CHAPTER

Neural Decoding for Motor and Communication Prostheses

7

Byron M. Yu[*†**], **Gopal Santhanam**[*], **Maneesh Sahani**[**], **Krishna V. Shenoy**[*†***]

[*]*Department of Electrical Engineering,* [†]*Neurosciences Program, Stanford University, Stanford, California,* [**]*Gatsby Computational Neuroscience Unit, UCL, London,* [***]*Department of Bioengineering, Stanford University*

7.1 INTRODUCTION

Neural prostheses, which are also termed brain–machine and brain–computer interfaces, offer the potential to substantially increase the quality of life for people suffering from motor disorders, including paralysis and amputation. Such devices translate electrical neural activity from the brain into control signals for guiding paralyzed upper limbs, prosthetic arms, and computer cursors. Several research groups have now demonstrated that monkeys (e.g., Serruya et al., 2002; Taylor et al., 2002; Carmena et al., 2003; Musallam et al., 2004; Santhanam et al., 2006; Mulliken et al., 2008; Velliste et al., 2008) and humans (e.g., Kennedy et al., 2000a; Leuthardt et al., 2004; Wolpaw and McFarland, 2004b; Hochberg et al., 2006; Kim et al., 2008; Schalk et al., 2008) can learn to move computer cursors and robotic arms to various target locations simply by activating neural populations that participate in natural arm movements.

Figure 7.1 illustrates the basic operating principle of neural prosthetic systems. Neural activity from various arm movement–related brain regions (e.g., Wise et al., 1997) is electronically processed to create control signals for enacting the desired movement. Non invasive sensors can collect neural signals representing the average activity of many neurons. When invasive permanently implanted arrays of electrodes are employed, as depicted, it is possible to use waveform shape differences to discriminate individual neurons (e.g., Lewicki, 1998; Santhanam et al., 2004). After determining how each neuron responds before and during a movement, which is typically accomplished by correlating arm movements made during a behavioral task with

Statistical Signal Processing for Neuroscience and Neurotechnology. DOI: 10.1016/B978-0-12-375027-3.00007-7

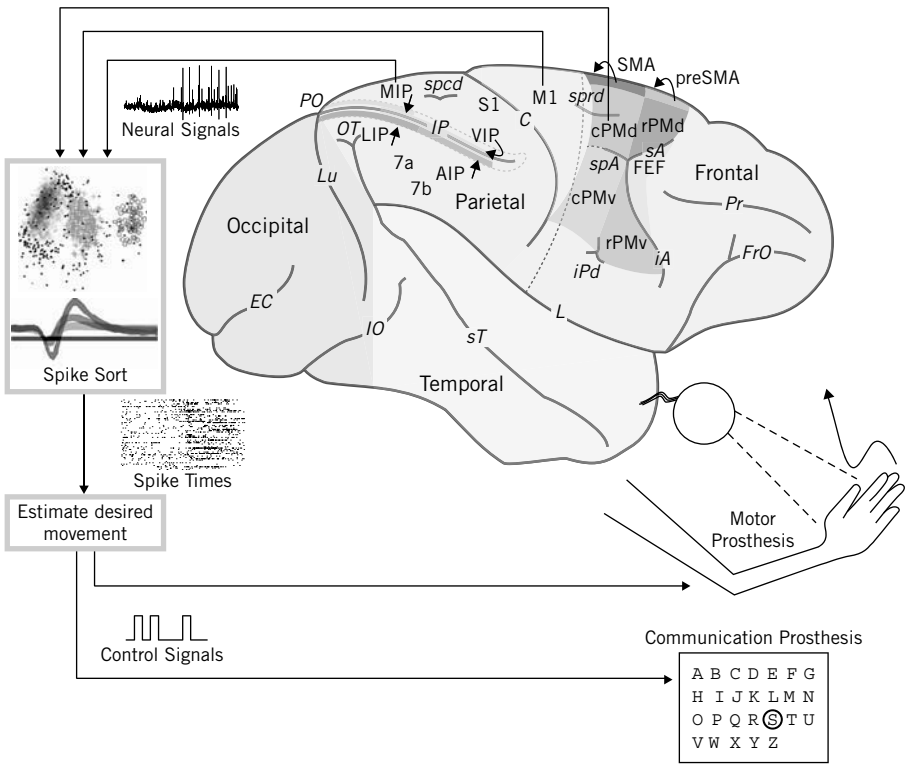

FIGURE 7.1

Concept sketch of cortically controlled motor and communication prostheses. Neural signals are recorded from cortical areas such as PMd, M1, and intraparietal area/parietal reach region (MIP/PRR) using intra-cortical electrode arrays. The times of action potentials (spikes) are identified and assigned to neural units. A decoder then translates the activity across a neural population into control signals for the prosthetic system. Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

associated neural activity, estimation algorithms can be designed to decode the desired movement trajectory (*continuous decoder*) or movement target (*discrete decoder*) from the pattern of neural activity. The system can then generate control signals appropriate for continuously guiding a paralyzed or prosthetic arm through space (*motor prosthesis*) or positioning a computer cursor on the desired letter on a keyboard (*communication prosthesis*).

Motor prostheses aim to provide natural control of the paralyzed limb, via functional electrical stimulation of de-innervated muscle, or of a prosthetic replacement limb. In the case of upper-limb prostheses, natural control

involves the precise movement of the arm along a desired path and with a desired speed profile. Such control is indeed a daunting ultimate goal, but presumably even intermediate steps along the way would lead to clinically viable systems. For example, simply being able to feed oneself would help thousands of tetraplegics (e.g., Velliste et al., 2008).

Communication prostheses do not aim to restore the ability to communicate in the form of natural voice or natural typing (i.e., moving the fingers). Instead, they aim to provide a fast and accurate communication mode, such as moving a computer cursor on an onscreen keyboard so as to type out words. Ideally performance could rival, or even surpass, the natural communication rate at which most people can speak or type. For example, "locked-in" ALS patients are altogether unable to converse with the outside world, and many other neurodegenerative diseases severely compromise the quality of speech. Being able to reliably type several words per minute on a computer will be a meaningful advance for these patients. In fact, many of the most severely disabled patients, who are the likely recipients of first-generation systems, will benefit from a prosthesis capable of typing even a few words per minute (e.g., Donoghue, 2008).

While motor and communication prostheses are quite similar conceptually, important differences critically affect their design. Motor prostheses must produce movement trajectories as accurately as possible to enact precisely the desired movement. The decoder translates, on a moment-by-moment basis, neural activity into a continuous-valued movement trajectory. In this case, the decoder solves a regression problem. In contrast, communication prostheses are concerned with information throughput from the subject to the world; this makes the speed and accuracy with which keys on a keyboard can be selected of primary importance. Although a continuously guided motor prosthesis can be used to convey information by moving to a key, only the key eventually struck actually contributes to information conveyance. The decoder, thus, acts as a classifier. This seemingly subtle distinction has implications that profoundly influence the type of neural activity and the class of decoder to be used.

In this chapter, we focus on the engineering goal of improving prosthetic system design, with particular emphasis on neural decoding algorithms. We make no claims here about these decoding algorithms furthering our basic scientific understanding of the relationship between neural activity and desired movements, although such findings may well inform next-generation algorithmic design. In the following sections, we will first describe the behavioral task and the different types of neural activity that are used to drive motor and communication prostheses. For both classes of prosthetic systems, we will then briefly review existing and detail new decoding algorithms that we

recently published (Yu et al., 2007; Santhanam et al., 2009). Finally, we will consider future directions for further improving decoding performance and increasing the clinical viability of neural prosthetic systems.

7.2 PLAN AND MOVEMENT NEURAL ACTIVITY

Two types of neural activity are commonly used for driving prosthetic systems: plan and movement activity. Plan activity is present before arm movements begin, or even without ultimate arm movement, and is believed to reflect preparatory processing required for the fast and accurate generation of movement (e.g., Churchland et al., 2006a,b,c; Churchland and Shenoy, 2007a). This activity is readily observed before movement initiation in a "delayed reach task" (Figure 7.2). In this task, a trial begins when the subject touches and visually fixates a central target. After a randomized delay period (typically 0–1000 ms), a "go cue" indicates that a reach (and saccade) may begin. Figure 7.2 illustrates that plan activity in a premotor cortex (PMd) unit of a monkey performing this task is tuned for the direction of the upcoming movement, with plan activity arising soon after target onset and persisting throughout the delay period. Plan activity has also been shown to reflect movement extent (e.g., Messier and Kalaska, 2000; Churchland et al., 2006b), peak velocity (Churchland et al., 2006b), eye position (Batista et al., 2007), and attention (e.g., Boussaoud and Wise, 1993). Movement activity follows plan activity in a delayed reach task, being present 100–200 ms before and during the movement. It is tuned for both the direction (as shown in Figure 7.2) and speed of arm movement (e.g., Moran and Schwartz, 1999a). PMd neurons typically exhibit strong plan activity, as well as some movement activity; neurons in primary motor cortex (M1) typically exhibit the opposite pattern (e.g., Shen and Alexander, 1997a,b).

Neural activity was recorded using a 96-channel silicon electrode array (Blackrock Micosystems, Salt Lake City, UT) implanted in two rhesus macaque monkeys (monkeys G and H). Several different data sets are used in this chapter. A typical data set comprises 1000–2000 trials and 100–200 simultaneously recorded units. Of these units, 30–40% are typically single-neuron and 60–70% are multi-neuron. For more details about the data sets used, see Yu et al. (2007) and Santhanam et al. (2009).

Until recently, both motor and communication prostheses have focused primarily on movement activity. As depicted in Figure 7.3(a), movement activity can be elicited merely by "thinking" about moving the arm and, surprisingly, it has been found that no movement or electromyographic (EMG) activity

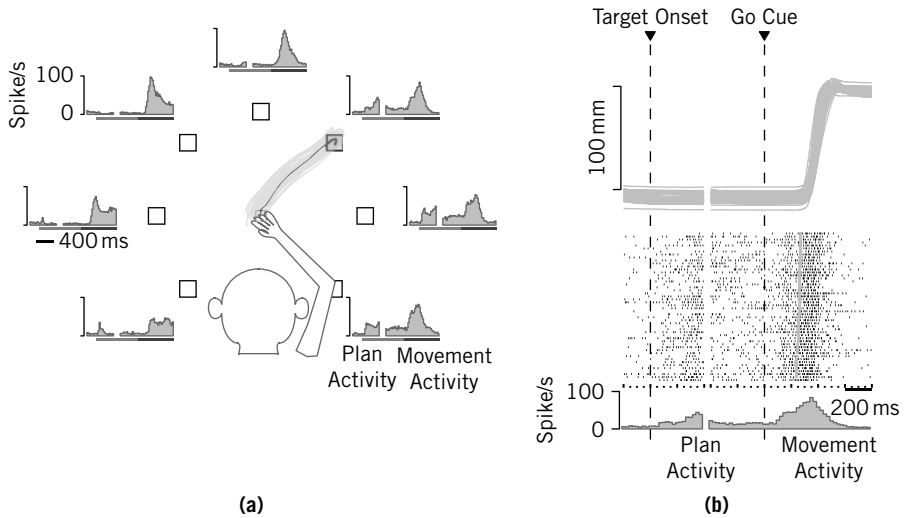

FIGURE 7.2

Plan and movement activity from a single PMd neuron in a delayed reach task. (a) Spike histograms showing average plan (medium gray bar) and movement (dark gray bar) activity associated with center-out reaches to peripheral targets. Fifty representative reach trajectories to the upward-right target are shown in light gray (mean trajectory in black). (b) Top panel: same 50 representative reach trajectories as in (a) shown as a function of time (horizontal component only); middle panel: spike times associated with each of these 50 reaches—each row corresponds to a trial, black tick marks indicate spike times, light bar indicates movement onset; bottom panel: a histogram of the average response.

need result (e.g., Taylor et al., 2002). With animal models, such as monkeys, neural activity that can be generated in the absence of arm movements (or presumably even *with* arm movements when descending nerves are intact and without nerve block (Moritz et al., 2008)) is currently considered to be an adequate proxy for neural signals from paralyzed subjects (e.g., Taylor et al., 2002; Velliste et al., 2008). Movement activity is then decoded to generate instantaneous direction and speed signals, which are used to guide a computer cursor (e.g., Serruya et al., 2002; Taylor et al., 2002; Carmena et al., 2003) or a robotic arm (e.g., Velliste et al., 2008) to a target. Motor prostheses incorporate movement activity because the goal is to recreate the desired movement path and speed. Although largely reflecting movement target, plan activity can play an important role in full trajectory estimation by providing an estimate of where the movement will end, and thereby helping to constrain the instantaneous movement estimates (Yu et al., 2007). This will be detailed in Section 7.3.

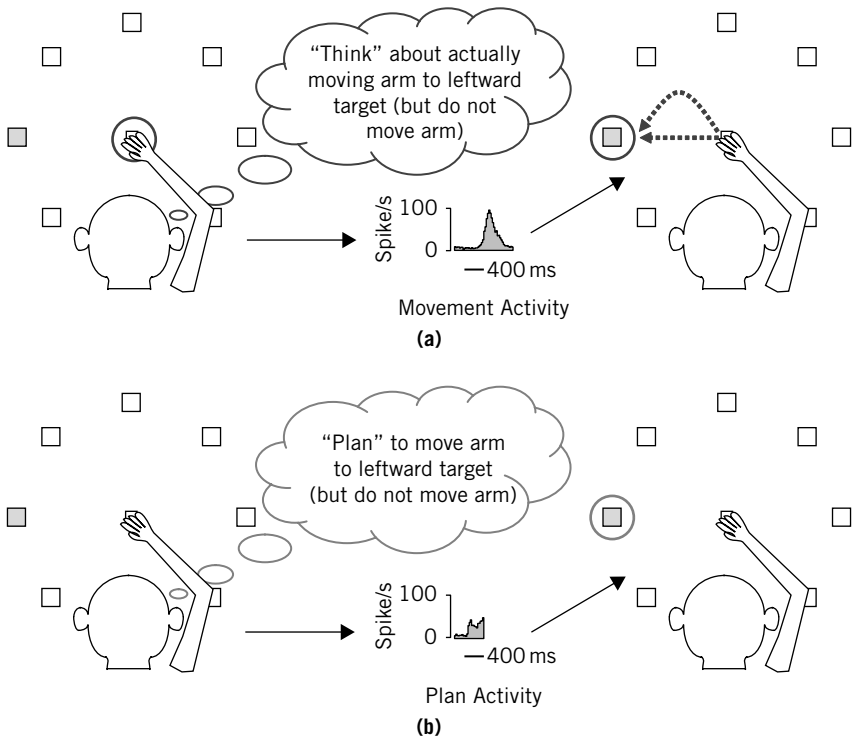

FIGURE 7.3

Differences between motor and communication prostheses. (a) For motor prostheses, movement activity that is generated by "thinking" about moving to the desired target is processed with a trajectory estimator, which guides a cursor (dark gray circle) along the desired path at the desired speed. (b) For communication prostheses, plan activity that is generated by "intending" to move to the desired target is then processed with a target estimator, which directly positions a cursor (medium gray circle) on the desired target.

In contrast, communication prostheses are not obliged to move along a continuous path in order to strike a target, such as a key on an onscreen keyboard (Figure 7.3(b)). Instead, if target location can be estimated directly from neural plan activity, the cursor can be positioned immediately on the desired key. Recent reports suggest that there may be a considerable performance benefit in using plan activity and direct-positional prosthesis control (e.g., Musallam et al., 2004; Santhanam et al., 2006). Figure 7.3(b) illustrates that plan activity can be elicited merely by "intending" to move the arm to a target/key location, and it is well established that it does not necessarily produce movements or EMG activity. Plan activity is then decoded to yield

the desired key; the prosthetic cursor then immediately appears and selects it. In this way, communication prostheses can rely on plan activity alone.

7.3 CONTINUOUS DECODING FOR MOTOR PROSTHESES

In the late 1960 and early 1970, Fetz and colleagues discovered that non-human primates could learn to regulate the firing rate of individual cortical neurons (Fetz, 1969; Fetz and Finocchio, 1971; Fetz and Baker, 1972). These pioneering experiments relied on straightforward forms of real-time feedback, but clearly demonstrated that firing rates could be brought to requested levels without accompanying muscle contraction, even in M1. In the 1970 and early 1980, Schmidt, Humphrey, and colleagues proposed that neural activity could be used to directly control prostheses (Humphrey et al., 1970). By the late 1990, technological advances and a much better understanding of how cortical neurons contribute to limb movement (e.g., Georgopoulos et al., 1986; Schwartz, 1994) sparked renewed interest in developing clinically viable systems. This required an ongoing series of experiments with animal models and disabled human patients with the goal of learning fundamental design principles and quantifying performance.

Nicolelis and colleagues investigated one-dimensional (1D) control of a motor prosthesis by training rats to press a lever to receive a liquid reward (Chapin et al., 1999). The apparatus was then altered such that the lever was controlled by movement activity across a population of cortical neurons. In particular, the lever movement was estimated using a combination of principal components analysis and an artificial neural network. The researchers found that rats could still control the lever to receive their reward. Animals soon learned that actual forelimb movement was not needed and stopped movements altogether while continuing to move the lever with brain-derived activity. This proved the basic feasibility of motor prostheses.

Meanwhile, investigations of 2D and full 3D control essential for re-creating natural arm movements were underway. Schwartz and colleagues demonstrated that 2D and 3D hand location could be reconstructed fairly accurately from the movement activity of a population of simultaneously recorded M1 neurons in rhesus monkeys (Isaacs et al., 2000), and Nicolelis and colleagues reported similarly encouraging reconstructions using simultaneous recordings from parietal cortex, PMd, and M1 (Wessberg et al., 2000). Together with similar recording studies from Donoghue and colleagues (Maynard et al., 1999), the stage was set for the first "closed-loop" 2D and 3D motor prosthesis experiments.

Schwartz and colleagues pursued 3D cursor control with rhesus monkeys (Taylor et al., 2002). A few tens of M1 neurons were recorded with a chronically implanted electrode array, and the monkeys made 3D reaching movements to visual targets appearing in a 3D virtual reality environment. Neural responses were characterized in terms of the movement direction eliciting maximal response (also known as the *preferred direction*) and were combined to form a modified population vector. Let $\mathbf{v}^i \in \mathbb{R}^{p\times 1}$ be a vector pointing in the preferred direction of neural unit i, where p is the number of dimensions of the workspace (in this case, three). In its simplest form, the population vector $\hat{\mathbf{x}}_t \in \mathbb{R}^{p\times 1}$ at time t is defined as

$$\hat{\mathbf{x}}_t = \frac{\sum_{i=1}^{q} y_t^i \, \mathbf{v}^i}{\sum_{i=1}^{q} y_t^i}, \tag{7.1}$$

where $y_t^i \in \mathbb{R}$ is the activity of unit $i \in \{1,\ldots,q\}$ at time t. The population vector indicates the instantaneous movement of the hand or, in prosthesis mode, the prosthetic cursor.

Monkeys then entered "brain-control" mode, wherein the 3D prosthetic cursor was controlled by the neural population vector, as opposed to the location of the hand. This group has since gone on to demonstrate that monkeys can use these 3D control signals to feed themselves with an anthropomorphic robotic arm, and open and close a hand-like gripper (Velliste et al., 2008).

The fitting procedure for the preferred directions $\mathbf{v}^i$ and more sophisticated variants of the population vector are described elsewhere (Zhang et al., 1998; Moran and Schwartz, 1999b; Taylor et al., 2002). The population vector assumes a uniform distribution of preferred directions; when this assumption is not satisfied, the population vector is biased and an optimal linear estimator (OLE) may be preferred (Salinas and Abbott, 1994). The OLE takes the same form as Eq. (7.1); the only difference is how the parameters $\mathbf{v}^i$ are obtained. It was recently shown that, in a closed-loop setting, the subject could compensate for a bias in the population vector arising from a nonuniform distribution of preferred directions (Chase et al., 2009).

Donoghue and colleagues pursued 2D cursor control with rhesus monkeys (Serruya et al., 2002). A few tens of M1 neurons were recorded simultaneously with a chronically implanted electrode array while monkeys moved a manipulandum to guide the cursor. By recording spike activity while the monkeys tracked a continuously, pseudorandomly moving target, a linear filter could be learned to relate neural activity to cursor movement. A linear filter takes a linear combination of activity from different simultaneously recorded neurons to estimate physical state variables, such as cursor position or velocity.

The estimated state variable $\hat{x}_t \in \mathbb{R}$ at time t is

$$\hat{x}_t = b + \sum_{i=1}^{q} \sum_{\tau=0}^{N-1} y_{t-\tau}^i a_\tau^i \tag{7.2}$$

where $b \in \mathbb{R}$ is an additive constant, $y_t^i \in \mathbb{R}$ is the activity of neural unit $i \in \{1,\ldots,q\}$ at time t, and $\{a_0^i,\ldots a_{N-1}^i\}$ are the N filter weights for unit i. These filter weights are fit using least squares (Warland et al., 1997). For a multi-dimensional state variable, a separate set of filter weights and additive constants is fit for each dimension. Unlike the population vector and OLE, the linear filter can combine neural activity across multiple time steps and thus produce smoother trajectory estimates. This linear filter was used in a new task, where neural activity guided the prosthetic cursor to hit visual targets appearing at random locations. Together with collaborators and Cyberkinetics Neurotechnology Inc., this group has since gone on to show for the first time that a population of M1 neurons from tetraplegic patients can be used to control 2D cursors (Hochberg et al., 2006).

Nicolelis and colleagues pursued 2D cursor control, along with a form of prosthetic grasping, with rhesus monkeys (Carmena et al., 2003). Hundreds of M1, PMd, supplementary motor area (SMA), primary sensory area (S1), and posterior parietal neurons were recorded with chronically implanted electrode arrays while monkeys performed each of three behavioral tasks. The animals were trained to move a pole to control the position of an onscreen cursor and to grip the pole to control the size of the cursor, which indicated grip force. Linear filters (Eq. (7.2)) were trained to relate neural activity to a variety of motor parameters including hand position, velocity, and gripping force. These models were used in "brain control" mode to translate neural activity into cursor movement and cursor size. This study demonstrates (1) that grip force may be controlled (as does the more recent Velliste et al. (2008) report), which is essential once a prosthetic arm arrives as the desired location; and (2) that combined positioning and gripping are possible.

7.3.1 Recursive Bayesian Decoders

While linear filters and population vectors are effective, recursive Bayesian decoders have been shown to provide more accurate trajectory estimates in offline (i.e., open-loop) settings (Brown et al., 1998; Brockwell et al., 2004; Wu et al., 2004a, 2006) and online (i.e., closed-loop) settings (Wu et al., 2004b; Kim et al., 2008). In contrast to linear filters and population vectors, recursive Bayesian decoders are based on the specification of an explicit probabilistic model, comprising (1) a *trajectory model*, which

describes how the arm state changes from one time step to the next, and (2) an *observation model*, which describes how the observed neural activity relates to the time-evolving arm state. The trajectory and observation models each include a probabilistic component, which attempts to capture trial-to-trial differences in the arm trajectory and neural spiking noise, respectively. These probabilistic relationships provide confidence regions for the arm state estimates, in contrast to linear filters and population vectors. If the modeling assumptions are satisfied, Bayesian estimation makes optimal use of the observed data. With an explicit probabilistic model, it is straighforward to relax the modeling assumptions and propose extensions, whereas such changes are not easily incorporated in a linear filter or a population vector. We will exploit this flexibility in the next subsection.

A widely used recursive Bayesian decoder is the Kalman filter, which is based on a linear Gaussian trajectory model:

$$\mathbf{x}_t \mid \mathbf{x}_{t-1} \sim \mathcal{N}(A\mathbf{x}_{t-1}, Q) \tag{7.3}$$

$$\mathbf{x}_1 \sim \mathcal{N}(\boldsymbol{\pi}, V) \tag{7.4}$$

and a linear Gaussian observation model:

$$\mathbf{y}_t \mid \mathbf{x}_t \sim \mathcal{N}(C\mathbf{x}_t, R), \tag{7.5}$$

where $\mathbf{x}_t \in \mathbb{R}^{p\times 1}$ is the arm state (which may include terms such as arm position, velocity, and acceleration) at time $t \in \{1,\ldots,T\}$, and $\mathbf{y}_t \in \mathbb{R}_+^{q\times 1}$ is the movement-period activity across the q neural units at time t. The parameters $A \in \mathbb{R}^{p\times p}$, $Q \in \mathbb{R}^{p\times p}$, $\boldsymbol{\pi} \in \mathbb{R}^{p\times 1}$, $V \in \mathbb{R}^{p\times p}$, $C \in \mathbb{R}^{q\times p}$, and $R \in \mathbb{R}^{q\times q}$ do not depend on time and are fit to training data using maximum likelihood (Wu et al., 2006). Linear Gaussian models have been successfully applied to decoding the path of a foraging rat (Brown et al., 1998; Zhang et al., 1998), as well as arm trajectories in ellipse-tracing (Brockwell et al., 2004), pursuit-tracking (Wu et al., 2004a, 2006; Shoham et al., 2005), and "pinball" tasks (Wu et al., 2004a; Wu et al., 2006).

The task of decoding a continuous arm trajectory involves finding the likely sequences of arm states corresponding to the observed neural activity. At each time step t, we seek to compute the distribution of the arm state $\mathbf{x}_t$ given the movement neural activity $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t$ (denoted by $\{\mathbf{y}\}_1^t$) observed up to that time. This distribution is written $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$ and termed the *state posterior*. State posteriors can be obtained by iterating the following two updates. First, the one-step prediction is found by applying Eq. (7.3) to the state posterior at

the previous time step:

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t-1}\right) = \int P\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) P\left(\mathbf{x}_{t-1} \mid \{\mathbf{y}\}_1^{t-1}\right) d\mathbf{x}_{t-1}. \tag{7.6}$$

Second, the state posterior at the current time step is computed using Bayes' rule

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t}\right) = \frac{P\left(\mathbf{y}_t \mid \mathbf{x}_t\right) P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t-1}\right)}{P\left(\mathbf{y}_t \mid \{\mathbf{y}\}_1^{t-1}\right)}. \tag{7.7}$$

Note that $P\left(\mathbf{y}_t \mid \mathbf{x}_t, \{\mathbf{y}\}_1^{t-1}\right)$ has been replaced by $P\left(\mathbf{y}_t \mid \mathbf{x}_t\right)$ to obtain Eq. (7.7) since, given the current arm state $\mathbf{x}_t$, the current observation $\mathbf{y}_t$ does not depend on the previous observations $\{\mathbf{y}\}_1^{t-1}$ (Eq. (7.5)). The terms in the numerator of Eq. (7.7) are the observation model from Eq. (7.5) and the one-step prediction from Eq. (7.6). The denominator of Eq. (7.7) can be obtained by integrating the numerator over $\mathbf{x}_t$.

The linear Gaussian trajectory and observation models (Eqs. (7.3) through (7.5)) can then be substituted into Eqs. (7.6) and (7.7) to obtain the key equations of the Kalman filter. Because all variables $\mathbf{x}_t$ and $\mathbf{y}_t$ defined by Eqs. (7.3) through (7.5) are jointly Gaussian, the one-step prediction $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t-1}\right)$ and state posterior $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t}\right)$ are also Gaussian in $\mathbf{x}_t$. Thus, the distributions are fully specified by their means and variances. Let $E\left[\mathbf{x}_t \mid \{\mathbf{y}\}_1^{\tau}\right]$ be denoted by $\mathbf{x}_t^{\tau}$ and let $\mathrm{Var}(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{\tau})$ be denoted by V_t^{τ}. The mean and variance of the one-step prediction are

$$\mathbf{x}_t^{t-1} = A\mathbf{x}_{t-1}^{t-1} \tag{7.8}$$

$$V_t^{t-1} = AV_{t-1}^{t-1}A' + Q \tag{7.9}$$

The mean and variance of the state posterior are

$$\mathbf{x}_t^{t} = \mathbf{x}_t^{t-1} + K_t(\mathbf{y}_t - C\mathbf{x}_t^{t-1}) \tag{7.10}$$

$$V_t^{t} = (I - K_t C)V_t^{t-1}, \tag{7.11}$$

where

$$K_t = V_t^{t-1}C'\left(CV_t^{t-1}C' + R\right)^{-1}$$

The recursions specified by Eqs. (7.8) through (7.11) start with $\mathbf{x}_1^0 = \boldsymbol{\pi}$ and $V_1^0 = V$. Thus, the Kalman filter produces an arm trajectory estimate with

mean $\mathbf{x}_1^1,\ldots,\mathbf{x}_T^T$ and corresponding variance $V_1^1,\ldots,V_T^T$. The variances define the confidence intervals corresponding to the estimated mean trajectory. As shown by Black and colleagues (Wu et al., 2006), the Kalman filter can be viewed as a linear filter in which (1) all past neural activity is taken into account, rather than just $N-1$ timesteps in the past (Eq. (7.2)), and (2) a confidence interval is produced to accompany the estimated trajectory.

7.3.2 Mixture of Trajectory Models

The function of the trajectory model is to build into the recursive Bayesian decoder prior knowledge about the form of the reaches. The model may reflect (1) the hard, physical constraints of the limb (for example, the elbow cannot bend backwards), (2) the soft, control constraints imposed by neural mechanisms (for example, the arm is more likely to move smoothly than in a jerky motion), and (3) the physical surroundings of the patient and his/her objectives in that environment. The degree to which the trajectory model reflects the kinematics of the actual reaches directly affects the accuracy with which trajectories can be decoded from neural data. When selecting a trajectory model, one is typically faced with a trade-off between how accurately a model captures the movement statistics and the computational demands of its corresponding decoder. For example, in real-time applications, we may decide to use a relatively simple trajectory model because of its low computational cost, even if it fails to capture some of the salient properties of the observed movements. While we may be able to identify a more complex trajectory model that could yield more accurate decoded trajectories, the computational demands of the corresponding decoder may be prohibitive in a real-time setting.

Here, we review a general approach to constructing trajectory models that can exhibit rather complex dynamical behaviors, whose decoder can be implemented to have the same running time as that of simpler trajectory models. The core idea was to combine simple trajectory models, each accurate within a limited regime of movement, in a probabilistic mixture of trajectory models (MTM) (Kemere et al., 2004; Yu et al., 2007). The development of the mixture framework was made possible by viewing the neural decoding problem as state estimation based on an explicit probabilistic model. We demonstrated the utility of this approach by developing a particular mixture of trajectory models suitable for goal-directed movements in settings with multiple goals. In the following, we will review (1) how to decode movements from neural activity under the general MTM framework, (2) the development of a particular MTM decoder for goal-directed movements, and (3) how this decoder

can naturally incorporate prior goal information, when available, to improve the accuracy of the decoded trajectories.

7.3.2.1 ***Mixture of Trajectory Models Framework***

Ideally, we wanted to construct a complete model of neural motor control that captures the hard physical constraints of the limb (Chan and Moran, 2006), the soft control constraints imposed by neural mechanisms, and the physical surroundings and context. One way to approximate such a complete model is to probabilistically combine trajectory models that are each accurate within a limited regime of movement. Examples of movement regimes include different parts of the workspace, different reach speeds, and different reach curvatures. At the onset of a new movement, the movement regime is unknown, or imperfectly known, so the full trajectory model is composed of a mixture of the individual, regime-specific trajectory models. Here, we review the development of a recursive Bayesian decoder based on a mixture of trajectory models (Yu et al., 2007).

As with the Kalman filter, we sought to compute the state posteriors $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$, which define the decoded trajectory. If the actual movement regime $m^\star$ is perfectly known before the reach begins, then we can compute the state posterior based only on the individual trajectory model corresponding to that regime. This distribution is written $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m^\star\right)$ and termed the *conditional state posterior*. However, in general the actual movement regime is unknown or imperfectly known, so we needed to compute $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right)$ for each $m \in \{1,\ldots,M\}$, where M is the number of movement regimes (also referred to as mixture components).

To combine the M conditional state posteriors, we simply expanded $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$ by conditioning on the movement regime m:

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right) = \sum_{m=1}^{M} P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right) P\left(m \mid \{\mathbf{y}\}_1^t\right) \tag{7.12}$$

In other words, the state posterior is a weighted sum of the conditional state posteriors. The weights $P\left(m \mid \{\mathbf{y}\}_1^t\right)$ represent the probability that the actual movement regime is m, given the observed spike counts up to time t. Bayes' rule was then applied to these weights in Eq. (7.12), yielding the key equation for the MTM framework:

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right) = \sum_{m=1}^{M} P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right) \frac{P\left(\{\mathbf{y}\}_1^t \mid m\right) P(m)}{P\left(\{\mathbf{y}\}_1^t\right)} \tag{7.13}$$

The conditional state posteriors $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right)$ in Eq. (7.13) can be computed or approximated using any of a number of different recursive Bayesian decoding techniques, including the Kalman filter (Wu et al., 2006), Bayes' filter (Brown et al., 1998), and particle filters (Brockwell et al., 2004; Shoham et al., 2005). This is usually based on the recursions Eqs. (7.6) and (7.7). The likelihood terms $P\left(\{\mathbf{y}\}_1^t \mid m\right)$ in Eq. (7.13) can be expressed as

$$P\left(\{\mathbf{y}\}_1^t \mid m\right) = \prod_{\tau=1}^{t} P\left(\mathbf{y}_\tau \mid \{\mathbf{y}\}_1^{\tau-1}, m\right) \tag{7.14}$$

where

$$P\left(\mathbf{y}_t \mid \{\mathbf{y}\}_1^{t-1}, m\right) = \int P\left(\mathbf{y}_t \mid \mathbf{x}_t\right) P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t-1}, m\right) d\mathbf{x}_t \tag{7.15}$$

The integral in Eq. (7.15) includes the observation model $P\left(\mathbf{y}_t \mid \mathbf{x}_t\right)$ and the one-step prediction $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^{t-1}, m\right)$ (Eq. (7.6)) using the parameters of mixture component m. If available, prior information about the identity of the movement regime can be incorporated naturally into the MTM framework via $P(m)$ in Eq. (7.13). This information must be available before the reach begins and may differ from trial to trial. If no such information is available, the same $P(m)$ (e.g., a uniform distribution) can be used across all trials.

The computational complexity of the MTM decoder is M times that of computing $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right)$ and $P\left(\{\mathbf{y}\}_1^t \mid m\right)$ for a particular mixture component m. Because the computations for each mixture component can theoretically be carried out in parallel, it is possible to set up the MTM decoder so that its running time remains constant, regardless of the number of mixture components M. In other words, the MTM approach enables the use of more flexible, and potentially more accurate, trajectory models without a necessary penalty in decoder running time. Furthermore, the MTM decoder preserves the real-time properties of its constituent estimators and thus is suitable for real-time prosthetic applications.

7.3.2.2 *Mixture of Trajectory Models for Goal-Directed Movements*

For goal-directed movements, we proposed using the following mixture of linear Gaussian trajectory models

$$\mathbf{x}_t \mid \mathbf{x}_{t-1}, m \sim \mathcal{N}\left(A_m \mathbf{x}_{t-1} + \mathbf{b}_m, Q_m\right) \tag{7.16}$$

$$\mathbf{x}_1 \mid m \sim \mathcal{N}\left(\boldsymbol{\pi}_m, V_m\right) \tag{7.17}$$

where $m \in \{1,\ldots,M\}$ indexes reach target and M is the number of reach targets (Yu et al., 2007). In other words, each movement regime corresponds to movements heading toward a particular reach goal. As before, $\mathbf{x}_t \in \mathbb{R}^{p\times 1}$ is the arm state at time $t \in \{1,\ldots,T\}$ containing position, velocity, and acceleration terms. We used the observation model

$$y_t^i \mid \mathbf{x}_t \sim \text{Poisson}\left(e^{\mathbf{c}_i'\mathbf{x}_t + d_i}\Delta\right) \tag{7.18}$$

where $y_t^i \in \{0,1,2,\ldots\}$ is the movement-period spike count for unit $i \in \{1,\ldots,q\}$ taken in a time bin of width Δ. For notational convenience, the spike counts across the q simultaneously recorded units were assembled into a $q \times 1$ vector $\mathbf{y}_t$, whose i^{th} element is y_t^i. The parameters $A_m \in \mathbb{R}^{p\times p}$, $\mathbf{b}_m \in \mathbb{R}^{p\times 1}$, $Q_m \in \mathbb{R}^{p\times p}$, $\boldsymbol{\pi}_m \in \mathbb{R}^{p\times 1}$, $V_m \in \mathbb{R}^{p\times p}$, $\mathbf{c}_i \in \mathbb{R}^{p\times 1}$, $d_i \in \mathbb{R}$ do not depend on time and are fit to training data (for details, see Yu et al. (2007)). Note that, whereas each mixture component indexed by m in the trajectory model (Eqs. (7.16) and (7.17)) can have different parameters leading to different arm state dynamics, the observation model (Eq. (7.18)) is the same for all m. It is also possible to probabilistically mix other related trajectory models, in which trajectories are directed toward a particular target location (Srinivasan et al., 2006; Kulkarni and Paninski, 2008).

For this model, the conditional state posteriors can be obtained using the recursions in Eqs. (7.6) and (7.7). Unlike for the Kalman filter, the observation model (Eq. (7.18)) is not linear Gaussian. This leads to distributions that are difficult to manipulate, and the integral in Eq. (7.6) cannot be computed analytically. We instead employed a modified Kalman filter that uses a Gaussian approximation during the measurement update step (Eq. (7.7)). We approximated the conditional state posterior as a Gaussian matched to the location and curvature of the mode of $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right)$ (MacKay, 2003). This Gaussian approximation allows the integral in Eq. (7.6) to be computed analytically, since each mixture component of the full trajectory model (Eq. (7.16)) is linear Gaussian. This yields a Gaussian one-step prediction, which is fed back into Eq. (7.7).

7.3.2.3 Incorporating Target Information from Plan Activity

Up to this point, the neural activity discussed has been movement activity, which takes place around the time of movement and specifies the moment-by-moment details of the arm trajectory. In the delayed reach task, there is also neural activity present during an instructed delay period that directly precedes the "go cue" (termed *plan activity*). Neurons with plan activity are typically also active in the absence of an instructed delay during the reaction time

period (Crammond and Kalaska, 2000; Churchland et al., 2006c). Rather than specifying the moment-by-moment details of the trajectory, plan activity has been shown to reliably indicate the upcoming reach target (Shenoy et al., 2003; Hatsopoulos et al., 2004; Musallam et al., 2004; Yu et al., 2004; Santhanam et al., 2006). Both types of activity may be emitted by the same unit on a single trial, as can be seen in Figure 7.2.

The following describes how the reach target can be decoded from plan activity by applying Bayes' rule. Let $\mathbf{z}$ be a $q \times 1$ vector of spike counts across the q simultaneously recorded units in a prespecified time window during the delay period on a single trial. The distribution of spike counts (from training data) for each reach target m are fit to a Gaussian (Maynard et al., 1999; Yu et al., 2004):

$$\mathbf{z} \mid m \sim \mathcal{N}(\boldsymbol{\mu}_m, R_m) \tag{7.19}$$

where $\boldsymbol{\mu}_m \in \mathbb{R}^{q \times 1}$ is a vector of mean counts for reach target $m \in \{1, \ldots, M\}$, and the covariance matrix $R_m \in \mathbb{R}^{q \times q}$ is assumed to be diagonal. In other words, the units are assumed to be independent given the reach target m. In Section 7.4, we will describe how to relax this conditional independence assumption.

For any test trial, the probability that the upcoming reach target is m given the plan activity $\mathbf{z}$ can be computed by applying Bayes' rule:

$$P(m \mid \mathbf{z}) = \frac{P(\mathbf{z} \mid m)P(m)}{P(\mathbf{z})} = \frac{P(\mathbf{z} \mid m)}{\sum_{m'} P(\mathbf{z} \mid m')} \tag{7.20}$$

where $P(m)$ in Eq. (7.20) is assumed to be uniform. The target information from the plan activity, $P(m \mid \mathbf{z})$, can be incorporated naturally in the MTM framework in the place of $P(m)$ in Eq. (7.13). The distribution $P(m)$ in Eq. (7.13) represents the prior knowledge (i.e., prior to movement onset) that the upcoming reach target is m. Because the plan activity entirely precedes movement onset and provides information about the upcoming reach target, it can be used to set $P(m)$ in Eq. (7.13) on a per-trial basis.

It is important to note that the most likely target from Eq. (7.20) is not simply assumed here to be the target of the upcoming reach. On a given trial, the plan activity may not definitively indicate the target of the upcoming reach (e.g., two different reach targets may have significant probability), or it may indicate an incorrect target for the upcoming reach. In this case, you want to allow the subsequent movement activity, to determine the target of the reach, or even correct the mistake, "in flight". Instead of making a hard target decode

based on plan activity, the entire distribution $P(m \mid \mathbf{z})$ is retained and passed to the MTM framework.

7.3.3 Results

Here, we review the performance comparison of four decoders. The first is a state-of-the-art decoder presented by Kass and colleagues (Brockwell et al., 2004) based on a random-walk trajectory model (RWM) in acceleration. The second decoder is based on a single linear Gaussian trajectory model (STM) shared across reaches to all targets. It is defined by Eqs. (7.16) and (7.17) for special case of $M = 1$. The STM decoder uses the observation model shown in Eq. (7.18). The RWM and STM decoders provided points of comparison for the following two MTM decoders, both of which are based on Eqs. (7.16) and (7.18). Whereas the MTM_{M} decoder uses only movement activity, the MTM_{DM} decoder uses plan activity as well. In Eq. (7.13), the same $P(m)$ (in this case, a uniform distribution) is used across all trials for MTM_{M}. A different $P(m)$ is used on each trial for MTM_{DM} based on the prior target information extracted from plan activity.

For each of the four decoders, we first fit the model parameters to training data, as detailed in Yu et al. (2007). The test data for a single trial consisted of (1) the arm trajectory, taken from 50 ms before movement onset to 50 ms after movement end at $dt = 10$ ms time steps; (2) the movement period spike counts, taken in non-overlapping $\Delta = 10$ ms bins and temporally offset from the arm trajectory by an optimal lag found for each unit; and (3) the plan period spike counts, taken in a single 200 ms bin starting 150 ms after the appearance of the reach target. Arm trajectories in the test phase were used to evaluate offline the accuracy of the trajectories estimated from neural data. Note that natural arm movements accompanied the recorded movement activity (as is typical in offline studies), in contrast to most online closed-loop studies in which few or no arm movements are produced.

Figure 7.4 details, for a particular test trial, how the MTM-decoded trajectory was obtained and compares the trajectory estimates produced by the different decoders. From Eq. (7.12), the MTM-decoded trajectory $E\left[\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right]$ is a weighted sum of component trajectory estimates $E\left[\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right]$, one for each reach target indexed by $m \in \{1, \ldots, 8\}$. In Figure 7.4(b) and (c), the component trajectory estimates are plotted in the upper panels, while the middle panels show how the corresponding weights $P\left(m \mid \{\mathbf{y}\}_1^t\right)$ evolved during the course of the trial.

The values of the weights at time zero ($t = 0$) represent the probability that the upcoming reach target is m, before any movement neural activity is observed. The distribution of weights at $t = 0$ is precisely $P(m)$ in Eq. (7.13).

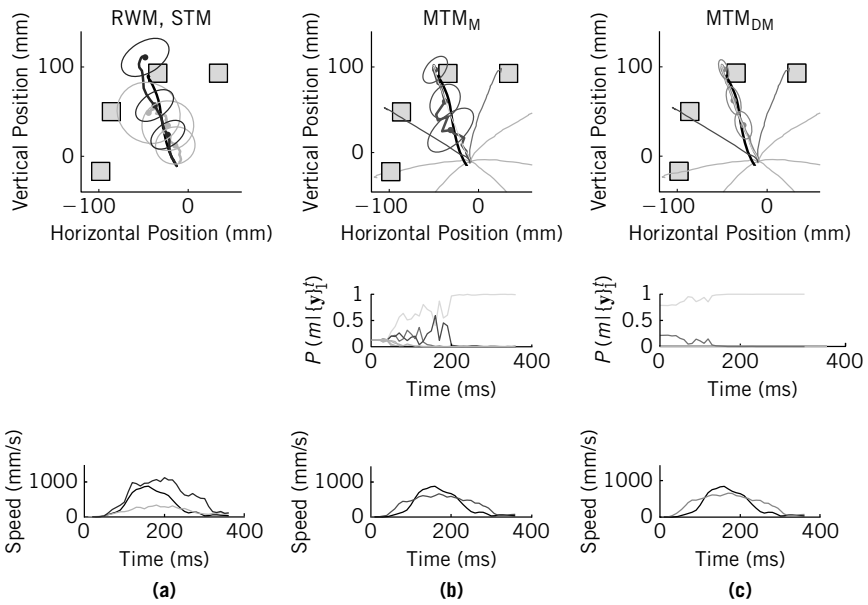

FIGURE 7.4

A representative test trial in which the use of plan activity improved the MTM-decoded trajectory. The upper panels compare the actual trajectory (all panels, black) with the decoded trajectories for (a) RWM (dark green) and STM (light green), (b) MTM_M (red), and (c) MTM_{DM} (orange). Ellipses denote 95% confidence intervals at three different time steps. Yellow squares represent the visual reach targets presented to the monkey in actual dimensions. The upper panels in (b) and (c) also show the eight component trajectory estimates for the MTM (cyan, blue, magenta for the three components with the largest weights; gray for the other five components). Their corresponding weights, as they evolve during the trial, are plotted in the middle panels. The lower panels compare the actual and estimated single-trial speed profiles using the same color conventions as in the upper panels. Time zero corresponds to 60 ms before movement onset (i.e., one time step before we begin to decode movement). Note that the red and orange traces in the upper panels are overlaid with the cyan trace. For this trial, E_{rms} was 11.8, 26.9, 10.9, and 10.5 mm for the RWM, STM, MTM_M, and MTM_{DM}, respectively. Monkey G, 98 units. (Experiment G20040508, trial ID 474). Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

Source: Reproduced with permission from Yu et al., 2007.

In Figure 7.4(b), we assumed that there was no information available about the identity of the upcoming reach target before the reach began (i.e., no plan activity), so all eight targets were equiprobable (i.e., $P(m) = 1/8$ for $m \in \{1, \ldots, 8\}$). As time proceeded, these weights were updated as more and more movement activity was observed. Recall that $P\left(m \mid \{\mathbf{y}\}_1^t\right)$ represents the probability that the actual reach target is m, given the observed neural activity up to time t.

During the first 200 ms, the actual reach target (cyan) was more likely than the other seven reach targets at nearly every time step; however, there was some competition with the neighboring reach targets (blue and magenta). It was only after approximately 200 ms that the decoder became certain of the actual reach target (i.e., $P\left(m \mid \{\mathbf{y}\}_1^t\right)$ approached unity) and remained certain for the rest of the trial. A weighted sum of the eight component trajectory estimates (upper panel) using these weights (middle panel) yields the MTM-decoded trajectory (upper panel, red; E_{rms}: 10.9 mm). To measure decoding performance, we computed the root mean square position error (E_{rms}) between the actual and decoded trajectories.

If plan activity is available, it can be used to set a nonuniform $P(m)$ in Eq. (7.13) on a per-trial basis, as previously discussed. The only difference between Figure 7.4(b) and (c) is that the MTM decoder used plan activity in the latter but not the former. In Figure 7.4(c) (middle panel), the weights at $t = 0$ represent the probabilities of each reach target based only on plan activity, before any movement activity was observed. In this case, the plan activity indicated that the actual reach target (cyan) was more probable than the other targets. This prior knowledge of the identity of the upcoming reach target was then taken into account when updating the weights $P\left(m \mid \{\mathbf{y}\}_1^t\right)$ during the course of the trial as more and more movement activity was observed. Note that using plan activity only affected $P(m)$ in Eq. (7.13); the conditional state posteriors $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t, m\right)$ and the likelihood terms $P\left(\{\mathbf{y}\}_1^t \mid m\right)$ remained unchanged. As the means of the conditional state posteriors, the component trajectory estimates therefore also remained unchanged, as can be verified by comparing Figure 7.4(b) and (c) (upper panels). For the trial shown in Figure 7.4, the use of plan activity reduced the competition between the actual reach target (cyan) and the neighboring targets (blue and magenta). Compared to Figure 7.4(b) (middle panel), the weight for the actual reach target (cyan) in Figure 7.4(c) (middle panel) was higher at every time point, the clearest effect seen during the first 200 ms. In other words, by using plan activity, the decoder was more certain of the actual reach target throughout the trial. In Figure 7.4(c), a weighted sum of the eight component trajectory estimates (upper panel) using these weights (middle panel) yields the MTM-decoded trajectory (upper panel, orange; E_{rms}: 10.5 mm).

By comparing the MTM-decoded trajectories with the actual trajectory in Figure 7.4(b) and (c) (upper panels), we see that the use of plan activity decreased the decoding error and tightened the confidence ellipses for this trial. Both MTM-decoded trajectories had lower decoding error than the RWM (E_{rms}: 11.8 mm) and STM (E_{rms}: 26.9 mm), whose decoded trajectories are plotted in Figure 7.4(a) (upper panel). On this trial, the RWM decoder produced a reasonably accurate decoded trajectory, while the STM-decoded

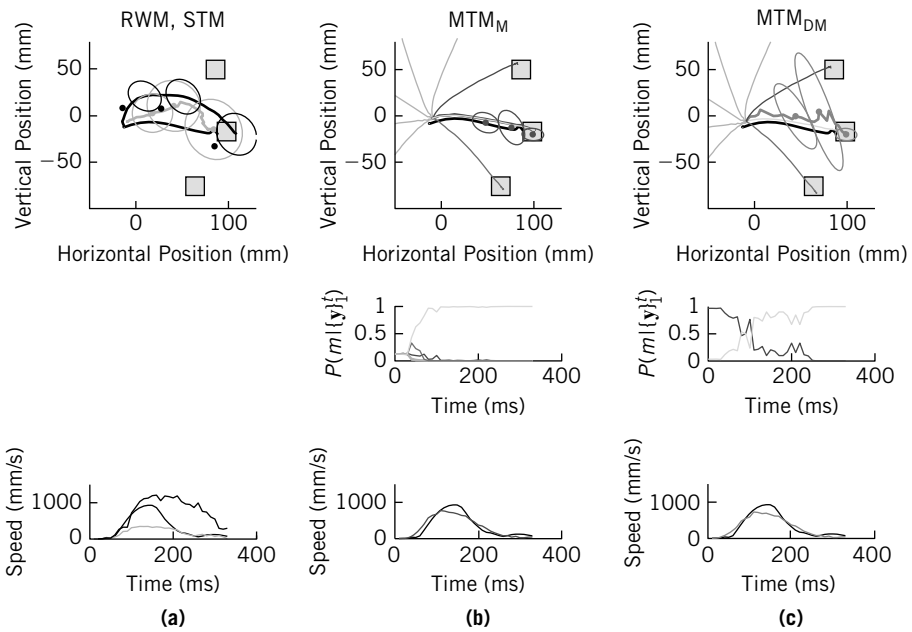

FIGURE 7.5

A representative test trial in which the movement activity corrected an incorrect target identification from the plan activity. Figure conventions are identical to those in Figure 7.4. For this trial, E_{rms} was 27.7, 24.2, 11.2, and 13.0 mm for RWM, STM, MTM$_M$, and MTM$_{DM}$, respectively. Monkey G, 98 units. (Experiment G20040508, trial ID 676). Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

Source: Reproduced with permission from Yu et al., 2007.

trajectory proceeded slowly outwards with wide confidence intervals. These decoders can also be used to estimate the bell-shaped speed profile of the actual reach (Figure 7.4, lower panels).

In contrast to Figure 7.4, Figure 7.5 shows a trial where movement activity alone was able to quickly determine the actual reach target without much competition from neighboring targets. This can be seen in Figure 7.5(b) (middle panel), where the weight corresponding to the actual reach target (cyan) rose to unity after approximately 100 ms and stayed there for the remainder of the trial. As a result, the resulting MTM-decoded trajectory (upper panel, red; E_{rms}: 11.2 mm) was quite accurate. As in Figure 7.4, we can incorporate plan activity if available; however, in this case, the dominant weight at $t = 0$ (blue) did *not* correspond to the actual reach target (cyan), as seen in Figure 7.5(c) (middle panel). In other words, the plan activity incorrectly indicated the identity of the upcoming reach target. However, as these weights were updated by

the observation of movement activity, this "error" was soon corrected (within approximately 100 ms). From that point on, the weight corresponding to the actual reach target dominated. Despite this error at the beginning of the trial, the MTM decoded trajectory in Figure 7.5(c) (upper panel, orange; E_{rms}: 13.0 mm) still headed to the correct target and provided a reasonably accurate estimate of the arm trajectory. The larger confidence ellipses for MTM_{DM} compared to MTM_{M} reflect the competition between the actual (cyan) and neighboring (blue) reach targets. The decoded trajectories for the RWM (dark green) and STM (light green) are shown in Figure 7.5(a) for comparison. As in Figure 7.4, both MTM-decoded trajectories yielded lower decoding error than the RWM (E_{rms}: 27.7 mm) and STM (E_{rms}: 24.2 mm). Furthermore, as shown in the lower panels, the speed profiles estimated by the MTM decoders (red and orange) tracked the actual bell-shaped speed profile (black) more closely than those estimated by the RWM (dark green) and STM (light green) decoders.

Figures 7.4 and 7.5 together illustrate the benefits of the joint use of plan and movement activity. When one type of activity is unable to definitively identify (or incorrectly identifies) the actual reach target, the MTM framework allows the other type of activity to strengthen (or overturn) the target identification in a probabilistic manner. In Figure 7.4, the movement activity alone was unable to definitively identify the actual reach target during the first 200 ms, as there was competition with a neighboring target. When prior target information from plan activity was incorporated, the decoder was more certain of the actual reach target throughout the trial. In Figure 7.5, the plan activity incorrectly indicated the identity of the upcoming reach target. However, the movement activity overturned this incorrect target identification early on and rescued the decoder from incurring a large E_{rms} on this trial.

Having demonstrated how the MTM framework produces trajectory estimates in individual trials, we now review the quantification and comparison of the average performance of the four decoders (RWM, STM, MTM_{M}, MTM_{DM}) across entire data sets. Figure 7.6 illustrates the following two main results, which hold true across both monkeys. First, a mixture of linear Gaussian trajectory models (MTM_{M}) provides lower decoding error than either of the nonmixture trajectory models (RWM and STM) (Wilcoxon paired-sample test, $p < 0.01$). Compared to the STM decoder, the MTM_{M} decoder reduced E_{rms} from 22.5 to 13.9 mm (22.8 to 11.8 mm) in monkey G (H). Second, the use of prior target information $P(m)$ in the MTM framework (MTM_{DM}) can further decrease decoding error (Wilcoxon paired-sample test, $p < 0.01$). Compared to the MTM_{M} decoder, the MTM_{DM} decoder reduced E_{rms} from 13.9 to 11.1 mm (11.8 to 10.5 mm) in monkey G (H). Because the MTM decoder is inherently parallelizable (as described in Section 7.3.2.1),

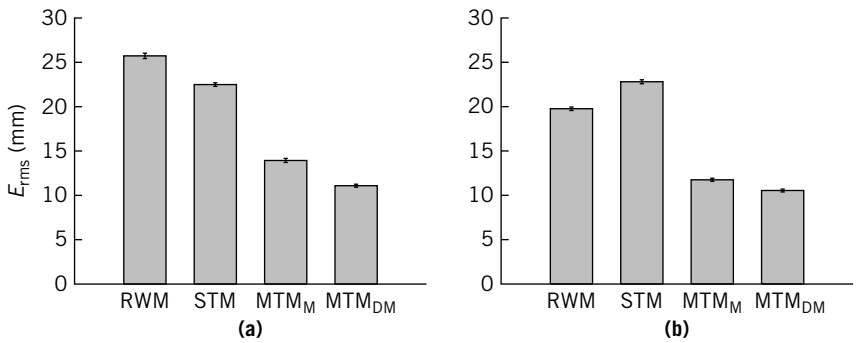

FIGURE 7.6

E_{rms} (mean ± SE) comparison for the RWM, STM, MTM_M, and MTM_{DM} decoders. (a) Monkey G (98 units). (b) Monkey H (99 units).
Source: Reproduced with permission from Yu et al., 2007.

these performance gains can be obtained *without* an associated increase in decoder running time. The superior performance of the MTM_M compared to the RWM and STM can be attributed to the fact that the MTM better captures the kinematics of goal-directed reaches (see Figure A2 in Yu et al. (2007)). If plan activity is available, this additional source of information can be naturally incorporated in the MTM framework to further improve decoding performance (MTM_{DM}).

7.4 DISCRETE DECODING FOR COMMUNICATION PROSTHESES

There has been considerable research on cortically controlled communications prostheses since at least the 1980s. This interest has been especially prevalent among the noninvasive community, and the intra-cortical community has recently begun to explore this domain as well. Using noninvasive EEG recordings, researchers have explored several approaches for engineering prostheses that allow discrete target selection. Some commonly studied categories of EEG signals include slow cortical potentials (SCPs) (Hinterberger et al., 2004), sensorimotor (μ and β) rhythms (McFarland et al., 2003; Wolpaw and McFarland, 2004a), and evoked potentials (P300) (Serby et al., 2005). More recently, minimally invasive communication prostheses relying on subdural electrocorticographic (ECoG) surface arrays

(Leuthardt et al., 2004; Schalk et al., 2008) and extra-cortical local field potentials (Kennedy et al., 2004) have been developed.

In the invasive intra-cortical electrode domain, Kennedy and colleagues demonstrated that just one or two neurons from the motor cortex of locked-in ALS patients could be used to move a cursor across a virtual keyboard to type out messages (Kennedy and Bakay, 1998; Kennedy et al., 2000b). Patients reported simply "imagining" or "thinking" about moving various parts of their bodies, and eventually the computer cursor itself, to guide the cursor.

In this section, we focus on decoding algorithms for communication prostheses based on large populations of neurons, typically recorded using intra-cortical electrode arrays. One possibility is to adapt a motor prosthesis from Section 7.3 for use as a communication prosthesis. Schwartz and colleagues demonstrated that the continuous trajectory output from a prosthetic system designed to reach to discrete targets could be processed by an algorithm that chooses the most likely target location (Taylor et al., 2003). If only a very early portion of the trajectory is needed to make the discrete classification, the system can simply cut the trial short and prepare to decode a new target.

More recent studies have shown that directly classifying the neural activity (rather than taking a two-stage approach of first estimating cursor position, then classifying) can lead to greater speed and accuracy of target selection (Shenoy et al., 2003; Musallam et al., 2004; Santhanam et al., 2006). These studies were performed using plan activity in medial intraparietal area (MIP), area 5 and PMd in rhesus monkeys. In these studies, the authors computed the most likely target $\widehat{m}$ given the observed neural activity $\mathbf{z}$:

$$\widehat{m} = \underset{m}{\operatorname{argmax}}\ P(m \mid \mathbf{z}) \tag{7.21}$$

where $P(m \mid \mathbf{z})$ is given in Eq. (7.20). The performance of the decoder (measured in terms of speed and accuracy of target selection) depends on many factors, including the duration and placement of the time window in which the spikes $\mathbf{z}$ are counted, the arrangement of targets in the monkey's workspace, and the spike count model $P(\mathbf{z} \mid m)$ (Eq. (7.20)) (Santhanam et al., 2006). The choice of $P(\mathbf{z} \mid m)$ is the topic of this section.

7.4.1 Independent Gaussian and Poisson Models

The most commonly used probabilistic models for spike counts are Gaussian (Eq. (7.19)) (Maynard et al., 1999; Hatsopoulos et al., 2004; Santhanam et al., 2006; Yu et al., 2007) and Poisson (Shenoy et al., 2003; Hatsopoulos et al.,

2004; Santhanam et al., 2006)

$$P(\mathbf{z} \mid m) = \prod_{i=1}^{q} P(z_i \mid m) \tag{7.22}$$

$$z_i \mid m \sim \text{Poisson}(\lambda_{i,m})$$

where $z_i \in \{0,1,2,\ldots\}$ is the spike count for neural unit $i \in \{1,\ldots,q\}$, and $\lambda_{i,m} \in \mathbb{R}_+$ is the mean spike count for unit i and reach target $m \in \{1,\ldots,M\}$. By construction, the units in Eq. (7.22) are assumed to be independent given the reach target. For the Gaussian model, it is common to make the same conditional independence assumption by constraining R_m in Eq. (7.19) to be diagonal.

The assumption of conditional independence in both the Gaussian (Eq. (7.19)) and Poisson (Eq. (7.22)) cases is largely for practical reasons. In the Gaussian case, there are typically too few training trials to fit a full covariance matrix, especially with a large number of units (many tens to hundreds). In other words, a full covariance matrix is underconstrained; this leads to overfitting of the training data, thereby reducing decoding performance for the test data. Furthermore, the full covariance that is fit is often not invertible, which is problematic when used for decoding in Eq. (7.21) (Maynard et al., 1999). In the Poisson case, there is no standard multivariate generalization.

Historically, Poisson-based algorithms have been found to perform better than Gaussian-based algorithms in decoding applications (Hatsopoulos et al., 2004; Santhanam et al., 2006). The observed spike counts are, by definition, non-negative integers and can be relatively small in some cases (e.g., <5 spikes). In this regime, a Poisson distribution is typically a better fit to the data than a Gaussian distribution. One might suspect that the data are not well suited for a Gaussian distribution, especially since a Gaussian distribution has nonzero probability density for noninteger values, as well as negative values. It has been shown that taking the square root of the counts can improve the Gaussian fit (Thacker and Bromiley, 2001). We employ this square root transform when using Gaussian-based decoding algorithms here.

AQ:1

7.4.2 Factor Analysis Methods

While the assumption of conditional independence provides a simple and effective decoder, we found that its performance began to deteriorate in a closed-loop setting where monkeys planned to different targets in rapid succession (Santhanam et al., 2006). Upon inspection of the neural activity recorded across multiple trials while the monkey planned to the same target,

we observed that the firing rates across the neural population fluctuated together in a systematic way that was dependent on chain position (i.e., the number of targets that had already been planned in rapid succession) (Gilja et al., 2005; Kalmar et al., 2005). In other words, it appeared that the activity of the units was *not* conditionally independent, which suggested that the conditional independence assumption may have been limiting the accuracy of the decoder. This provided the motivation for introducing conditional dependencies into the spike count model. However, as described above, using a fully dependent Gaussian distribution (i.e., a Gaussian distribution with a full covariance matrix) is problematic due to practical considerations. In this section, we explore a compromise by parameterizing the covariance matrix so as to introduce conditional dependencies without having to fit a full covariance matrix.

In addition to chain position, there is evidence that other factors may modulate PMd plan activity from trial to trial (for the same reach target) in a systematic way across the neural population. These factors include reach curvature (Hocherman and Wise, 1991), reach speed (Churchland et al., 2006b), force (Riehle and Requin, 1994), the required accuracy (Gomez et al., 2000), and the type of grasp (Godschalk et al., 1985). There may also be unobserved influences, including attentional states (Musallam et al., 2004; Chestek et al., 2007) that further modulate cortical activity, even when observable behavior is held fixed. These various influences may inadvertently mask the signal of interest (i.e., reach target) and thereby reduce decoding accuracy.

Figure 7.7(a) provides an example of the trial-to-trial variability present in the activity of two recorded neural units. For each unit and reach target, the distribution of plan activity across many trials is shown. Note the large variability in spike counts across trials for each target, which results in overlap between the ranges of spike counts for the reach target. Part of this variability can be attributed to the multitude of potential factors described above. These factors presumably affect the activity of each unit in a different, yet reliable, way (e.g., as the intended reach speed increases, the firing rate for unit 1 increases while that for unit 2 decreases) and are thus *shared* across the neural population. The other part of this variability, which will simply be termed "spiking noise," can arise from various sources, including channel noise in the spiking process, variability in active dendritic integration, synaptic failure, general quantal (in the vesicle sense) release variation, and axonal failure in incoming spikes (Faisal et al., 2008). These sources of variability presumably yield nonsystematic, and therefore *independent*, perturbations across the neural population. The challenge for the decoder is to avoid mistaking these variations (both shared and independent) as being the signature for an entirely different reach target.

Figure 7.7(b) shows simulated data for two neurons that exhibit independent variability and no shared variability for a particular reach target. The spike counts for each neuron are different in each trial, but there is no inherent correlation between the observations with regard to the magnitude or polarity of these perturbations. In contrast, Figure 7.7(c) shows simulated data from two neurons that exhibit shared variability *and* independent variability. The inset shows a hypothetical factor, such as intended reach speed, undergoing modulation across trials for the same reach target (dark gray points correspond to fast reaches; medium gray dots, to slow reaches). In this construction, when

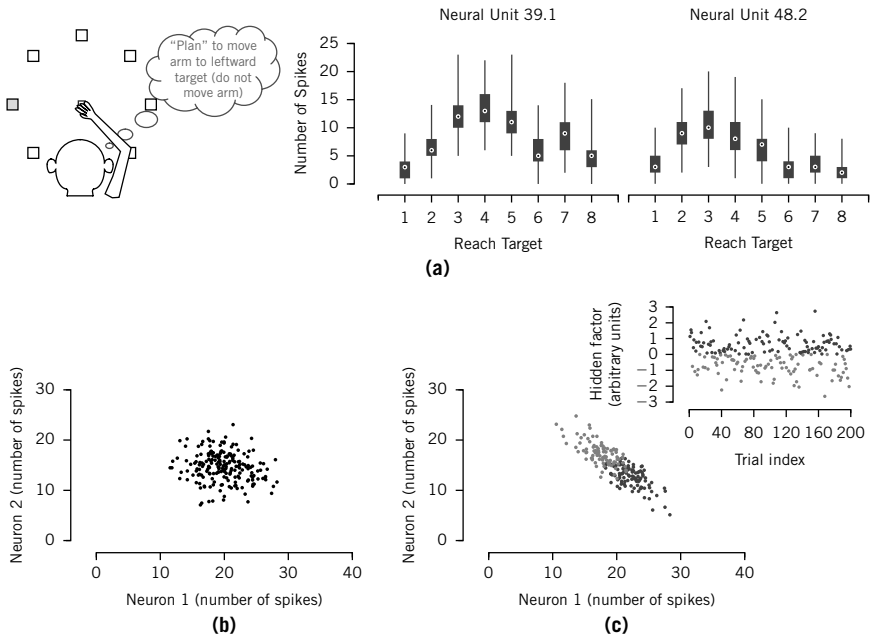

FIGURE 7.7

(a) Schematic of delayed reach task where subject reaches to one of eight reach targets. Data from two real neural units are shown where spikes were counted in a 200-ms window after target presentation for each trial. Trials were segregated by reach target, and a box plot was generated denoting the median spike count (white circles), the 25th to 75th percentile of the spike counts (dark gray rectangles), and the ranges of the outliers (thin "whiskers") for each target. (b) Simulation of two neurons whose trial-to-trial spike counts were perturbed independently of one another. (c) Simulation of two neurons whose spike counts were primarily perturbed by a shared latent factor (see inset). The diagonal orientation relative to the main axes is a tell-tale sign of a correlated relationship between these two neurons. Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

Source: Reproduced with permission from Santhanam et al., 2009.

the speed is greater than average, neuron 1 emits slightly more spikes than average and neuron 2 emits slightly fewer than average. When the factor is below average, the reverse is true. By assuming conditional independence (Eqs. (7.19) and (7.22)), both the shared and independent sources of variability are treated as independent across the neural population. However, if we explicitly account for this shared component of variability, we can potentially increase decoding accuracy, as discussed below.

In reality, we do not have direct access to the inset in Figure 7.7(c), since many different uncontrollable factors may be involved and many of them are simply unobservable (e.g., cognitive attentiveness to the task). Although we cannot explicitly identify these factors, we attempted to infer a set of shared underlying factors for each trial, along with the mapping between these factors and the observed data. By "mapping," we mean the precise polarity and magnitude by which a factor perturbs the response of a particular neuron. Such mapping defines the systematic relationship between the latent factors and the observed neural activity, providing a model of how the spike counts of a population of neurons are expected to covary.

7.4.2.1 *Modeling Shared Variability Using Factor Analysis*

For a particular reach target, the independent Gaussian (Eq. (7.19)) and Poisson (Eq. (7.22)) models treat all trial-to-trial variability as independent between neural units. A model that could successfully capture spike count correlations across the population might well yield a more accurate decoding algorithm. As discussed above, fitting a Gaussian model with a full covariance matrix to a large number of units would require an impractically large number of trials. Instead, we recently proposed to use a parametrized form of covariance matrix (Santhanam et al., 2009), which models correlations in a limited number of principal directions. This is the central topic of this section. The form we used derives from a standard statistical technique called factor analysis (FA) (Everitt, 1984), where the observed quantities are modeled as a linear function of unobserved, shared factors.

For our prosthetic system, the observed variables are the neural spiking data that we recorded from the electrode array. The underlying factors represent the physical, cognitive, or cortical network states of the subject that are uncontrolled or effectively uncontrollable. We used the larger number of observed variables to help triangulate the smaller number of unobserved factors in the system. FA is a well-established technique of multivariate statistics, and we review the method here in the context of our prosthetic decoding application.

For each experimental trial, as before, we collected the (square root of the) number of spikes observed from each neural unit within the neural integration

window into the q-dimensional vector $\mathbf{z}$. We also considered an abstract set of latent factors, which were assumed to be Gaussian-distributed and assembled into a vector $\mathbf{x} \in \mathbb{R}^{p \times 1}$. Without loss of generality, we took these p factors to be independent and to have unit variance around $\mathbf{0}$.[1] The observation $\mathbf{z}$ was also taken to be Gaussian-distributed, but with a mean that depended on the latent factors $\mathbf{x}$. The complete model is as follows:

$$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, I) \tag{7.23}$$

$$\mathbf{z} \mid \mathbf{x}, m \sim \mathcal{N}(C_m \mathbf{x} + \boldsymbol{\mu}_m, R_m) \tag{7.24}$$

As in Eq. (7.19), $\boldsymbol{\mu}_m \in \mathbb{R}^{q \times 1}$ contains the number of spikes that each neural unit would produce for target $m \in \{1, \ldots, M\}$ if there were only target-related variability in the system, and $R_m \in \mathbb{R}^{q \times q}$ is a diagonal matrix that captures the independent variability present in $\mathbf{z}$ from trial to trial. However, there is now an additional mechanism by which trial-to-trial variability can emerge in the neural observations: specifically, the neural data vector $\mathbf{z}$ is perturbed by a linear function of the underlying factors. The matrix $C_m \in \mathbb{R}^{q \times p}$ defines the mapping between the factors and the observations. Given that the underlying factors are shared between the observed neural units, there is a predefined correlation structure between the neural units built into the model. Integrating over all possible $\mathbf{x}$, we obtained the likelihood of the data $\mathbf{z}$ for a reach target m:

$$\mathbf{z} \mid m \sim \mathcal{N}\left(\boldsymbol{\mu}_m, C_m C_m' + R_m\right) \tag{7.25}$$

Equation (7.25) shows that FA effectively partitions the data covariance matrix into two components. The first term in the covariance, $C_m C_m'$, captures the *shared* variability across the neural population. The second term, R_m, captures the remaining unexplained variability, which is assumed to be *independent* across units. It may include biophysical spiking noise and other nonshared sources of variability. Note that FA can be viewed as a generalization of principal component analysis (PCA). As the diagonal elements of R_m approach zero, the space spanned by the columns of C_m will be equivalent to the space found by PCA (Roweis and Ghahramani, 1999).

[1] The following is why there is no loss of generality when assuming $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, I)$. Suppose that $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, V)$ for some $V \neq I$. In other words, suppose that the latent factors are dependent and have nonunit variance. This model is formally equivalent to one in which $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, I)$ and $\mathbf{z} \mid \mathbf{x}, m \sim \mathcal{N}\left(C_m V^{\frac{1}{2}} \mathbf{x} + \boldsymbol{\mu}_m, R_m\right)$, in the sense that both models would give the same decoded target $\widehat{m}$ and data likelihood $P(\mathbf{z})$.

7.4.2.2 *Decoding Reach Targets Using Factor Analysis*

The model parameters ($\boldsymbol{\mu}_m$, C_m, R_m for each target m) are fit by maximum likelihood using the neural data $\mathbf{z}$ and the reach target m across all trials in the training data set. Although $P(\mathbf{z} \mid m)$ is Gaussian (Eq. (7.25)), the parametric form of the covariance makes direct calculation of the parameters difficult. Instead, we applied the expectation-maximization (EM) algorithm (Dempster et al., 1977), which iteratively updates the model parameters and latent factors until convergence is reached. The update equations for fitting FA using EM can be found elsewhere (Roweis and Ghahramani, 1999; Santhanam et al., 2009).

When decoding test trials, we used the maximum likelihood estimator previously described (cf. Eqs. (7.20) and (7.21)), where $P(\mathbf{z} \mid m)$ is given in Eq. (7.25). We refer to this approach as FA_{sep} since there is a separate C_m and R_m per reach target m.[2] To reiterate, the FA model provides a method by which we can approximate the covariance structure of our observed variables without fitting a full covariance matrix. Once we have fit the model, the decoding procedure does not rely on finding the underlying factor vector $\mathbf{x}$, but rather operates solely with the distribution $P(\mathbf{z} \mid m)$.

An important question is how to select p, the number of underlying factors. With too many factors in the model, the model parameters will be fit to any idiosyncratic correlation structure in the training data and these features may not appear in the test data. If p is too small (i.e., too few factors), the model will be unable to capture all underlying correlation present in the training data. One must ensure that the model is not overfit to the training data and that it generalizes well for test data. Selecting the optimal p, or p^*, is part of the process of model selection. It is important to keep in mind that the underlying factors identified for each model are abstract. They are a vehicle by which we are able to describe shared variability—the exact number of factors selected does not necessarily have physical significance. The model selection procedure is a simple, necessary step to ensure that a model fit with the training trials will perform well when decoding the test data.

We used the standard approach of partitioning data into training and validation sets to find the value of p beyond which overfitting became a problem (Hastie et al., 2003). Each data set was split approximately into two equal halves, one to serve as a training set and the other as a test set. The trials in the training set were further divided in order to perform a 5-fold or 10-fold cross-validation. For example, in 5-fold cross-validation, roughly four-fifths

[2]Note that the overall model for $\mathbf{z}$ is very similar to the "mixture of factor analyzers" proposed by Ghahramani and Hinton (1997), except that FA_{sep} allows for a separate covariance R_m per mixture component.

of the training trials would be used to train the FA model and one-fifth to validate the model's prediction error. This would be repeated five times until all of the original training set was used for validation. A series of models, each with a different number of underlying factors p, were tested using this cross-validation method. The p^* was selected by identifying the model with the lowest decoding error. The performance of this one model was then assessed on the test data to compare the classification performance of the optimal FA model against the independent Gaussian and Poisson models.

7.4.2.3 A Unified Latent Space

The FA_{sep} approach in Eq. (7.25) is a straightforward extension of the independent Gaussian model of (Eq. (7.19)). It defines a separate latent space for each reach target indexed by m (Eq. (7.24)). To visualize the data points across different reach targets in a unified latent space, we developed a second, novel approach to FA that defines a single output mapping and incorporates the separation of reach targets through the shared latent space (Santhanam et al., 2009). The model is

$$\mathbf{x} \mid m \sim \mathcal{N}(\boldsymbol{\mu}_m, I) \tag{7.26}$$

$$\mathbf{z} \mid \mathbf{x} \sim \mathcal{N}(C\mathbf{x}, R) \tag{7.27}$$

where $\boldsymbol{\mu}_m$ is now a p-dimensional vector that describes the reach target influence on the neural data within the lower-dimensional space. The mapping between the low- and high-dimensional spaces (defined by $C \in \mathbb{R}^{q \times p}$) is shared across reach targets, thereby unifying the effect of target-related and non-target-related factors on neural data. For each trial, the shared target-related effect on the neural observations is $C\boldsymbol{\mu}_m$, while the shared non-target-related factors perturb the observations by $C(\mathbf{x} - \boldsymbol{\mu}_m)$. As before, R is diagonal and describes the independent variability. We refer to this model as FA_{cmb} , since the output mapping is "combined" across reach targets.

We fit the model parameters ($\boldsymbol{\mu}_m$ for each target m, C, R), again using the EM algorithm. The EM update equations for FA_{cmb} can be found in Santhanam et al. (2009). To decode the reach target for test trials, we rewrote Eqs. (7.26) and (7.27) as

$$\mathbf{z} \mid m \sim \mathcal{N}\left(C\boldsymbol{\mu}_m, CC' + R\right) \tag{7.28}$$

and applied the same maximum likelihood decoder shown in Eqs. (7.20) and (7.21).

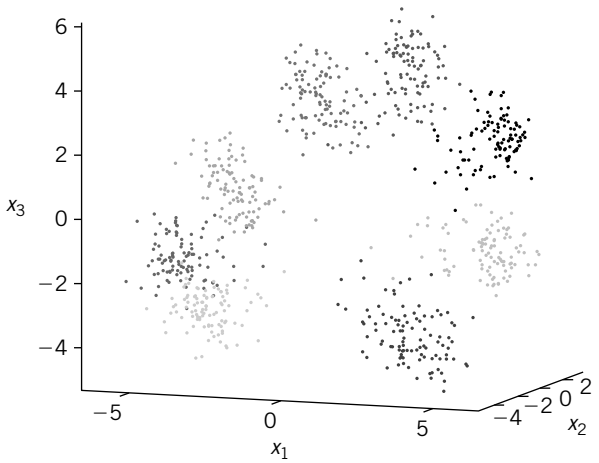

FIGURE 7.8

Visualization of FA_{cmb} with a three-dimensional latent space. Each point corresponds to the inferred $\mathbf{x}$ for a given trial. Axis scaling is determined by the fitted model. The various shading of the data points denotes the upcoming reach target. Clusters correspond to different reach targets. Analysis performed on data set G20040428. Please see this figure in color at the companion web site www.elsevierdirect.com /companions/9780123750273. *Source*: Reproduced with permission from Santhanam et al., 2009.

Figure 7.8 is an example of how these unified factors might manifest themselves over many trials.[3] There are three latent factors that capture both target and non-target-related influences on the neural data $\mathbf{z}$ (which is not shown and is much higher-dimensional). Each differently shaded cloud of points corresponds to a different reach target, with trials to a given target clustered next to one another. Note that even in this three-dimensional instantiation of FA_{cmb} , there is visible separation between the clouds, suggesting that it is possible to discriminate between targets.

By comparing Eqs. (7.25) and (7.28), we can gain further insight into the relationship between FA_{sep} and FA_{cmb}. Both models define a mixture of M Gaussians in the high-dimensional space in which $\mathbf{z}$ lies. In both cases, the Gaussian covariances take the same parameterized form of low-rank plus diagonal. Whereas FA_{sep} can have a different covariance for each reach target indexed by m (e.g., each Gaussian cloud can be oriented in a different direction), they are constrained to be the same in FA_{cmb}. In FA_{sep}, the Gaussians

[3]We fix the number of factors in this example to be three to allow for convenient plotting of the data; ordinarily, the number of factors (p) would be optimized using the model selection procedure described earlier.

can be centered anywhere in the high-dimensional space. In FA_{cmb}, however, they must lie within the lower-dimensional space spanned by the columns of C, which is also used to describe the shared variability. In other words, the single, low-dimensional space in FA_{cmb} must be large enough (i.e., have a high enough dimensionality) to capture both between-class and within-class spread (where the latter is synonymous with shared variability). In FA_{sep}, each of the M low-dimensional spaces needs only to be large enough to capture the within-class spread for that reach target. As will be reviewed in the next section, we found that the optimal latent dimensionality of FA_{cmb} was typically larger than that of FA_{sep}.

7.4.3 Results

We first chose the number of latent factors (p) that maximized cross-validated decoding performance. Conceptually, we had to ensure that we had enough factors to sufficiently describe the shared variability in the data but not so many that we overfit the training data. This was a necessary step before we could compare FA models to the independent Gaussian and Poisson models. We examined two data sets, `G20040401` and `H20040916`, which comprised 185 and 200 total trials per reach target, respectively. This yielded ~80 training and ~20 validation trials per reach target for each cross-validation fold. A spike count window of 250 ms was used for this particular analysis, although we saw similar results for smaller spike count windows as well.

Figure 7.9(a) shows the validation performance for these two data sets as a function of p when fitting the FA_{sep} algorithm. The performance for $p = 0$ corresponds to an independent Gaussian model where there is no C_m matrix and only a diagonal covariance matrix, R_m, per reach target. The best validation performance for FA_{sep} was obtained for low p: $p^* = 1$ for `G20040401` and $p^* = 3$ for `H20040916`. The results show that as p was increased beyond p^*, performance steadily declined. Overfitting was an issue for even relatively small values of p. Why might this be? There are surely many factors that influence the upcoming reach—direction, distance, curvature, speed, force, and so forth. However, for our data set, reach trajectories to the same target were highly similar. Moreover, the shared variability (captured by trial-to-trial modulation of the factors) may be small relative to the independent variability. Given a small number of training trials for each reach target (~80) relative to the number of units (~100), it is likely that we were unable to identify more latent factors without poorly estimating the model parameters and overfitting. The penalty for overfitting is evident: by $p = 30$, all of the benefits from using FA_{sep} over the independent Gaussian approach ($p = 0$) are lost.

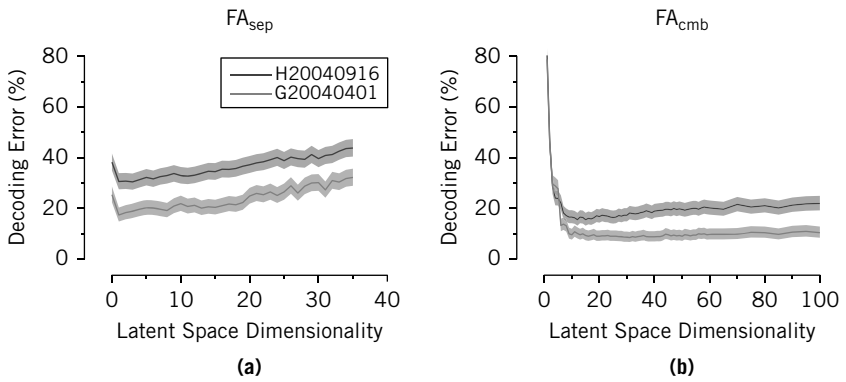

FIGURE 7.9

Model selection for FA_{sep} and FA_{cmb}. The shaded area denotes the 95% confidence interval (Bernoulli process) around the mean performance (embedded line). (a) Optimal cross-validated performance for FA_{sep} occurs for a low number of factors. (b) Optimal cross-validated performance for FA_{cmb} occurs between 10 and 20 factors. Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

Source: Reproduced with permission from Santhanam et al., 2009.

We performed the same sweep of p for FA_{cmb}, and the validation performance is plotted in Figure 7.9(b). The best validation performance for the combined FA model was obtained for higher p than for FA_{sep}: $p^* = 31$ for G20040401 and $p^* = 12$ for H20040916. This makes sense because the latent space for FA_{cmb} must capture both within-class and between-class spread, whereas each of the M latent spaces for FA_{sep} need only capture within-class spread, as previously described. While at first it seems that the optimal number of factors for G20040401 is substantially higher than that for H20040916, one can see that the validation performance reached a floor around $p^* = 15$ for the G20040401 data set. There was negligible improvement when the dimensionality was increased past that point. This, combined with the fact that performance degradation due to overfitting in the H20040916 data set was relatively minor, demonstrated that FA_{cmb} is less sensitive to the exact choice of p than is FA_{sep}

Next, we compared the performance of four different decoders grouped into two pairs. One pair consisted of the independent Gaussian and Poisson models. The second pair was the FA-based models, FA_{sep} and FA_{cmb}. We computed decoding accuracy based on a spike count window of 250 ms. For the independent algorithms, half of the trials in each data set were used to train the model parameters, and the model was used to predict the reach target for the other half. For the FA-based algorithms, the training set was first used

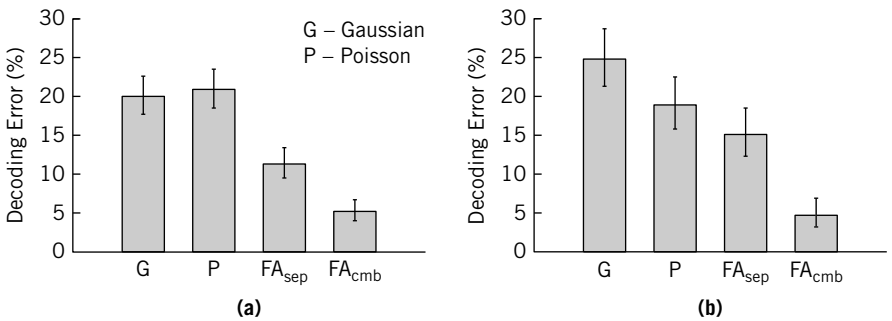

FIGURE 7.10

Comparison of decoding algorithms on two example data sets. Error bars denote the 95% confidence interval (Bernoulli process) for each measurement. (a) Data set G20040429. (b) Data set H20040928.

Source: Reproduced with permission from Santhanam et al., 2009.

to select the optimal dimensionality p^* using 5-fold cross-validation, and the training set was then used in its entirety to fit a model with p^* dimensionality. As with the independent algorithms, the fitted model was finally applied to the test data to compute the average decoding performance.

Figure 7.10 shows the relative performance of these four algorithms. There were a grand total of 1024 and 528 test trials for the G20040429 and H20040928 data sets, respectively. For data set G20040429, the independent Gaussian and Poisson algorithms had roughly the same performance, while for data set H20040928 the independent Gaussian algorithm had the worst performance, with $\sim$5% worse accuracy than that of the independent Poisson algorithm. The FA_{sep} algorithm reduced the decoding error for monkey G's data set by nearly 10%, but showed only a modest improvement for monkey H's data set. The FA_{cmb} algorithm showed by far the best performance. Our novel decoder was able to drastically reduce the decoding error to only $\sim$5%.

7.5 DISCUSSION

We described two classes of neural prosthetic systems—motor and communication—that aim to assist disabled patients by translating their thoughts into actions in the outside world. While many challenges remain (Andersen et al., 2004; Schwartz et al., 2006; Donoghue, 2008; Ryu and Shenoy, 2009), one of the major challenges is interpreting, or decoding, the neural signals. In this chapter, we reviewed current decoding approaches and detailed advances that we recently published (Yu et al., 2007; Santhanam et al., 2009). For motor

prostheses, we showed that arm trajectories can be more accurately decoded by probabilistically mixing trajectory models, each of which is accurate within a limited regime of movement. The MTM is a general approach to constructing trajectory models that can exhibit rather complex dynamical behaviors, whose decoder can be implemented to have the same running time as as those of simpler trajectory models. Furthermore, prior information about the identity of the upcoming movement regime can be incorporated in a principled way. For communication prostheses, we showed that the reach target can be more accurately decoded by taking into account the correlation structure across the neural population. These correlations arise because of unobserved or uncontrolled aspects of the behavioral task, possibly due to the subject being more attentive or more fatigued. Using factor analysis methods, we exploited this statistical structure in a maximum likelihood classifier.

For goal-directed reaches, the observed neural activity provides two categorically different types of information about the arm trajectory to be estimated. One type is informative of the moment-by-moment details of the arm trajectory (dynamic); the other is informative of the identity of the upcoming reach target (static). If the moment-by-moment details of the arm trajectory can be decoded perfectly using only the neural activity present during movement, there is no need for target information in motor prostheses. However, the moment-by-moment details of the arm trajectory and the target identity are each decoded with varying levels of uncertainty. When both types of information are available, it is desirable to combine them in a way that takes into account their relative uncertainty and yields a coherent arm trajectory estimate. The MTM framework provides a principled way to combine these two types of information.

In this work, we extracted both types of information from the same cortical areas: PMd and M1. The type of information being decoded depends on *when* the neural activity occurs relative to the reach, which we assumed to be known. In settings where the subject is free to decide when to reach, it will be necessary to implement a state machine (Shenoy et al., 2003; Achtman et al., 2007; Kemere et al., 2008) that determines the type of information being conveyed by the neural activity at each time point.

The MTM can be viewed as a discrete approximation of a single, unified trajectory model with nonlinear dynamics. One may consider directly decoding using a nonlinear trajectory model without discretization. Although this makes the decoding problem more difficult, there are effective approximate methods that can be applied (Yu et al., 2006b, 2008). In addition to the MTM and decoding using a nonlinear model, other important extensions to the basic Kalman filter include a switching state model (Srinivasan et al., 2007), a switching observation model (Wu et al., 2004a; Srinivasan et al.,

2007), and an adaptive Kalman filter whose parameters change over time (Wu and Hatsopoulos, 2008).

The advances presented in Section 7.4 can be applied in a straightforward way to the MTM. In Section 7.3, we used the independent Gaussian model (Eq. (7.19), assuming conditional independence across different units) to obtain the prior target information $P(m \mid \mathbf{z})$ (Eq. (7.20)). The results from Section 7.4 suggest that a more accurate target prior can be obtained by using either a Poisson model (Eq. (7.22)) or a Gaussian model that captures the correlated activity across the neural population (Eq. (7.25) or (7.28)). This could provide performance improvements beyond that shown in Section 7.3.

In Section 7.4, we introduced latent variables to capture unobserved or uncontrolled aspects of the behavioral task (e.g., the subject's level of attention or fatigue) in the context of discrete decoding for communication prostheses. The same idea can be applied to continuous decoding for motor prostheses. Paninski and colleagues showed that a Kalman filter with latent variables can outperform the standard Kalman filter (Wu et al., 2009).

For discrete decoding, we found that the independent Poisson model yielded similar or better decoding accuracy compared to the independent Gaussian model (see Figure 7.10). It is natural to ask whether the accuracy of the independent Poisson model can be improved by if one takes into account correlated activity across the neural population by replacing Eq. (7.24) or (7.27) with a Poisson observation model. In Santhanam et al. (2009), we developed a family of models called "factor analysis with Poisson output" (FAPO). We found that, while these models often outperformed the independent Poisson model, they tended to have higher levels of decoding error than their Gaussian counterparts ($\mathrm{FA}_{\mathrm{sep}}$ and $\mathrm{FA}_{\mathrm{cmb}}$). There are several possible explanations for this result, including the fixed mean–variance relationship for a Poisson distribution and the approximations required for fitting FAPO models (Santhanam et al., 2009).

Although we sought to maximize classification performance in Section 7.4, we trained a generative model by maximizing the likelihood of the observed data. One would hope for a monotonic relationship between classification performance and data likelihood, but there is no guarantee that this will be the case. On the other hand, discriminative classifiers, such as support vector machines (Vapnik, 1995), directly maximize classification performance. One advantage of using a generative model is that we can recover and attempt to interpret the latent variables (see Figure 7.8), whereas discriminative methods tend to yield a "black box" that is difficult to interrogate. Jordan and colleagues compared these two types of classifiers and found that generative classifiers tended to outperform discriminative classifiers in the regime of small training set sizes, while the reverse was true for larger training set sizes

(Ng and Jordan, 2002). We leave the exploration of discriminative classifiers for communication prostheses, and their comparison to generative classifiers, as a topic of future work.

7.6 FUTURE DIRECTIONS

There has been considerable progress in the development of neural prosthetic systems in recent years, including FDA-approved clinical trials (Hochberg et al., 2006; Kim et al., 2008). However, many critical research challenges remain before invasive, electrode-based systems are ready for "clinical prime time" (Donoghue et al., 2007; Ryu and Shenoy, 2009). Among them are increasing decoding performance, minimizing surgical invasiveness, increasing electrode lifetime and recording stability (Polikov et al., 2005), and developing fully implantable, low-power electronics (Sodagar et al., 2007; Chestek et al., 2009; Harrison et al., 2009). In this chapter, we considered only the first of these challenges, neural decoding.

Within the scope of neural decoding, there remain many challenges, including (1) furthering our understanding of the time course of plan and movement activity and their relationship to arm movement, and (2) designing decoders that are effective in a closed-loop setting. For the first challenge, in both Sections 7.3 and 7.4 we collapsed the timecourse of plan activity by taking spike counts within a large time window. However, from Figure 7.2, it appears that the time-varying nature of plan activity can, in principle, be leveraged to improve the prior target information (Section 7.3) and target classification (Section 7.4). For movement activity, most decoders (including population vectors, linear filters, and all recursive Bayesian decoders mentioned in this work) assume a simple relationship (e.g., linear cosine tuning or some variant thereof) between arm state and neural activity. However, there is mounting evidence that this relationship is more complex than previously appreciated (Churchland and Shenoy, 2007b; Scott, 2008), and leads to model mismatch (see Figure A2 in Yu et al., 2007) in the training phase and subsequent decoding error in the testing phase. In brief, we need methods for studying the time course of plan and movement activity across a neural population that does not collapse neural activity across a large window, or assume an overly restrictive relationship with arm movement. One possible approach is to extract a low-dimensional *neural trajectory* that summarizes the activity recorded simultaneously from many units (Yu et al., 2006a, 2009; Churchland et al., 2007, 2010). An advantage with this approach is that neural trajectories are extracted using neural activity alone, and no assumption needs

to be made about the relationship between the neural activity and arm movement. Figure 7.11 shows neural trajectories extracted from 61 units in PMd and M1 during a delayed reach task. Each trace corresponds to a single trial of planning and executing a reach to the upper-right target (medium gray traces) or lower-left target (light gray traces). For different reach targets, the trajectories head off toward different parts of the low-dimensional space in a reliable fashion. After the neural trajectories are extracted, they can then be related to the subject's behavior, such as arm kinematics and muscle activity (Afshar et al., 2008). Such analyses should provide insight into how the time course of plan activity should be incorporated and how to better capture the relationship between movement activity and desired arm movement in prosthetic decoders.

For the second challenge, Kass and colleagues have shown that the open-loop performance of decoding algorithms does not necessarily reflect their closed-loop performance (Chase et al., 2009). The reason is that the subject receives visual (and potentially other sources of) feedback in a closed-loop

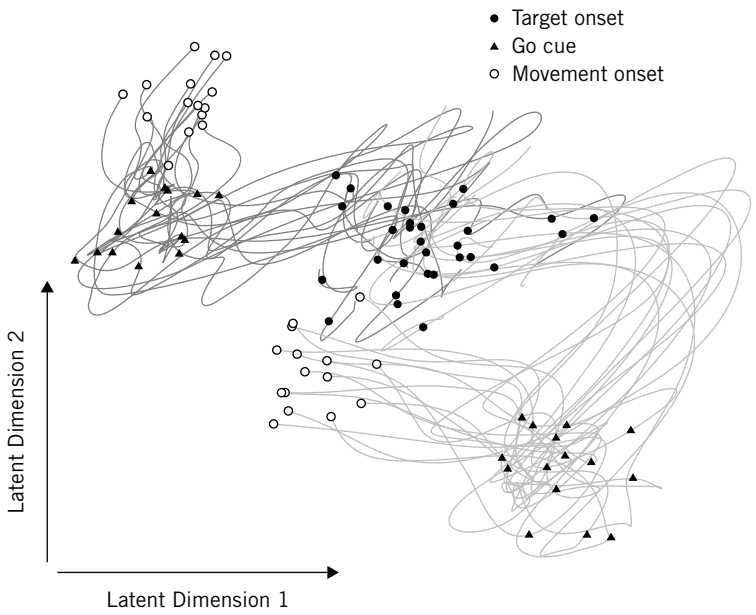

FIGURE 7.11

Neural trajectories extracted by Gaussian-process factor analysis (GPFA). Each trace corresponds to a single trial of planning and executing a reach to the upper-right target (dark gray traces) or the lower-left target (light gray traces). Indicated are time of reach target onset (black dots), go cue (triangles), and movement onset (open circles). Please see this figure in color at the companion web site www.elsevierdirect.com/companions/9780123750273.

setting, which enables the subject to compensate for systematic errors in the decoded cursor movements. Furthermore, there is increasing evidence that neurons can change their firing properties between open-loop and closed-loop settings (Taylor et al., 2002; Carmena et al., 2003), and that the brain can learn the mapping between neural activity and cursor movement (Jarosiewicz et al., 2008; Ganguly and Carmena, 2009). This suggests that we not only should carefully compare existing decoders in a closed-loop setting (Kim et al., 2008; Chase et al., 2009), but should also design decoding algorithms that are specifically tailored for closed-loop use. Currently missing from decoding algorithm development are control theoretic concepts (Scott, 2004), such as feedback, signal delays (in the nervous system and in the signal processing machinery of the prosthetic system), and system stability. To design a decoder that accounts for sensory feedback, we need to develop a deeper understanding of how the subject alters his neural activity based on feedback received to achieve task success. Because the cursor motion is different from one experimental trial to the next, simply averaging the neural activity across trials will likely obscure neural signatures of the brain attempting to correct for errors in cursor motion. Instead, we need statistical methods that can characterize the time course of neural population activity on a *single-trial* basis (Yu et al., 2006a, 2009; Churchland et al., 2007, 2010). Such methods will allow us to study (1) how corrections in cursor motion are brought about by changes in the pattern of neural population activity, and (2) how the pattern of neural activity shifts when transitioning from an open-loop to a closed-loop setting.

Acknowledgments

We thank Stephen Ryu, Afsheen Afshar, Caleb Kemere, Vikash Gilja, and Teresa Meng for their contributions to the original research articles that are reviewed in this chapter, John Cunningham for his contributions to the development of the GPFA algorithm (Figure 7.11), Sandra Eisensee for administrative assistance, Melissa Howard for surgical assistance and veterinary care, and Drew Haven for computing support.

This work was supported by NIH-NINDS-CRCNS 5-R01-NS054283, NDSEG Fellowships, NSF Graduate Research Fellowships, Gatsby Charitable Foundation, Christopher and Dana Reeve Foundation, Burroughs Wellcome Fund Career Award in the Biomedical Sciences, Stanford Center for Integrated Systems, NSF Center for Neuromorphic Systems Engineering at Caltech, Office of Naval Research, Sloan Foundation and Whitaker Foundation.

References

Achtman, N., Afshar, A., Santhanam, G., Yu, B.M., Ryu, S.I., Shenoy, K.V., 2007. Free paced high-performance brain-computer interfaces. J. Neural. Eng. 4, 336–347.

Afshar, A., Yu, B.M., Santhanam, G., Ryu, S.I., Shenoy, K.V., Sahani, M., 2008. Evidence for smooth and inertial dynamics in the evolution of neural state. Soc. Neurosci. Abstr. (319.8).

Andersen, R.A., Burdick, R.W., Musallam, S., Pesaran, B., Cham, J.G., 2004. Cognitive neural prosthetics. Trends Cogn. Sci. 8, 486–493.

Batista, A.P., Santhanam, G., Yu, B.M., Ryu, S.I., Afshar, A., Shenoy, K.V., 2007. Reference frames for reach planning in macaque dorsal premotor cortex. J. Neurophysiol. 98, 966–983.

Boussaoud, D., Wise, S.P., 1993. Primate frontal cortex: neuronal activity following attentional versus intentional cues. Exp. Brain. Res. 95 (1), 15–27.

Brockwell, A.E., Rojas, A.L., Kass, R.E., 2004. Recursive Bayesian decoding of motor cortical signals by particle filtering. J. Neurophysiol. 91 (4), 1899–1907.

Brown, E.N., Frank, L.M., Tang, D., Quirk, M.C., Wilson, M.A., 1998. A statistical paradigm for neural spike train decoding applied to position prediction from the ensemble firing patterns of rat hippocampal place cells. J. Neurosci. 18 (18), 7411–7425.

Carmena, J.M., Lebedev, M.A., Crist, R.E., O'Doherty, J.E., Santucci, D.M., Dimitrov, D.F., et al., 2003. Learning to control a brain-machine interface for reaching and grasping by primates. PLoS Biol. 1 (2), 193–208.

Chan, S.S., Moran, D.W., 2006. Computational model of a primate arm: from hand position to joint angles, joint torques and muscle forces. J. Neural. Eng. 3, 327–337.

Chapin, J.K., Moxon, K.A., Markowitz, R.S., Nicolelis, M.A.L., 1999. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. Nat. Neurosci. 2, 664–670.

Chase, S.M., Schwartz, A.B., Kass, R.E., 2009. Bias, optimal linear estimation, and the differences between open-loop simulation and closed-loop performance of spiking-based brain-computer interface algorithms. Neural Netw. 22 (9), 1203–1213.

Chestek, C.A., Batista, A.P., Santhanam, G., Yu, B.M., Afshar, A., Cunningham, J.P., et al., 2007. Single-neuron stability during repeated reaching in macaque premotor cortex. J. Neurosci. 27, 10742–10750.

Chestek, C. A., Gilja, V., Nuyujukian, P., Kier, R. J., Solzbacher, F., Ryu, S. I., Harrison, R. R., and Shenoy, K. V. (2009). HermesC: Low-power wireless neural recording system for freely moving primates. IEEE Trans Neural Syst Rehabil Eng. 17 (4), 330–338.

Churchland, M.M., Afshar, A., Shenoy, K.V., 2006a. A central source of movement variability. Neuron 52, 1085–1096.

Churchland, M.M., Santhanam, G., Shenoy, K.V., 2006b. Preparatory activity in premotor and motor cortex reflects the speed of the upcoming reach. J. Neurophysiol. 96 (6), 3130–3146.

Churchland, M.M., Shenoy, K.V., 2007a. Delay of movement caused by disruption of cortical preparatory activity. J. Neurophysiol. 97, 348–359.

Churchland, M.M., Shenoy, K.V., 2007b. Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex. J. Neurophysiol. 97, 4235–4257.

Churchland, M.M., Yu, B.M., Ryu, S.I., Santhanam, G., Shenoy, K.V., 2006c. Neural variability in premotor cortex provides a signature of motor preparation. J. Neurosci. 26 (14), 3697–3712.

Churchland, M.M., Yu, B.M., Sahani, M., Shenoy, K.V., 2007. Techniques for extracting single-trial activity patterns from large-scale neural recordings. Curr. Opin. Neurobiol. 17 (5), 609–618.

Churchland, M.M., Yu, B.M., Cunningham, J.P., Sugrue, L.P., Cohen, M.R., Corrado, G.S., Newsome, W.T., Clark, A.M., Hosseini, P., Scott, B.B., Bradley, D.C., Smith, M.A., Kohn, A., Movshon, J.A., Armstrong, K.M., Moore, T., Chang, S.W., Snyder, L.H., Lisberger, S.G., Priebe, N.J., Finn, I.M., Ferster, D., Ryu, S.I., Santhanam, G., Sahani, M., and Shenoy, K.V., 2010. Stimulus onset quenches neural variability: a widespread cortical phenomenon. Nat Neurosci. 13 (3), 369–378.

Crammond, D.J., Kalaska, J.F., 2000. Prior information in motor and premotor cortex: activity during the delay period and effect on pre-movement activity. J. Neurophysiol. 84, 986–1005.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). J. R. Stat. Soc. Ser. B 39, 1–38.

Donoghue, J.P., 2008. Bridging the brain to the world: a perspective on neural interface systems. Neuron 60, 511–521.

Donoghue, J.P., Nurmikko, A., Black, M., Hochberg, L.R., 2007. Assistive technology and robotic control using motor cortex ensemble-based neural interface systems in humans with tetraplegia. J. Physiol. 579 (3), 603–611.

Everitt, B.S., 1984. An Introduction to Latent Variable Models. Chapman and Hill, London.

Faisal, A.A., Selen, L.P.J., Wolpert, D.M., 2008. Noise in the nervous system. Nat. Rev. Neurosci. 9, 292–303.

Fetz, E.E., 1969. Operant conditioning of cortical unit activity. Science 163, 955–957.

Fetz, E.E., Baker, M.A., 1972. Operantly conditioned patterns of precentral unit activity and correlated responses in adjacent cells and contralateral muscles. J. Neurophysiol. 36, 179–204.

Fetz, E.E., Finocchio, D.V., 1971. Operant conditioning of specific patterns of neural and muscular activity. Science 174, 431–435.

Ganguly, K., Carmena, J.M., 2009. Emergence of a stable cortical map for neuroprosthetic control. PLoS Biol. 7, e1000153.

Georgopoulos, A.P., Schwartz, A.B., Kettner, R.E., 1986. Neuronal population coding of movement direction. Science 233, 1416–1419.

Ghahramani, Z., Hinton, G., 1997. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, Department of Computer Science, University of Toronto.

Gilja, V., Kalmar, R.S., Santhanam, G., Ryu, S.I., Yu, B.M., Afshar, A., et al., 2005. Trial-by-trial mean normalization improves plan period reach target decoding. Soc. Neurosci. Abstr. (519.18).

Godschalk, M., Lemon, R.N., Kuypers, H.G., van der Steen, J., 1985. The involvement of monkey premotor cortex neurones in preparation of visually cued arm movements. Behav. Brain Res. 18, 143–157.

Gomez, J.E., Fu, Q., Flament, D., Ebner, T.J., 2000. Representation of accuracy in the dorsal premotor cortex. Eur. J. Neurosci. 12 (10), 3748–3760.

Harrison, R.R., Kier, R.J., Chestek, C.A., Gilja, V., Nuyujukian, P., Ryu, S.I., et al., 2009. Wireless neural recording with single low-power integrated circuit. IEEE Trans. Neural. Syst. Rehabil. Eng. 17 (4), 322–329.

Hastie, T., Tibshirani, R., Friedman, J.H., 2003. The Elements of Statistical Leaning, second ed. Springer, New York.

Hatsopoulos, N., Joshi, J., O'Leary, J.G., 2004. Decoding continuous and discrete motor behaviors using motor and premotor cortical ensembles. J. Neurophysiol. 92, 1165–1174.

Hinterberger, T., Schmidt, S., Neumann, N., Mellinger, J., Blankertz, B., Curio, G., et al., 2004. Brain-computer communication and slow cortical potentials. IEEE Trans. Biomed. Eng. 51(6), 1011–1018.

Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., et al., 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 442, 164–171.

Hocherman, S., Wise, S.P., 1991. Effects of hand movement path on motor cortical activity in awake, behaving rhesus monkeys. Exp. Brain Res. 83, 285–302.

Humphrey, D.R., Schmidt, E.M., Thompson, W.D., 1970. Predicting measures of motor performance from multiple cortical spike trains. Science 170, 758–762.

Isaacs, R.E., Weber, D.J., Schwartz, A.B., 2000. Work toward real-time control of a cortical neural prosthesis. IEEE Trans. Rehabil. Eng. 8, 196–198.

Jarosiewicz, B., Chase, S.M., Fraser, G.W., Velliste, M., Kass, R. E., Schwartz, A.B., 2008. Functional network reorganization during learning in a brain-computer interface paradigm. Proc. Natl. Acad. Sci. USA 105 (49), 19486–19491.

Kalmar, R.S., Gilja, V., Santhanam, G., Ryu, S.I., Yu, B.M., Afshar, A., et al., 2005. PMd delay activity during rapid sequential-movement plans. Soc. Neurosci. Abstr. (519.17).

Kemere, C., Santhanam, G., Yu, B.M., Afshar, A., Ryu, S.I., Meng, T.H., et al., 2008. Detecting neural state transitions using hidden Markov models for motor cortical prostheses. J. Neurophysiol. 100, 2441–2452.

Kemere, C., Shenoy, K.V., Meng, T.H., 2004. Model-based neural decoding of reaching movements: a maximum likelihood approach. IEEE Trans. Biomed. Eng. 51 (6), 925–932.

Kennedy, P., Andreasen, D., Ehirim, P., King, B., Kirby, T., Mao, H., et al., 2004. Using human extra-cortical local field potentials to control a switch. J. Neural. Eng. 1 (2), 72–77.

Kennedy, P.R., Bakay, R.A.E., 1998. Restoration of neural output from a paralyzed patient by a direct brain connection. NeuroReport 9, 1707–1711.

Kennedy, P.R., Bakay, R.A.E., Moore, M.M., Adams, K., Goldwaithe, J., 2000a. Direct control of a computer from the human central nervous system. IEEE Trans. Rehabil. Eng. 8, 198–202.

AQ:2

Kennedy, P.R., Bakay, R.A.E., Moore, M.M., Adams, K., Goldwaithe, J., 2000b. Direct control of a computer from the human central nervous system. IEEE Trans. Rehabil. Eng. 8, 198–202.

Kim, S.P., Simeral, J.D., Hochberg, L.R., Donoghue, J.P., Black, M.J., 2008. Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia. J. Neural. Eng. 5, 455–476.

Kulkarni, J.E., Paninski, L., 2008. State-space decoding of goal-directed movements. IEEE Signal. Process. Mag. 25 (1), 78–86.

Leuthardt, E.C., Schalk, G., Wolpaw, J.R., Ojemann, J.G., Moran, D.W., 2004. A brain-computer interface using electrocorticographic signals in humans. J. Neural. Eng. 1, 63–71.

Lewicki, M.S., 1998. A review of methods for spike sorting: the detection and classification of neural action potentials. Network: Comput. Neural Syst. 9 (4), R53–R78.

MacKay, D.J.C., 2003. Information Theory, Inference, and Learning Algorithms. Cambridge University Press, Cambridge, UK.

Maynard, E.M., Hatsopoulos, N.G., Ojakangas, C.L., Acuna, B.D., Sanes, J.N., Normann, R.A., et al., 1999. Neuronal interactions improve cortical population coding of movement direction. J. Neurosci. 19, 8083–8093.

McFarland, D.J., Sarnacki, W.A., Wolpaw, J.R., 2003. Brain-computer interface (BCI) operation: optimizing information transfer rates. Biol. Psychol. 63 (3), 237–251.

Messier, J., Kalaska, J.F., 2000. Covariation of primate dorsal premotor cell activity with direction and amplitude during a memorized-delay reaching task. J. Neurophysiol. 84, 152–165.

Moran, D.W., Schwartz, A.B., 1999a. Motor cortical activity during drawing movements: population representation during spiral tracing. J. Neurophysiol. 82, 2693–2704.

Moran, D.W., Schwartz, A.B., 1999b. Motor cortical representation of speed and direction during reaching. J. Neurophysiol. 82, 2676–2692.

Moritz, C.T., Perlmutter, S.I., Fetz, E.E., 2008. Direct control of paralysed muscles by cortical neurons. Nature 456, 639–642.

Mulliken, G.H., Musallam, S., Andersen, R.A., 2008. Decoding trajectories from posterior parietal cortex ensembles. J. Neurosci. 28 (48), 12913–12926.

Musallam, S., Corneil, B.D., Greger, B., Scherberger, H., Andersen, R.A., 2004. Cognitive control signals for neural prosthetics. Science 305, 258–262.

Ng, A.Y., Jordan, M.I., 2002. On Discriminative Vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (Eds.), Advances in Neural Information Processing Systems 14. MIT Press, Cambridge, MA.

Polikov, V.S., Tresco, P.A., Reichert, W.M., 2005. Response of brain tissue to chronically implanted neural electrodes. J. Neurosci. Methods 148, 1–18.

Riehle, A., Requin, J., 1994. Are extent and force independent movement parameters? preparation- and movement-related neuronal activity in the monkey cortex. Exp. Brain Res. 99, 56–74.

Roweis, S., Ghahramani, Z., 1999. A unifying review of linear Gaussian models. Neural Comput. 11 (2), 305–345.

Ryu, S.I., Shenoy, K.V., 2009. Human cortical prostheses: lost in translation? Neurosurg. Focus 27 (1), E5.

Salinas, E., Abbott, L.F., 1994. Vector reconstruction from firing rates. J. Comput. Neurosci. 1, 89–107.

Santhanam, G., Ryu, S.I., Yu, B.M., Afshar, A., Shenoy, K.V., 2006. A high-performance brain-computer interface. Nature 442, 195–198.

Santhanam, G., Sahani, M., Ryu, S.I., Shenoy, K.V., 2004. An extensible infrastructure for fully automated spike sorting during online experiments. In: Proceedings of the 26th Annual Conference of the IEEE EMBS, 4380–4384.

Santhanam, G., Yu, B.M., Gilja, V., Ryu, S.I., Afshar, A., Sahani, M., et al., 2009. Factor-analysis methods for higher-performance neural prostheses. J. Neurophysiol. 102, 1315–1330.

Schalk, G., Miller, K.J., Anderson, N.R., Wilson, J.A., Smyth, M.D., Ojemann, J.G., et al., 2008. Two-dimensional movement control using electrocorticographic signals in humans. J. Neural Eng. 5 (1), 75–84.

Schwartz, A.B., 1994. Direct cortical representation of drawing. Science 265, 540–542.

Schwartz, A.B., Cui, X.T., Weber, D.J., Moran, D.W., 2006. Brain-controlled interfaces: movement restoration with neural prosthetics. Neuron 52 (1), 205–220.

Scott, S.H., 2004. Optimal feedback control and the neural basis of volitional motor control. Nat. Rev. Neurosci. 5, 532–546.

Scott, S.H., 2008. Inconvenient truths about neural processing in primary motor cortex. J. Physiol. 586, 1217–1224.

Serby, H., Yom-Tov, E., Inbar, G.F., 2005. An improved p300-based brain-computer interface. IEEE Trans. Neural Syst. Rehabil. Eng. 13 (1), 89–98.

Serruya, M.D., Hatsopoulos, N.G., Paninski, L., Fellows, M.R., Donoghue, J.P., 2002. Instant neural control of a movement signal. Nature 416, 141–142.

Shen, L., Alexander, G.E., 1997a. Neural correlates of a spatial sensory-to-motor transformation in primary motor cortex. J. Neurophysiol. 77, 1171–1194.

Shen, L., Alexander, G.E., 1997b. Preferential representation of instructed target location versus limb trajectory in dorsal premotor area. J. Neurophysiol. 77, 1195–1212.

Shenoy, K.V., Meeker, D., Cao, S., Kureshi, S.A., Pesaran, B., Mitra, P., et al., 2003. Neural prosthetic control signals from plan activity. NeuroReport 14, 591–596.

Shoham, S., Paninski, L.M., Fellows, M.R., Hatsopoulos, N.G., Donoghue, J.P., Normann, R.A., 2005. Statistical encoding model for a primary motor cortical brain-machine interface. IEEE Trans. Biomed. Eng. 52 (7), 1313–1322.

Sodagar, A.M., Wise, K.D., Najafi, K., 2007. A fully-integrated mixed-signal neural processor for implantable multi-channel cortical recording. IEEE Trans. Biomed. Eng. 54, 1075–1088.

Srinivasan, L., Eden, U.T., Mitter, S.K., Brown, E.N., 2007. General-purpose filter design for neural prosthetic devices. J. Neurophysiol. 98, 2456–2475.

Srinivasan, L., Eden, U.T., Willsky, A.S., Brown, E.N., 2006. A state-space analysis for reconstruction of goal-directed movements using neural signals. Neural Comput. 18 (10), 2465–2494.

Taylor, D.M., Tillery, S.I.H., Schwartz, A.B., 2002. Direct cortical control of 3D neuroprosthetic devices. Science 296, 1829–1832.

Taylor, D.M., Tillery, S.I.H., Schwartz, A.B., 2003. Information conveyed through brain-control: cursor vs. robot. IEEE Trans. Neural Syst. Rehabil. Eng. 11, 195–199.

Thacker, N.A., Bromiley, P.A., 2001. The Effects of a Square Root Transform on a Poisson Distributed Quantity. Technical Report 2001-010, Imaging Science and Biomedical Engineering Divison, University of Manchester.

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer Verlag, New York.

Velliste, M., Perel, S., Spalding, M.C., Whitford, A.S., Schwartz, A.B., 2008. Cortical control of a prosthetic arm for self-feeding. Nature 453 (7198), 1098–1101.

Warland, D.K., Reinagel, P., Meister, M., 1997. Decoding visual information from a population of retinal ganglion cells. J. Neurophysiol. 78, 2336–2350.

Wessberg, J., Stambaugh, C.R., Kralik, J.D., Beck, P.D., Laubach, M., Chapin, J.K., et al., 2000. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature 408, 361–365.

Wise, S.P., Boussaoud, D., Johnson, P.B., Caminiti, R., 1997. Premotor and parietal cortex: corticocortical connectivity and combinatorial computations. Annu. Rev. Neurosci. 20, 25–42.

Wolpaw, J., McFarland, D., 2004a. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. Proc. Natl. Acad. Sci. USA 101 (51), 17849–17854.

Wolpaw, J.R., McFarland, D.J., 2004b. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. Proc. Natl. Acad. Sci. USA 101 (51), 17849–17854.

Wu, W., Black, M.J., Mumford, D., Gao, Y., Bienenstock, E., Donoghue, J.P., 2004a. Modeling and decoding motor cortical activity using a switching Kalman filter. IEEE Trans. Biomed. Eng. 51 (6), 933–942.

Wu, W., Gao, Y., Bienenstock, E., Donoghue, J.P., Black, M.J., 2006. Bayesian population decoding of motor cortical activity using a Kalman filter. Neural Comput. 18 (1), 80–118.

Wu, W., Hatsopoulos, N.G., 2008. Real-time decoding of nonstationary neural activity in motor cortex. IEEE Trans. Neural Syst. Rehabil. Eng. 16 (3), 213–222.

Wu, W., Kulkarni, J.E., Hatsopoulos, N.G., Paninski, L., 2009. Neural decoding of hand motion using a linear state-space model with hidden states. IEEE Trans. Biomed. Eng. 17 (4), 370–378.

Wu, W., Shaikhouni, A., Dongohue, J.P., Black, M.J., 2004b. Closed-loop neural control of cursor motion using a Kalman filter. In: Proceedings of the 26th Annual Conference of the IEEE EMBS, 4126–4129.

Yu, B.M., Afshar, A., Santhanam, G., Ryu, S.I., Shenoy, K.V., Sahani, M., 2006a. Extracting dynamical structure embedded in neural activity. In: Weiss, Y., Schölkopf, B., Platt, J. (Eds.), Advances in Neural Information Processing Systems 18 MIT Press, Cambridge, MA, pp. 1545–1552.

Yu, B.M., Cunningham, J.P., Santhanam, G., Ryu, S.I., Shenoy, K.V., Sahani, M., 2009. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. J. Neurophysiol. 102, 614–635.

Yu, B.M., Cunningham, J.P., Shenoy, K.V., Sahani, M., 2008. Neural Decoding of Movements: From Linear to Nonlinear Trajectory Models. Volume 4984 of Lecture Notes in Computer Science. Springer, Berlin/Heidelberg, pp. 586–595.

Yu, B.M., Kemere, C., Santhanam, G., Afshar, A., Ryu, S.I., Meng, T.H., et al., 2007. Mixture of trajectory models for neural decoding of goal-directed movements. J. Neurophysiol. 97, 3763–3780.

Yu, B.M., Ryu, S.I., Santhanam, G., Churchland, M.M., Shenoy, K.V., 2004. Improving neural prosthetic system performance by combining plan and peri-movement activity. In: Proceedings of the 26th Annual Conference of the IEEE EMBS, 4516–4519.

Yu, B.M., Shenoy, K.V., Sahani, M., 2006b. Expectation propagation for inference in non-linear dynamical models with Poisson observations. In: Proceedings IEEE Nonlinear Statistical Signal Processing Workshop.

Zhang, K., Ginzburg, I., McNaughton, B.L., Sejnowski, T.J., 1998. Interpreting neuronal population activity by reconstruction: unified framework with application to hippocampal place cells. J. Neurophysiol. 79, 1017–1044.