, while limited to discrete force values (15 for the $x-$ and y-axis each, see Sect. 2). Note that an ideal function would achieve this by mapping arbitrary inputs to the nearest discrete muscle force value. Thus, the input space would effectively be categorized into k linearly separable regions if there were k discrete muscle forces. This is precisely what the hidden layer of a sigmoidal network does; regions of the input space separated by the probability, $p = 0.5$ of eliciting a spike (the line obeying $\sum w_{ij}^1 \Delta x_j + b_i^1 = 0$), are effectively decision boundaries, determining whether a spike is likely or not. In other words, the hidden layer acts like a common flash analog-to-digital converter (Walden 1999). As such, the hidden layer requires $k - 1$ neurons to precisely encode the k discrete input space regions (see Fig. 2).

Note that since our muscle units contain identical positive and negative values, for n output units the network can only generate $k = 2^{n/2+1} - 1$ unique forces, not 2^n. Recognizing this, the hidden layer's weights can be chosen to create these decision boundaries by setting $W^1 = K$, and B^1 to the values of 14 linearly spaced forces between -30 and 30 N (see Fig. 2). Therefore, in total, the network had 40 neurons: 28 in the hidden layer, and 12 in the output layer.

Determining the output layer's weights was similarly straightforward. Each of the 15 discrete force values is generated from a set of output spikes. These force values also correspond to inputs from a unique region of the input, with an associated expected hidden layer activity. The output weights were initially chosen to map these expected hidden layer spikes to values that would increase the probability of the appropriate forces.

To explain, let $\bar{H}^i$ be the expected hidden neuron values corresponding to the ith discrete muscle force value, and μ^{i*} be the desired expected output for this same force value. $\mu^{i*} = E[O^i] = \sigma(W^2\bar{H}_i + B^2)$ is composed of terms approximately zero and one, so the argument must correspond to large positive and negative numbers (which we arbitrarily chose as ± 3). Therefore for each of the discrete force values, and associated hidden and output values, we assume $W^2\bar{H}_i + B^2 = 6\mu^{i*} + 3I_n$, and solve a linear set of equations for W^2 and B^2. With these designed initial weights, the network approximately represented the LQR controller. To fine-tune these weights, we performed gradient descent on the expected error in the output probabilities (see Methods, Appendix).

With the network complete, muscle forces, obtained by sampling over a probabilistically firing group of neurons, are noisy and discrete valued, and may vary from the desired force (Fig. 3a). However, we can analytically compute the expected value, which varies continuously, and variance of a muscle force conditioned on the current input. For unperturbed movements, f_x^m is only a function of deviations in x and v_x, so we can depict the expected muscle force, the expected error in muscle forces and its variance as a surface (Fig. 3b–d). Although the network can only generate 15 discrete force values over a range of 60 Newtons, the expected error in f_x^m averaged over the workspace was 0.45 ± 0.72 Newtons (mean and standard error of data in Fig. 3c), while the analytically derived standard deviation in the expected error, averaged over the workspace was 3.14 ± 0.23 Newtons (mean and standard error of data in Fig. 3d).

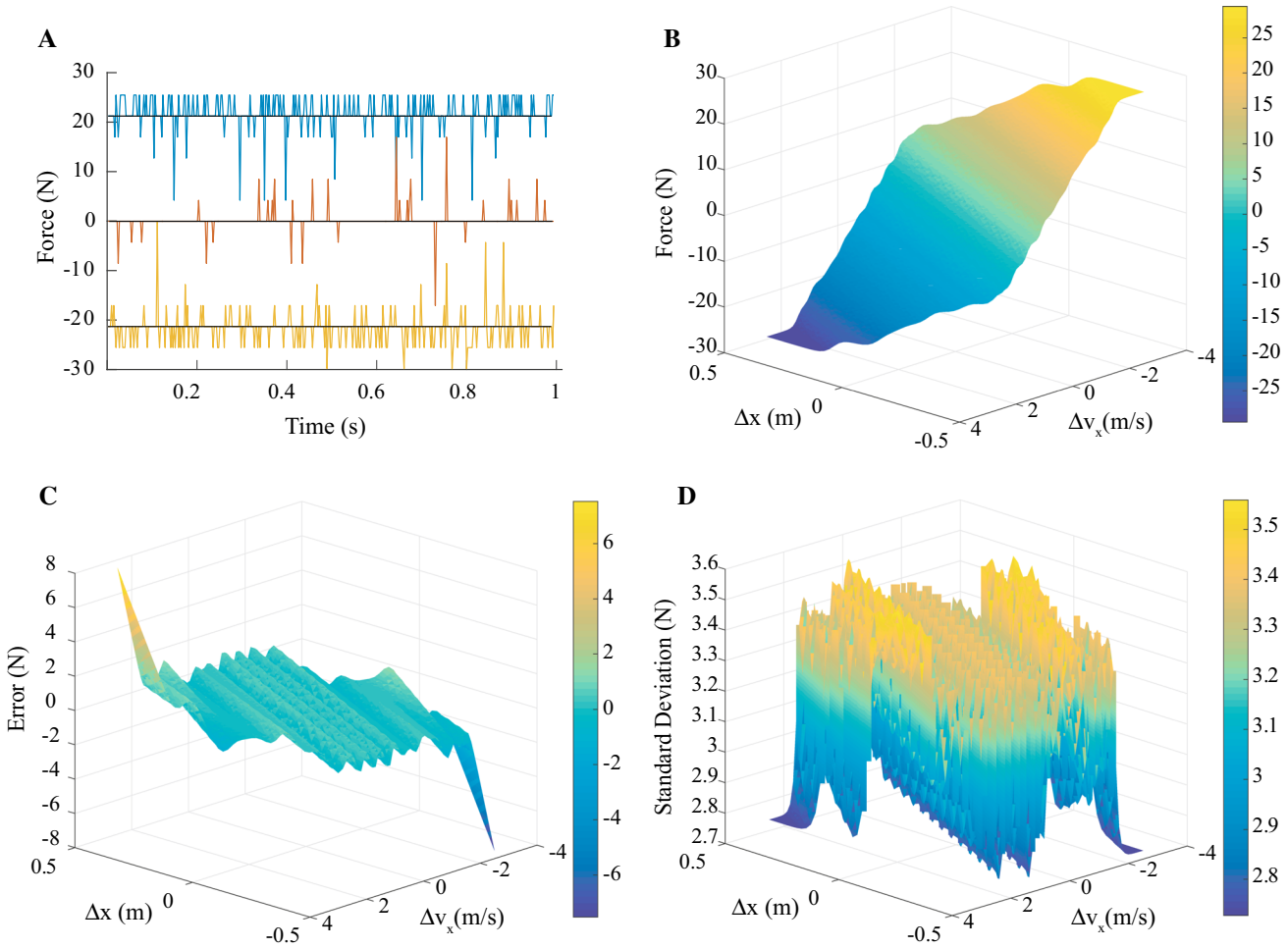

Fig. 3 **a** Sample forces obtained from the network. The input is held constant for 1 second and muscle forces are sampled every 5 ms. The desired output forces are 21.4, 0 and −21.4 Newtons (black lines). **b** The expected forces the network generates over the workspace in the x-component (y-component is identical). **c**, **d** The expected error between the network and an LQR controller and standard deviation in muscle forces can also be computed analytically across the limb's entire workspace

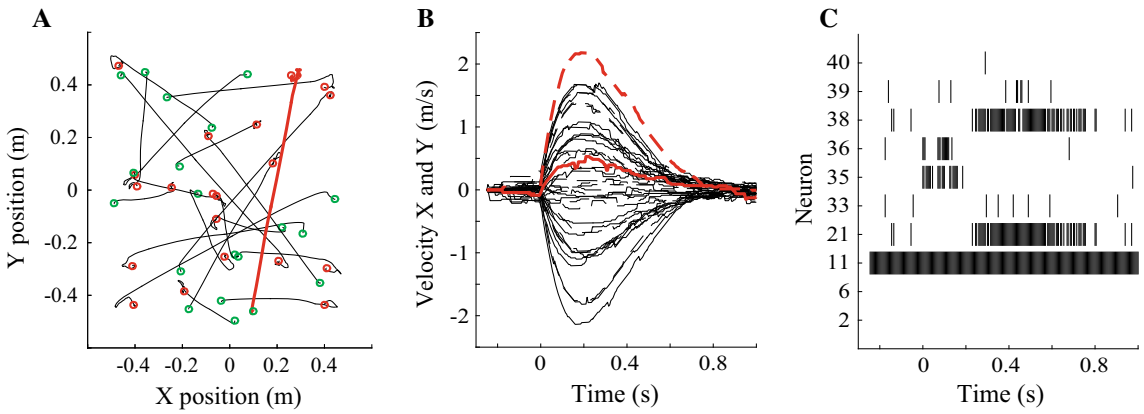

Fig. 4 **a** Randomly generated reaches with the neuromotor control model. Each reach starts at a green circle and ends at a red circle. In red, one such example reach is shown. **b** The corresponding velocities of these reaches exhibit roughly bell-shaped profiles. The x- and y-components are shown in solid and dashed lines, respectively. In red are the velocity profiles for example reach. **c** For each simulated reach, there is an associated history of neural activity. A raster plots is shown for 10 illustrative neurons for example reach in red. Neuron indices are arbitrary labels in the network with indices 1–28 corresponding to hidden units and indices 29–40 corresponding to output, muscle units

3.2 Neuromotor model dynamics

To build our neuromotor model, we couple the network with the limb model: the current state error is fed into the network, which generates muscle forces that accelerate the limb (Fig. 1). To simulate movement, the limb's state is integrated forward in time discretely, at each time step getting a randomly sampled muscle force from the network using the previous time step's state information. To illustrate, 20 random reaches were simulated (Fig. 4a). As was expected, reaches exhibit realistic characteristics: straight paths with approximately bell-shaped velocity profiles (Fig. 4b). As with experimental data, speed profiles are positively skewed (Mutha and Sainburg 2007; Corcos et al. 1989; Nagasaki 1989). Although movement times are not specified, they generally reached their target within approximately 0.8 seconds.

In addition to generating movement, the neuromotor model also generates spike trains, predicting a variety of patterns (Fig. 4c). For a given reach, some neurons are largely silent, others are tonically active and still others display phasic activity. Ignoring our normative understanding of the neuromotor model, we could analyze its spike train data at both the individual and population level as in conventional animal model experiments.

3.3 Virtual experiment and analysis

To model a typical electrophysiological study, e.g. (Georgopoulos 1982; Li et al. 2001; Churchland et al. 2012), we simulated center-out point-to-point reaches to eight evenly spaced targets, 50 times each for a total of 400 trials (see Fig. 5a, b). With this data, we obtained average spike rates and other commonly computed neural statistics.

As is seen experimentally, during these center-out reaches, half of our neurons were largely silent, and did not have firing rates significantly greater than zero (Georgopoulos 1982). Of those neurons that were active, most had phasic activity with clear signs for a preferred reach direction (Fig. 5c–h). We can quantify this preference with a cosine tuning curve. Of the 16 neurons that were active, 11 showed significant tuning ($R^2 > 0.5$, see Fig. 5c, d). Neurons with the largest tuning had a maximum of $R^2 = 0.873$, comparable to (Amirikian and Georgopoulos 2000). These results are consistent with the idea that individual neurons can be encoding either the actual, or a desired reach direction (Georgopoulos 1982; Georgopoulos et al. 1986).

We then used this cosine tuning to construct population vectors: computed as the vector sum of each neuron's preferred direction and scaled by its firing rate. We find, as has been found experimentally (Georgopoulos et al. 1986), these population vectors point in the direction of movement. Throughout the duration of the reach, the population vector for all eight directions consistently points toward the desired target and the direction of movement (Fig 6). The error between the mean population vector and reaching direction was $0.53° \pm 2.5°$. Thus, spike data from our model appear to accurately encode movement direction.

Do neural populations encode limb dynamics in their firing patterns? Interestingly, oscillatory firing rates have been observed in point-to-point movements, where there is no cyclic motion. This has led to the suggestion that the time-varying firing rates obtained during locomotion and reaching movements may reflect a model of the controlled limb's dynamics (Churchland et al. 2012). Performing the identical analysis as done in Churchland et al. (2012), we too find oscillatory firing rate trajectories for both our center-out reaches (Fig. 7a) as well as 100 random reaches (Fig. 7b). These results are interesting in that they appear to suggest cyclic dynamics, whereas the hand's movement during these point-to-point reaches are not. Thus, the simulated data of our normative neuromotor model are consistent with the hypothesis that neurons are encoding the dynamics of movement.

With our neuromotor model, we can also examine basic forms of motor learning. We trained our network to make reaches in a force field. The original network weights are re-trained to implement an LQR feedback controller for a velocity-dependent clockwise curl field. We examined how each neuron's tuning curve changed after learning the force field controller. Overall the average change in preferred direction was 25.94^o, thus rotating into the direction of the field. Interestingly, we find that while some neurons' preferred directions have changed, others have not. This too is consistent with experimental evidence (Li et al. 2001; Padoa-Schioppa et al. 2004), and has been suggested that neurons whose tuning properties do not change represent memory or kinematic cells (Li et al. 2001), since the kinematics of the movement before and after the reaches have largely remained invariant. On the other hand, neurons whose tuning properties have changed are dynamic cells (Li et al. 2001), since the forces required to make these reaches have been altered.

3.4 Normative explanation

With regard to the behavioral results, it has been well established that control of reaches can be closely approximated to LQR optimally controlled reaches: a straight trajectory and roughly bell-shaped velocity profile are attained when minimizing the effort of landing on target accurately. Thus, our use of an LQR controller for evaluating motor control is a recapitulation of a long history of research, and the behavioral findings were largely expected.

The findings on neural activity, however, were unanticipated. With our normative approach, we can uncover the "black box" of the network and analyze when, and why each neuron fires to understand the population as a whole. First we note that despite the evidence, our neurons are not encoding

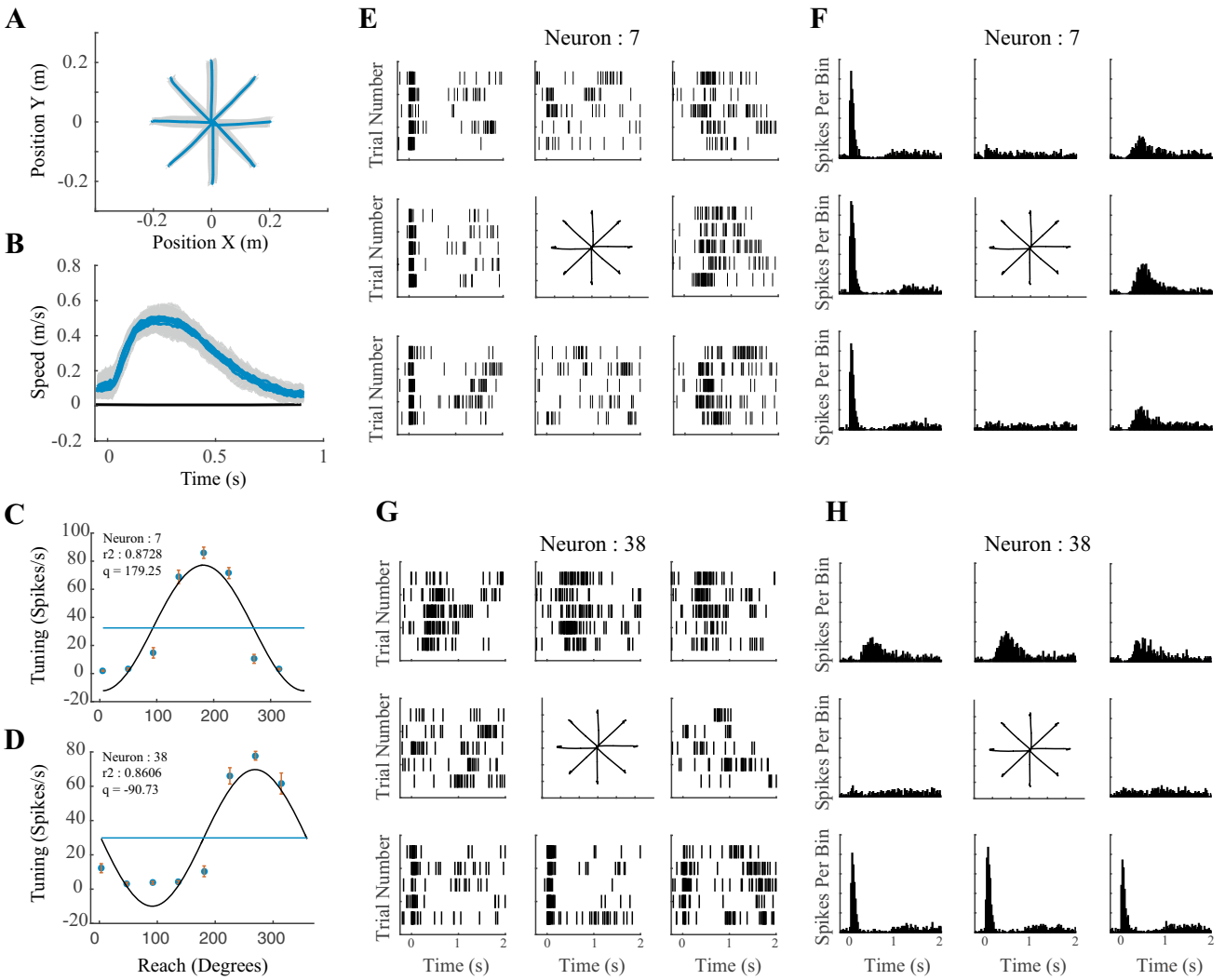

Fig. 5 **a**, **b** Average paths and speed profiles for the simulated 8 center-out reach directions. **c**, **d** Two neuron's tuning curves reveal a strong correlation with reach direction. **e**, **g**, Raster plots for the same two neurons are displayed for each of the 8 reach directions. They reveal phasic activity that initially increases during leftwards and downwards reaches, respectively. **f**, **h** PSTH's for the same neurons

any of the experimentally motivated descriptions described above (i.e., tuning curves and limb dynamics). These features are in fact emergent properties. The neurons collectively encode the linear mapping from state errors to motor units that results in the LQR controller.

As explained above, the hidden layer probabilistically encodes errors in the state. The neurons of the output layer use the spikes from the hidden layer to probabilistically command appropriate motor units and generate forces. Much like tuning curves, we can visualize these probabilities by looking at how they change as we vary errors in position and velocity (Fig. 8). While the hidden neurons only depend on these state errors (Fig. 8c), we can compute the probability of an output neuron spiking conditioned on state errors by integrating out their dependence on the hidden states (Fig. 8a, b). From these plots, we see how movement across the workspace results in activity that is correlated with reach direction and velocity. For example, some neurons fire with higher probability when the hand is to the left of the target (Fig. 8d), some to the right (Fig. 8e), whereas some are silent during these movements and only fire during larger speeds and positional errors (Fig. 8f).

Finally, with regard to the oscillatory components of the networks firing rates, we see that this behavior is due to the dynamics of the limb itself. Not only does the network not encode dynamics, it has no dynamics itself. The firing rate trajectories are a consequence of the coupling of the limb with the network. In particular, the velocity of the limb beginning and ending at zero gives rise to a subspace that appears oscillatory.

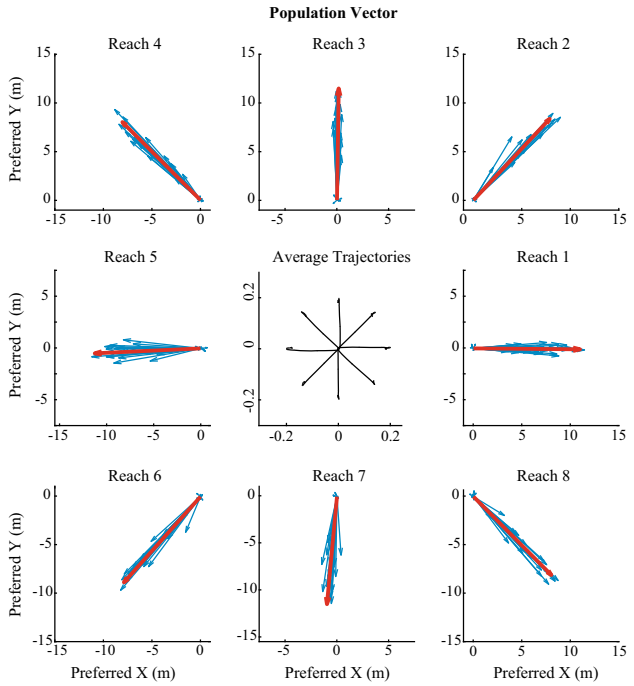

Fig. 6 The population vector evolving over the duration of the reach. Each blue arrow is the population vector at 5 milliseconds increments. The red arrows are the population vector averaged over the entire reach

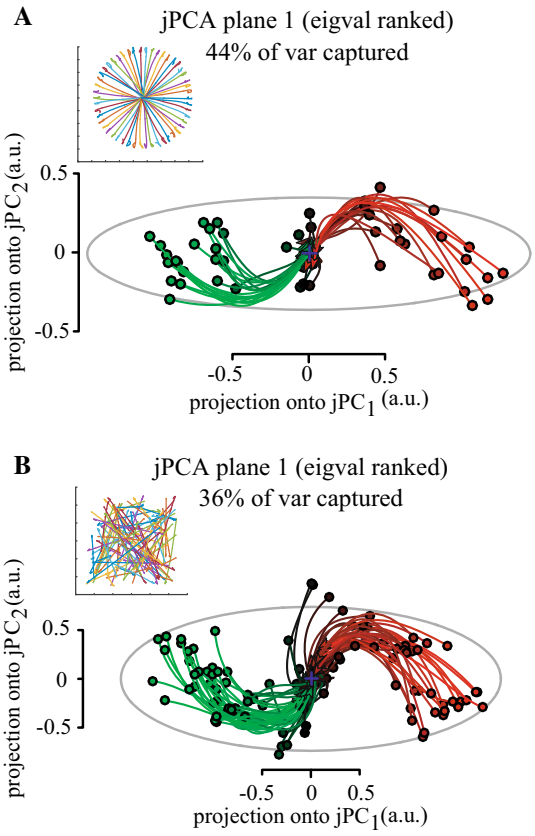

Fig. 7 A dynamical systems analysis of average firing rates similar to that presented in Churchland et al. (2012). Even though center-out (**a**) and random (**b**) reaches are not cyclic, a low-dimensional projection of their activity appears to be. Trajectories are color coded by their initial value in the first jPC (green to red are negative to positive, respectively). Inserts represent the movements that generated these low-dimensional trajectories

4 Discussion

Here we have argued for a holistic neuromotor approach to examining the motor system, combining known features of both limb biomechanics and the neurons that drive it. Taking a normative approach, we have designed a probabilistic network of spiking neurons to control movement of the limb. By design, the resulting reaches were largely consistent with known features of human and non-human primate movements. And while not designed as such, the spike data was also consistent with many statistical correlations observed experimentally. Importantly, we can analyze the causal mechanisms of our model to explain these correlations.

Our network model is relatively simple, with a small number (40) of neurons that lack dynamics. As such, the network model cannot duplicate many, perhaps important, features of biological neurons. There is the very real possibility that further biological realism is required before a model such as this can accurately describe even the function of a network of neurons, let alone the precise interactions. There is a long history, however, of seemingly simple and abstract models that not only exhibit complex behavior, but also yield important insight into the working of brains (Rumelhart et al. 1985; Le Quoc 2012). We emphasize that while we attempt to remain faithful to neurons when possible, our focus is on understanding their behavior at a functional level.

In our simulated reaching experiment, we found strong correlations with reach direction. This was because our model was encoding the forces needed to drive the limb to a desired target. Of course real limbs do not translate in x and y, but must rotate to acquire a target. Future versions of our neuromotor model can examine if a 2-link limb model exhibits these same correlations. This would provide useful predictions for whether or not an explicit representation of the hand in Cartesian-like coordinates is necessary to explain the tuning found experimentally.

Dynamics and computations based on discrete, all-or-none events are inherently nonlinear. As such it is not clear how rate-based models, which assume neurons share continuously varying spike rate information, can accurately describe how individual neurons encode information. This criticism only increases under stochastic conditions. To be clear, rate-based models may offer informative computational descriptions of how information is processed by networks of neurons, but it is not clear that they can accurately

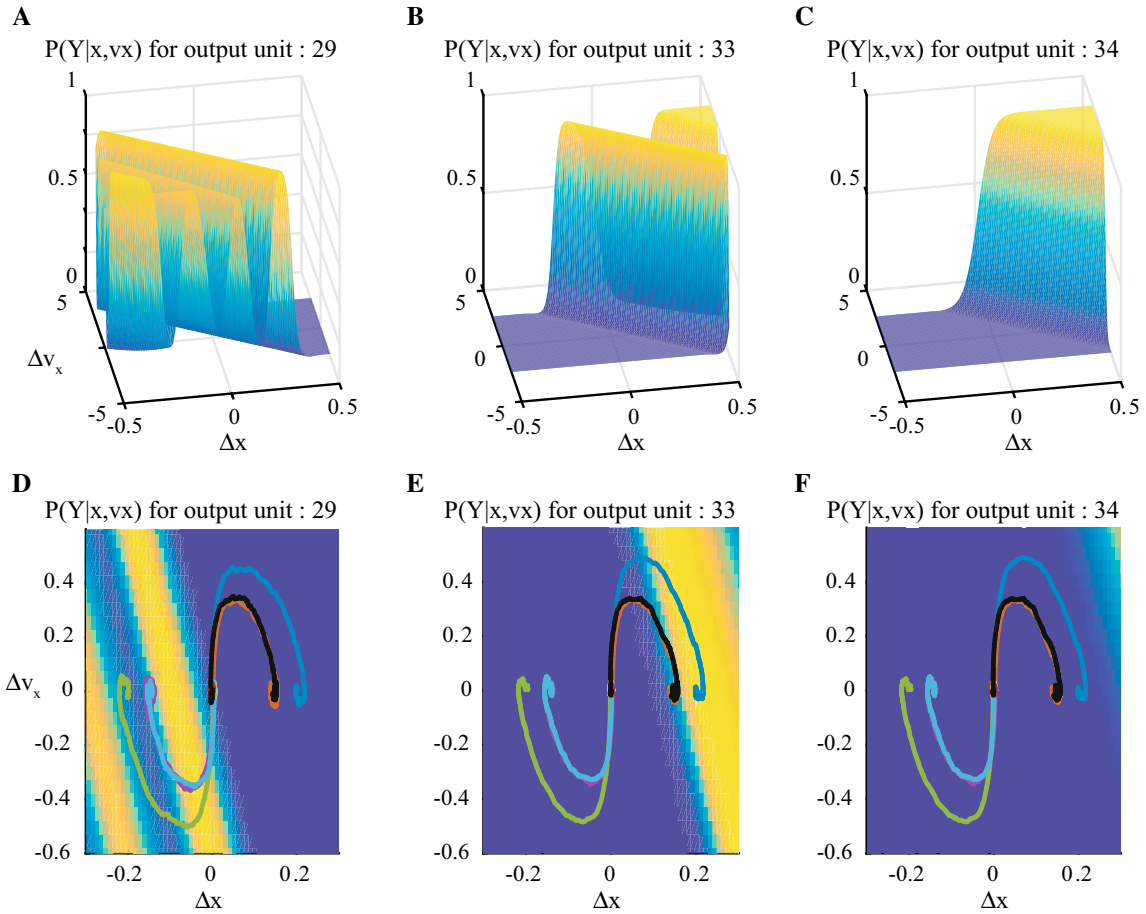

Fig. 8 **a**, **b**, **c** The probability of firing conditioned on errors in x and v_x for three of the twelve output neurons. The non-trivial shape is the result of designing the neurons to activate motor units that precisely approximate the LQR control signal. **d**, **e**, **f**, The x and v_x components of the average center-out reaches projected onto the same distributions shows how the neurons correlate with reach directions to the right and left or remain silent, respectively

account for how neurons themselves compute this information.

Similarly, the understanding we derive from a deterministic model of spiking neurons may break down under stochastic conditions. While neurons tend to spike reliably after receiving appropriate excitatory post-synaptic potentials, these spikes only result in probabilistic synaptic transmission (Moreno-Bote 2014). Until we have at least a functional understanding of the variability in spikes, they are effectively random events. Our model is motivated by this inherent stochasticity and the need to model the sharing of spikes, not rates.

Neural networks are extremely versatile tools, as evidenced by the often-stated description, "the second best way to solve any problem." The disadvantage however is that they are often, either unintentionally or by design, black boxes that offer no easy interpretation of how they solve the problem at hand. With our normative approach, we hope to mitigate this weakness by using them as a tool to implement our solution, rather than solve our problem.

To illustrate this, consider that while we have examined a relatively simple problem, a linear feedback controller, future studies may rely on similarly normative approaches to study more complex problems. For example, in biological systems state feedback is delayed and noisy, undermining the utility of feedback control. A feed-forward controller could be obtained with multiple networks that implement distinct processes, such as a forward model and state estimator. Each network would then be designed to solve a specific function and result in its own function-specific predictions for neural activity. Future work shall examine if this modular approach helps shed light on neural function while offering testable predictions for the neuromotor system.

Appendix

The gradient for the cost function in Eq. 8 is as follows,

$$L = \sum_{k}^{N} L_k + \lambda \sum_{i,j,p,q} \left[(w_{ij}^1)^2 + (w_{pq}^2)^2 \right] \tag{10}$$

$$\begin{aligned} L_k &= \frac{1}{2n} (\mu(\Delta X^k) - \bar{O}^k)^T (\mu(\Delta X^k) - \bar{O}^k) \\ &= \frac{1}{2n} \sum_{l}^{n} (\mu_l(\Delta X^k) - \bar{o}_l^k)^2 \end{aligned} \tag{11}$$

where

$$\begin{aligned} \bar{o}_l^k &= p(o_l = 1 | \Delta X^k) \\ &= \sum_{v}^{M} p(o_l = 1 | H^v) \prod_{i}^{m} p(h_i | \Delta X^k) \end{aligned} \tag{12}$$

then we find the following,

$$\begin{aligned} \frac{\partial L_k}{\partial W_{pq}^1} = \frac{-1}{n} \sum_{l}^{n} [\mu_l(\Delta X_k) - \bar{o}_l] \Bigg(\sum_{v}^{M} p(o_l = 1 | h^v) \ldots \\ \prod_{j \neq p}^{m} p(h_j^v | x) dp(h_p^v | x) x_q \Bigg) \end{aligned} \tag{13}$$

$$\begin{aligned} \frac{\partial L_k}{\partial B_{pq}^1} = \frac{-1}{n} \sum_{l}^{n} [\mu_l(\Delta X_k) - \bar{o}_l] \Bigg(\sum_{v}^{M} p(o_l = 1 | h^v) \ldots \\ \prod_{j \neq p}^{m} p(h_j^v | x) dp(h_p^v | x) \Bigg) \end{aligned} \tag{14}$$

$$\begin{aligned} \frac{\partial L_k}{\partial W_{pq}^2} = \frac{-1}{n} [\mu_p(\Delta X_k) - \bar{o}_p)] \Bigg(\sum_{v}^{M} dp(o_p = 1 | h^v) \ldots \\ \prod_{j}^{m} p(h_j^v | x) h_q^v \Bigg) \end{aligned} \tag{15}$$

$$\begin{aligned} \frac{\partial L_k}{\partial B_{pq}^2} = \frac{-1}{n} [\mu_p(\Delta X_k) - \bar{o}_p)] \Bigg(\sum_{v}^{M} dp(o_p = 1 | h^v) \ldots \\ \prod_{j}^{m} p(h_j^v | x) \Bigg) \end{aligned} \tag{16}$$

References

Amirikian B, Georgopoulos AP (2000) Directional tuning profiles of motor cortical cells. Neruosci Res 36(1):73–79

Bourdoukan R, Barrett D, Denève S, Machens CK (2012) Learning optimal spike-based representations. In: Advances in neural information processing systems, pp 2285–2293

Churchland MM, Cunningham JP, Kaufman MT, Foster JD, Nuyujukian P, Ryu SI, Shenoy KV (2012) Neural population dynamics during reaching. Nature 487:51

Corcos DM, Gottlieb GL, Agarwal GC (1989) Organizing principles for single-joint movements. II. A speed-sensitive strategy. J Neurophysiol 62(2):358–368

Denève S, Alemi A, Bourdoukan R (2017) The brain as an efficient and robust adaptive learner. Neuron 94:969–977

DeWolf T (2016) A spiking neural model of adaptive arm control. Proc R Soc B 283:20162134

Georgopoulos AP (1982) On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. J Neurosci 2:1527–1537

Georgopoulos AP, Schwartz AB, Kettner RE (1986) Neuronal population coding of movement direction. Science 233:1416–1419

Jordan MI, Gharamani Z, Jaakkola TS, Saul LK (1999) An introduction to variational methods for graphical models. Mach Learn 37:183–223

Kuo AD (1995) An optimal control model for analyzing human postural balance. IEEE Trans Biomed Eng 42:87–101

Laje R, Buonomano DV (2013) Robust timing and motor patterns by taming chaos in recurrent neural networks. Nat Neurosci 16(7):925

Le Quoc V (2012) Building high-level features using large scale unsupervised learning. In: IEEE international conference on acoustics, speech and signal processing (ICASSP)

Li CSR, Padoa-Schioppa C, Bizzi E (2001) Neuronal correlates of motor performance and motor learning in the primary motor cortex of monkeys adapting to an external force field. Neuron 30:593–607

Mante V, Sussillo D, Shenoy KV, Newsome WT (2013) Context-dependent computation by recurrent dynamics in prefrontal cortex. Nature 503:78–84

Moreno-Bote R (2014) Poisson-like spiking in circuits with probabilistic synapses. PLoS Comput Biol 10:1003522

Mutha PK, Sainburg RL (2007) Control of velocity and position in single joint movements. Hum Mov Sci 26(6):808–823

Nagasaki H (1989) Asymmetric velocity and acceleration profiles of human arm movements. Exp Brain Res 74(2):319–326

Neal RM (1990) Learning stochastic feedforward networks. Technical Report. CRG-TR-90-7, Connectionist Research Group, Department of Computer Science, University of Toronto, Ontario

Padoa-Schioppa C, Li CSR, Bizzi E (2004) Neuronal activity in the supplementary motor area of monkeys adapting to a new dynamic environment. J Neurophysiol 91:449–473

Rumelhart DE, Hinton GE, Williams RJ (1985) Learning internal representations by error propagation (No. ICS-8506). California Univ, San Diego, La Jolla Inst for Cognitive Science

Scott SH (2004) Optimal feedback control and the neural basis of volitional motor control. Nat Rev Neurosci 5:532

Sussillo D, Abbott LF (2009) Generating coherent patterns of activity from chaotic neural networks. Neuron 63:544–557

Sussillo D, Barak O (2013) Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. Neural Comput 25:626–649

Thalmeier D, Uhlmann M, Kappen HJ, Memmesheimer RM (2016) Learning universal computations with spikes. PLoS Comput Biol 12(6):e1004895

Todorov E (2004) Optimality principles in sensorimotor control. Nat Neurosci 7:907

Walden RH (1999) Analog-to-digital converter survey and analysis. IEEE J Sel Areas Commun 17(4):539–550