5 Intracortical Brain–Machine Interfaces

Emily R. Oby, Jay A. Hennig, Aaron P. Batista, Byron M. Yu, and Steven M. Chase

Abstract

A brain–machine interface, or BMI, directly connects the brain to the external world, bypassing damaged biological pathways. It replaces the impaired parts of the nervous system with hardware and software that translate a user's internal motor commands into action. In this chapter, we will discuss the four basic components of an intracortical BMI: an intracortical neural recording, a decoding algorithm, an output device, and sensory feedback. In Sect. 5.2 we will discuss intracortical signals, the electrodes used to record them, and where in the brain to record them. The salient features of the neural signal useful for control are extracted with a decoding algorithm. This algorithm translates the neural signal into an intended action which is executed by an output device, such as a robotic limb, the person's own muscles, or a computer interface. In Sect. 5.3 we will discuss classification decoders and

Electronic Supplementary Material: The online version of this chapter (https://doi.org/10.1007/978-3-030-43395-6_5) contains supplementary material, which is available to authorized users.

E. R. Oby
Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, Pittsburgh, PA, USA

Department of Neurobiology, University of Pittsburgh, Pittsburgh, PA, USA

University of Pittsburgh Brain Institute and Systems Neuroscience Center, Pittsburgh, PA, USA

J. A. Hennig
Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, Pittsburgh, PA, USA

Neuroscience Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, USA

A. P. Batista
Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA, USA

Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, Pittsburgh, PA, USA

University of Pittsburgh Brain Institute and Systems Neuroscience Center, Pittsburgh, PA, USA

B. M. Yu
Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, Pittsburgh, PA, USA

Neuroscience Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA

Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA

S. M. Chase (✉)
Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, Pittsburgh, PA, USA

Neuroscience Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: schase@cmu.edu

B. He (ed.), *Neural Engineering*, https://doi.org/10.1007/978-3-030-43395-6_5

how they can be implemented in a BMI for communication. In Sect. 5.4 we will discuss continuous decoders for moment-by-moment control of a computer cursor or robotic arm. In Sect. 5.5, we will discuss a BMI that controls electrical stimulation to directly activate a patient's own paralyzed muscles and reanimate their arm. Finally, in Sect. 5.6, we will discuss ongoing work toward expanding sensory feedback with the goal of making intracortical BMIs a clinically viable option for treating paralysis, as well as other research trends.

Keywords

Neural population activity · Motor cortex · Motor control · Neural decoding · Probabilistic models · Classifier · Kalman filter · Functional electrical stimulation

5.1 What Is a Brain–Machine Interface?

Humans are capable of a nearly endless repertoire of movements: we can walk, run, skip, reach, grab, kick, throw, dance, and more. The ease with which most of us perform these movements conceals the fact that motor control is one of the most complex tasks the brain performs. More brain resources are devoted to the problem of controlling our movements than are devoted to any other task we might perform. The primary motor cortex, as its name indicates, is the area of the brain chiefly responsible for sending axons to the spinal cord to exert control over the muscles. In addition to primary motor cortex, there are at least six other cortical areas that also send axons down the spinal cord to help control muscles: dorsal premotor cortex, ventral premotor cortex, supplementary motor area, and three separate regions of the cingulate motor area. In addition to cortex, several subcortical regions are engaged during motor control, including major parts of the thalamus, basal ganglia, and the spinal cord. The cerebellum, which is composed of more neurons than the rest of the brain combined, is involved in coordinating movements. Motor control only seems easy because we don't tend to think about it very much: we just do it. In fact, the only time we really think about motor control is when it is impaired.

Movements can become impaired for a number of reasons, including neurological injury or disorders at the level of the brain, spinal cord, peripheral nerves, and muscles. The most common causes of paralysis are stroke and spinal cord injury. There are approximately 291,000 Americans currently living with spinal cord injuries, with more than 17,500 new cases each year (National Spinal Cord Injury Statistical Center). About 40% of these individuals are paraplegic, i.e., their legs are paralyzed, and 60% are quadriplegic, i.e., their arms and legs are paralyzed. Fewer than 1% of patients fully recover from spinal cord injuries. Spinal cord injuries disrupt the natural pathway between the brain and the muscles but leave the cortical neurons involved in generating the movement signals intact. If we could leverage these intact control signals, and decode motor intent, we could create assistive technologies that bypass the damaged pathway to restore motor control to those who have lost it. This is the clinical goal of brain–machine interfaces.

A brain–machine interface, or BMI, directly connects the brain to the external world, bypassing damaged biological pathways. It replaces the impaired parts of the nervous system with hardware and software that translate a user's internal motor commands into action. BMI technologies serve as a neural engineering solution to replace or restore motor or sensory function to patients with neurological injury or disease.

An intracortical BMI (iBMI) is driven by the action potentials recorded from individual neurons. Action potentials, or "spikes," are the electrical signals by which neurons transmit information. Intracortical BMIs involve implanting electrodes directly into the cortex. This neural recording method provides greater spatial and temporal specificity than noninvasive recording techniques because the electrodes are only microns from the neurons. The greater specificity increases the decoding accuracy and allows for higher degree-of-freedom control.

5.1.1 History of Intracortical BMIs

In 1969, Eberhard Fetz created what is considered to be the first modern-day intracortical BMI. In this first example of an iBMI, he showed that monkeys could learn to volitionally modulate the activity of a single neuron in primary motor cortex in order to receive a food reward. To do so, Fetz recorded the spikes from a neuron while providing the monkey with visual or auditory feedback about the number of spikes that neuron generated per unit time (i.e., firing rate). When the monkey increased the firing rate above a certain threshold, he was rewarded. Following this operant conditioning, monkeys would quickly and consistently increase the firing rate of the recorded neuron to earn rewards. Monkeys were also asked to separately control two neurons, increasing the rate of one and decreasing the rate of the other. The independent control of two neurons demonstrated that the control was not simply achieved by a general increase of all neurons' firing rates. This early study provided the first proof of concept that a person might someday be able to modulate neural activity to control a computer cursor or robotic arm.

The Fetz study was near the beginning of several decades of intensive work to define the nature of the signals encoded in motor cortex. In 1970, Humphrey, Schmidt, and Thompson conducted a set of experiments that addressed the possibility of using neural signals to make quantitative predictions of simple motor behaviors. Using recordings from a small set of neurons during a wrist flexion and extension task, they predicted arm position, velocity, and net force exerted about the joint. They showed that force was quite accurately predicted and that arm position and velocity could also be predicted, though typically not as well. In the 1980s, Apostolos Georgopoulos showed that neural activity during whole-arm reaches predicted the direction of the reach quite well. Together these findings suggested that motor cortical neural activity was correlated with *extrinsic* motor control variables (e.g., the direction of the arm in space) as well as *intrinsic* motor control variables (e.g., the force exerted by the arm). These results have been the basis of BMI development since.

Over the following decades, technology developed that enabled researchers to record from populations of tens to hundreds of neurons. In 1999, neural signals recorded simultaneously from rat motor cortex were used to control a robotic arm [1]. Soon after, the hand trajectories of primates were predicted from the activity of a population of neurons [2], and monkeys were using neural activity to control computer cursors [3, 4] and robotic arms for reaching and grasping [5]. This was the beginning of a now-flourishing field of iBMI development and research.

In 2006, a group of researchers at BrainGate performed the first clinical trial to establish an iBMI in a human [42]. They recorded neural population activity from a paralyzed person as he imagined limb movements and used that activity to drive the movement of a cursor on a computer screen. Then, in 2012, the same group demonstrated that a person who had been paralyzed by brainstem stroke could directly control a robotic arm [43]. Specifically, she was able to control the velocity of the robot's hand to make reaches, and she simultaneously controlled a decoder that could execute one of four hand actions to grasp objects. With this level of control, she was able to use the robotic arm to grasp a bottle and bring it to her mouth for a drink. Compared to natural reaching and grasping movements, the brain-controlled robot's reaching and grasping were slower and less accurate. However, this result showed that it is possible to use tens of neurons to control a robotic device and interact with objects. Shortly afterward, a team at the University of Pittsburgh demonstrated that a person could control ten degrees of freedom of a robotic arm [6, 7]. The ten degrees of freedom consisted of three dimensions of translation, three dimensions of orientation of the robot's hand, and four dimensions of hand shape. By including the hand shape dimensions, the researchers could increase the repertoire of possible movements to include dexterous manipulation of objects. Ultimately, the person could

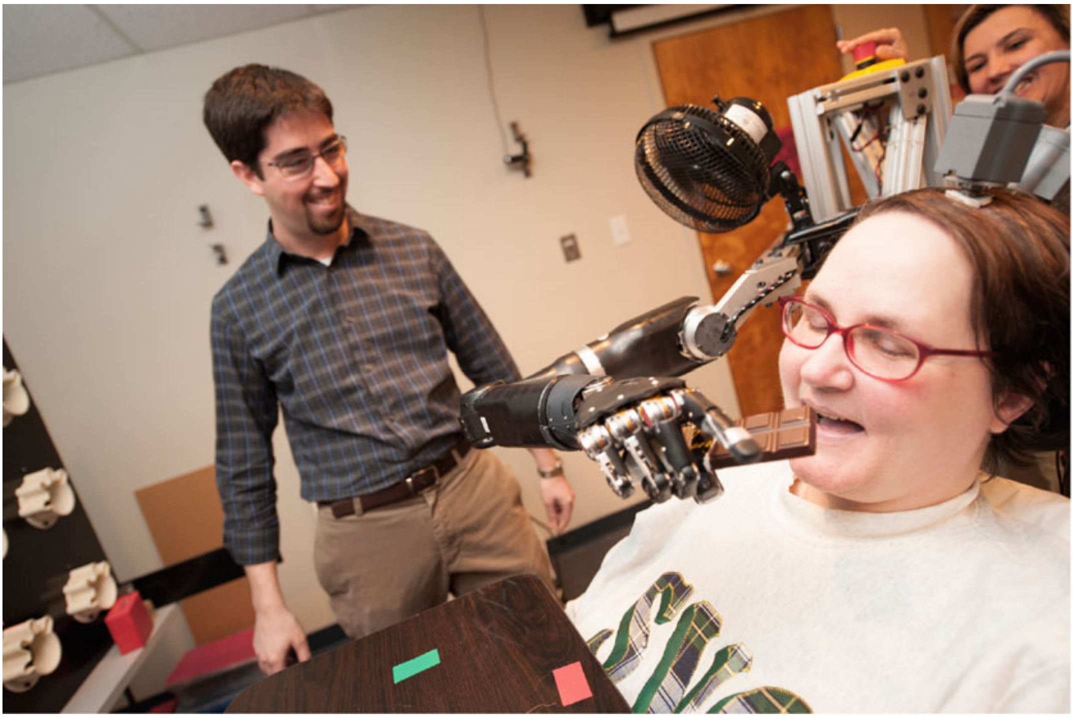


Fig. 5.1 An iBMI user at the University of Pittsburgh controls a robotic arm to eat a chocolate bar as members of the research team look on. (Photo credit: UPMC)

perform skillful and coordinated reach and grasp movements like those that are essential for daily activities, such as shaking hands or feeding oneself (Fig. 5.1).

The ultimate goal of iBMI systems for people with paralysis is to restore the function of their own arms, hands, and legs. Currently, the best prospect for this is to use neural commands to activate the muscles with electrical stimulation. Two groups have recently shown progress in iBMIs to control functional electrical stimulation (FES) of a user's own muscles. The electrical stimulation activates the muscles, causing them to contract and thus to generate movement. In 2016, a group of researchers at Battelle was the first to demonstrate successful control of muscle activation using intracortically recorded signals in a human [8]. Neural activity was decoded to control the stimulation of muscles in the forearm via electrodes in a custom-built sleeve. With the iBMI-controlled FES, the person was able to independently control his fingers as well as six wrist and hand movements, allowing him to perform some activities of daily living. In 2017, a group of researchers at Case Western Reserve University demonstrated that an individual with a high cervical spinal cord injury could use his own cortical activity to control a chronically implanted FES system to perform coordinated reaching and grasping movements with his own paralyzed arm and hand [9]. He could volitionally perform reaches to drink coffee and feed himself (Fig. 5.2). These studies are a major step toward a clinically viable BMI for reaching and grasping after paralysis.

We have been focusing on iBMIs for movement, but the principles of decoding motor intent can also be applied to solve the problem communication, allowing users to "type" by moving a computer cursor to different letters on a screen. There is a group of patients for whom restoring communication is crucial. These patients are referred to as "locked in" because, although they are awake and aware, they have lost control of

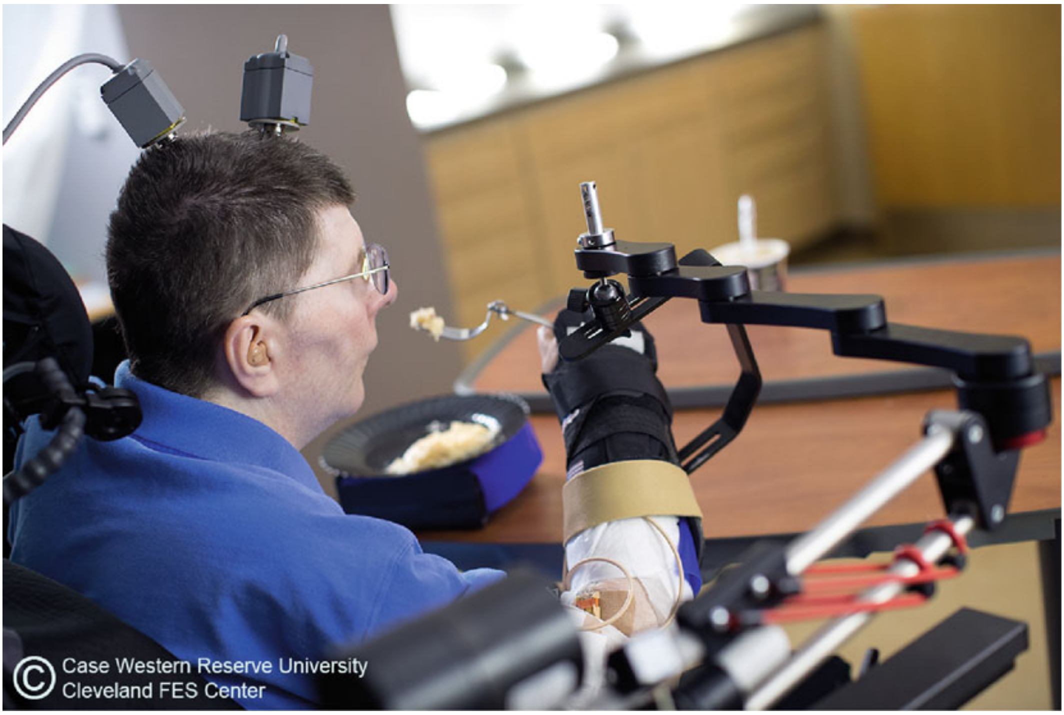


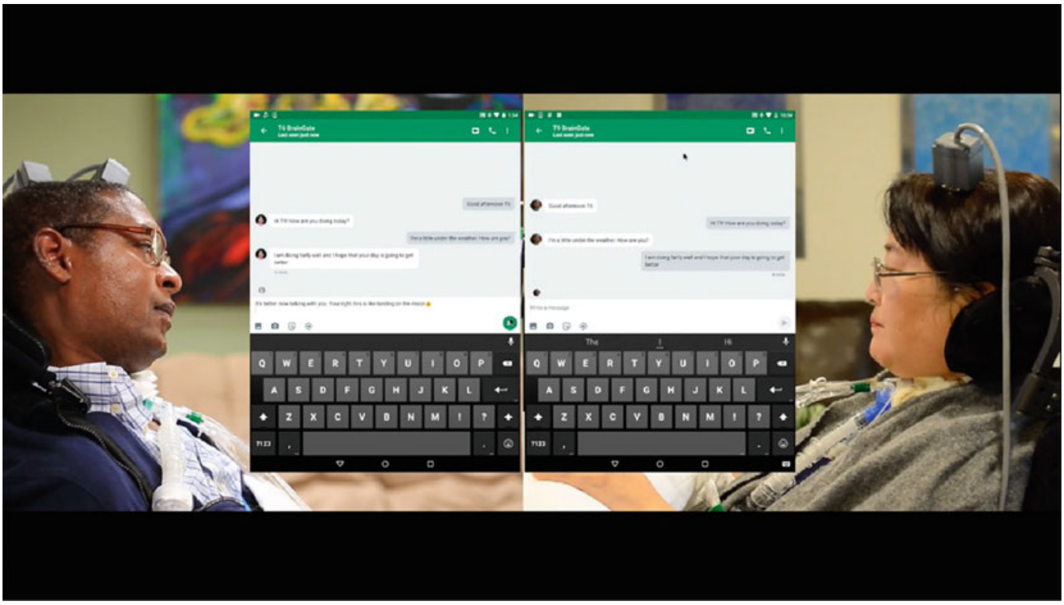
Fig. 5.2 An iBMI user controls functional electrical stimulation of muscles in his arm and hand to feed himself as part of the BrainGate2 clinical trials. (Photo credit: Russell Lee; Case Western Reserve University/Cleveland FES Center)

all voluntary muscles, except (in some cases) those that control vertical eye movements and blinking, due to brainstem stroke or amyotrophic lateral sclerosis (ALS). These patients have no way of speaking or producing facial expressions, so restoring some form of communication would dramatically enhance their quality of life. For people with paraplegia or quadriplegia who can still speak, a communication BMI could provide an important means to interact with others via email or texting. A number of studies have established the feasibility of iBMIs for communication [10, 11]. In 2017, a group at Stanford University developed a high-performance iBMI for communication that allows users to control a computer tablet to perform activities like browsing the web and texting (Fig. 5.3; [12]). Users were able to perform typing tasks that simulated real-world applications such as typing messages at a conversational pace, with a typing rate of 24 characters per minute.

In this chapter, we will discuss motor iBMIs for both movement and communication. We will describe the components of an iBMI, the state-of-the-art control, and future directions of research and development.

5.1.2 Components of an Intracortical BMI

A BMI consists of four basic components: a neural recording, a decoding algorithm, an output device, and sensory feedback (Fig. 5.4). Intracortical BMIs begin by recording neural signals from electrodes implanted in the cortex. Next, the salient features of the neural signal useful for control are extracted with a decoding algorithm. This algorithm translates the neural signal into an intended action which is executed by an output device, such as a robotic limb, the person's own muscles, or a computer interface. Finally, the user receives sensory feedback about the action, allowing them to make corrections if they move off course and also allowing them to improve over time with learning.



Good afternoon T6

Hi T9! How are you doing today?

I'm a little under the weather. How are you?

I am doing fairly well and I hope that your day is going to get better

Fig. 5.3 Two iBMI users who are part of BrainGate text each other. (Credit: BrainGate Collaboration)

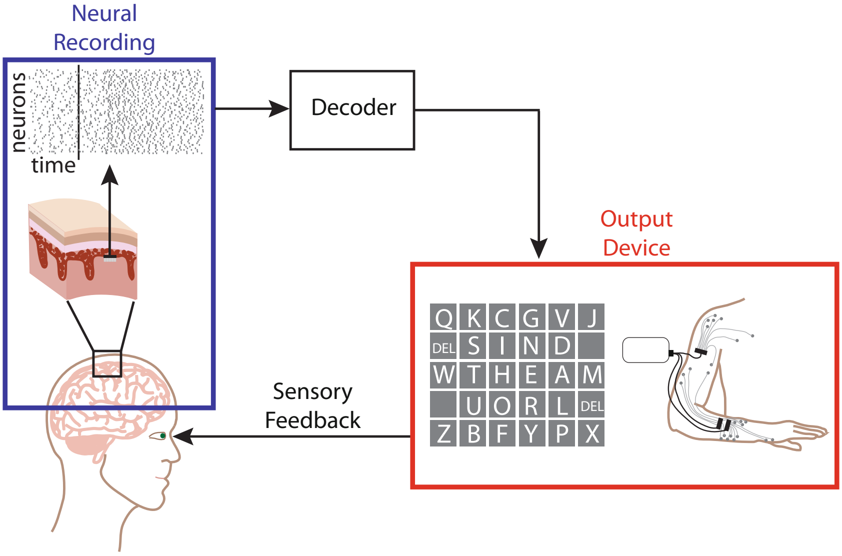


Fig. 5.4 The components of an iBMI consist of intracortical neural recordings, a decoder to translate the neural input into a control signal, an output device, and sensory feedback

Building an effective iBMI depends on choosing a brain area for the neural recordings, a decoding algorithm, an output device, and feedback for the desired use. These choices are interrelated. The most appropriate command signal to control the output device will depend on the goal of the task and the particular device being controlled. In turn, these choices influence which motor cortical area is most appropriate to record from and what type of signals to record. We will discuss the considerations related to the neural recordings in Sect. 5.2.

Once the neural signals have been recorded, a decoding algorithm (or "decoder") translates the user's movement intentions into a control signal suitable for guiding the output device. There are two classes of BMI decoders: discrete and continuous. A discrete decoder estimates one of several possible movement goals by solving a classification problem. The most common use for this is a communication device, where patients use their iBMI to type letters, much as one would if one were composing a text or an email. Communication BMIs focus on the speed and accuracy with which a desired key on a keyboard can be selected. We will discuss how these devices work in Sect. 5.3.

A continuous decoder estimates the moment-by-moment details of a movement trajectory. This is needed for guiding a computer cursor or robotic limb along a desired path. For example, a person may wish to guide a robotic limb to pick up a glass of milk without knocking over the milk carton. We will talk about continuous decoders in Sect. 5.4.

The decoder produces a control signal that is then fed into an output device. There are a variety of output devices for BMIs depending on the particular the needs of the user. One common device is a computer cursor, where a person controls the cursor by thinking about making a movement, much as they would control a computer mouse. Other common output devices for BMI users include robotic arms and motorized wheelchairs. Another type of output device is perhaps the most natural one: the person's own limb. In Sect. 5.5, we will talk about recent efforts using electrical stimulation to directly activate a patient's muscles to reanimate their own arm.

The final element of the BMI control loop is sensory feedback. The most common sensory feedback is visual: a user can look at the device they are controlling and see how it is responding, which allows them to make corrective movements and learn to better control the device. Feedback has been shown to dramatically improve BMI performance. Some tasks, however, require more than just visual feedback to be performed dexterously. Consider putting on a necklace. To fasten the necklace behind our head, we must rely on touch to manipulate the necklace clasp. Such tasks have motivated the inclusion of nonvisual feedback into BMI systems. Emerging bidirectional technology aims to "close the loop" by inputting sensory signals conveying naturalistic proprioceptive or somatosensory information directly to the nervous system via electrical stimulation. We will discuss recent progress toward closing the loop with somatosensory feedback in Sect. 5.6.

5.2 Choosing the Input for iBMIs

The most appropriate control signal for an iBMI will depend on the goal of the task and the device being controlled. In turn, these aspects of the iBMI influence the choice of brain area from which to record. In this section, we will discuss the intracortical input signals, the electrodes that can be used to record these signals, and the motor neurophysiology that underlies this choice.

5.2.1 Neural Signal Recordings

Intracortical recording techniques provide access to signals that consist of neural activity, which can come from individual neurons or groups of neurons near the electrode. There are three types of signals that can be recorded with intracortical electrodes: single-unit activity, multiunit activity, and local field potentials (LFP). Single-unit activity consists of action potentials which emanate from a single neuron. Multiunit activity consists of action potentials from a small group of neurons near the electrode tip that are not clearly discriminable from one another. The LFP signal is thought to reflect the summation of local neural activity, mostly changes in membrane potentials, and is comprised of the activity of perhaps hundreds to thousands of neurons. Single-unit activity has the most specific information about the fine details of intended movements, with each neuron responding uniquely to different aspects of movement. Multiunit activity and LFPs arise from averaging over many neurons. Thus, the resulting activity consists of a signal that is common to the contributing neurons. While multiunit and LFP signals are correlated with movement, the information is not as specific as that obtained from individual neurons.

Of the three signals, single-unit activity provides the most specific information about the fine details of intended movements and has been shown to lead to good iBMI performance. Single-unit activity is identified through a process known as "spike sorting" (Fig. 5.5). The action potentials recorded with a single electrode can come from potentially multiple neurons, and in spike sorting, we attempt to classify which action potential came from which neuron using the neurons' characteristic waveform shapes. Waveform shapes are determined by the particular combination of ion channels expressed by a neuron and the proximity of that neuron to the recording site and so provide a "fingerprint" that can be used to uniquely identify action potentials specific to that neuron. To identify which action potentials belong to a given neuron, the recorded voltage trace is typically first band-pass filtered (e.g., 600 Hz–6 kHz). After that, the waveform snippets are aligned to the time at which the voltage crosses a predetermined threshold. The snippets are then sorted (i.e., clustered) based on the specific shapes of the waveforms.

Multiunit activity also leads to good iBMI performance and does not require the intensive processing involved in identifying single-unit activity. Instead, a voltage threshold is set, and all waveform snippets that exceed that threshold (i.e., "threshold crossings," Fig. 5.5) are counted with no further assignment to particular neurons. The type and quality of information that can be extracted from multiunit activity depend on the threshold setting, because the threshold impacts the effective sampling radius of the electrode. A selective threshold (i.e., a threshold farther from zero) results in threshold crossings that are likely due to spikes from individual neurons within a small sampling radius, close to the electrode, akin to single-unit activity. A permissive threshold (i.e., a threshold closer to zero) results in more threshold crossings, because it enlarges the effective sampling radius of the electrode. The larger effective sampling radius captures threshold crossings from smaller neurons and neurons farther from the electrode than those captured with a selective threshold.

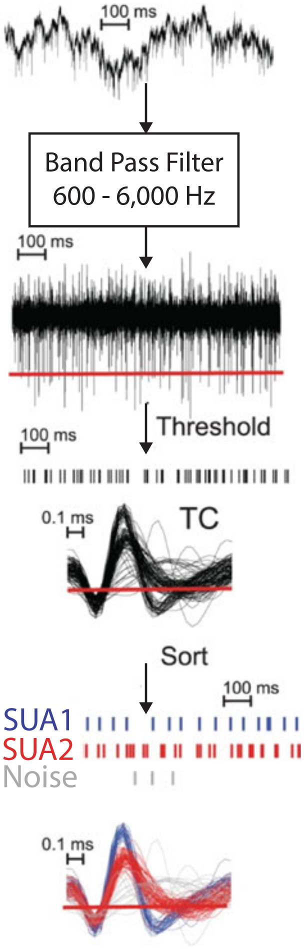


Fig. 5.5 Neural signal processing cascade for threshold crossings (TC) and single-unit activity (SUA). The raw voltage trace recorded from a single electrode is band-pass filtered. Then a voltage-based threshold is used to identify TCs. TCs can be further sorted into activity attributed to a single neuron or noise based on waveform shape. (Adapted from Perel et al. [13])

LFP signals can also be used for iBMIs, but they do not offer as much specific information as single- and multiunit activity about the movement. However, LFP signals are not as susceptible to degradation over time, so there is a tradeoff between resolution and duration in choosing a neural signal for BMI control. Often, LFPs can provide an alternative signal if single- and multiunit activity is no longer available. LFPs can be obtained from different band-pass filtering of the same electrode signal, typically between 0.3 and 300 Hz. Ultimately, the viability of iBMIs in a clinical setting will depend on the longevity of the implanted electrodes and their ability to reliably record signals that are informative about movement. Using LFP as a secondary input signal could extend the lifetime of the iBMI.

5.2.2 Multielectrode Arrays

The goal of controlling an output device with multiple degrees of freedom drove the development of multielectrode recording arrays. Recording arrays come in many shapes and sizes and enable recordings from a large number of neurons at the same time. Multielectrode arrays are the primary recording technology used for iBMIs. Each electrode within the array records from a small population of neurons close to the electrode tip. There are three main types of multielectrode arrays: microwires, flexible polymer-based microelectrode arrays, and silicon-based arrays. Microwires are typically made of stainless steel or platinum–iridium. They can be customized to include the desired number of electrodes in the desired configuration and of the desired length. Flexible arrays are made of polymers that are not as stiff as microwires and, as such, are a closer mechanical match to the soft brain tissue into which they are implanted. This design can lead to less damage to the tissue, a lower inflammatory response, and consequently better quality signals. While microwires and flexible arrays have many attractive features, they can be fragile. One of the most popular electrode arrays for iBMIs is a silicon-based array, the Utah Array (Fig. 5.6). It is the only array currently approved for clinical trials with human patients by the US Food and Drug Administration (FDA). It is a silicon-machined device which permits the simultaneous implantation of 100 platinum–iridium electrodes in a small region of cortex (16 mm^2). Each electrode array consists of a silicon base with a 10 × 10 grid of electrode shanks etched into it. Each electrode has an impedance of roughly 80–150 kΩ, and is separated from its neighbors by 400 μm.

5.2.3 Motor Neurophysiology

Decisions about where to implant multielectrode arrays and how to design decoding algorithms to extract movement information are guided by our understanding of how movement is controlled naturally. Let's consider the multiple processes involved in picking up a cup for a sip of coffee. The sight of the coffee cup might inspire a desire for a sip of coffee, and the desire for coffee is translated into a plan to reach for the cup. The hand is then shaped to grasp the cup, and the arm extends to bring the hand toward the cup. The cup is then grasped with an appropriate level of force, and the cup is brought to the mouth. Throughout this process, visual, tactile, and proprioceptive feedback are used to adjust the movement to ensure our actions are successful. For an iBMI to work as seamlessly as natural movement, we will likely want to make use of the natural control signals. For decades, neuroscientists have worked to understand how the motor cortex produces arm movements in healthy individuals. Understanding how movement occurs naturally can inform the design of technologies like iBMI to improve the quality of life for individuals with injury or disease.

Primary motor cortex (M1) has long been thought to be an ideal location for recording BMI control signals because it is involved in generating voluntary movements (Fig. 5.7). However, other brain areas also have signals related to aspects of movement. For example, premotor cortex (PMd) has movement planning signals, and posterior parietal cortex (PPC) has been shown to be involved in the transformation from visual representations of reach goals to the movement itself. The brain's "sensory areas" also often reflect internal representations of stimuli and movements. Each of these brain areas is comprised of neurons that modulate their activity in association with different aspects of movement. Thus, there are neural signals from many brain areas that could be used as the input

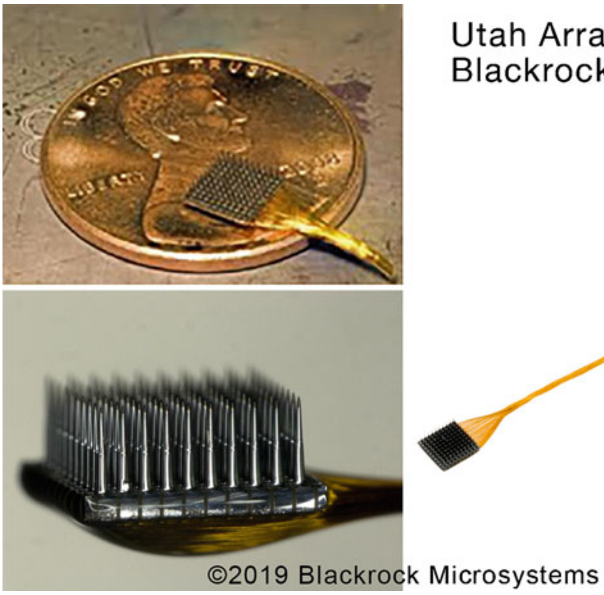


Fig. 5.6 The Utah array is a 100-electrode microelectrode array that is commonly used to record neural signals for iBMIs. (Credit: Utah Array- ©2019 Blackrock Microsystems)

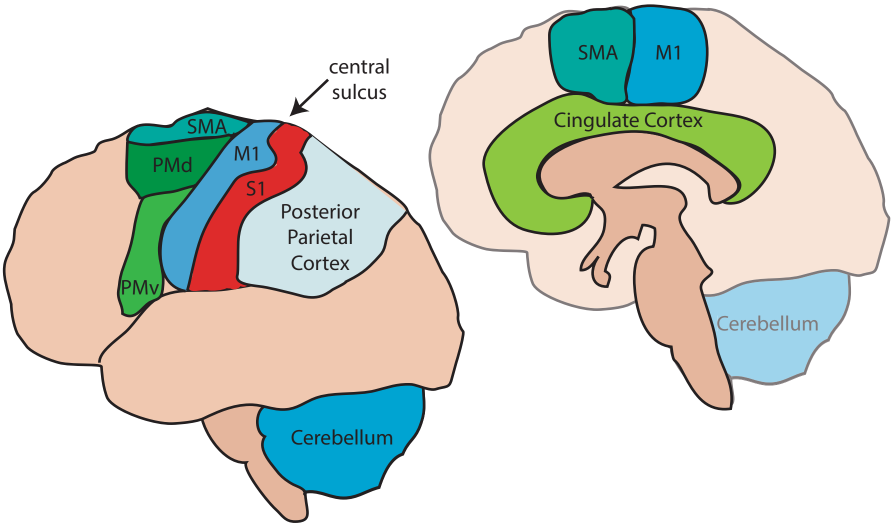


Fig. 5.7 Lateral (left) and medial (right) views of a human brain with some of the brain areas involved in motor control highlighted

to an iBMI depending on the type of information to be extracted and the decoding algorithm.

The desired use of the BMI helps determine the choice of brain area from which to record the input signal. Recordings from primary motor cortex (M1), dorsal and ventral premotor cortex (PMd and PMv), supplementary motor area (SMA), posterior parietal cortex (PPC), and primary somatosensory cortex (S1) can all contribute information to real-time predictions of hand position, velocity, grip force, and muscle activity [5]. However, the different cortical areas vary in the quality of their predictions of different aspects of movement. Thus, the source of the input signal should be a brain area which is informative about the desired aspects of movement.

One way to design a BMI that controls the movement of a device in a natural way would be to mimic how the brain controls arm movements. To do this requires an understanding of how the brain produces arm movements. However, for the purposes of a BMI, we can do quite well by just considering how neural signals are correlated with different aspects of movement. There is a debate in the field of motor control research about whether M1 generates arm movements by directly controlling muscle activity or by signaling aspects of movement like position, velocity, and acceleration that are then transformed into muscle activity by downstream neural networks such as the spinal cord. There is evidence for both representations, which can be summarized by the results of two seminal studies. The first was an experiment performed by Ed Evarts in awake behaving monkeys [14]. He trained monkeys to make wrist flexion and extension movements against opposing or assistive loads while recording single-neuron activity in M1 and muscle activity (EMG). The experimental design dissociated the movement itself from the force required to produce it. For example, a given flexion displacement under an opposing load required greater activity of the wrist flexor muscles than under an assistive load. The experimental results showed that force was reflected in the firing rate of the M1 neurons he recorded. Thus, activity in the motor cortex reflects kinetic aspects of movement, i.e., force or muscle activity.

The second relevant study was an experiment by Georgopoulos and his colleagues in which they found that most (75%) of the neurons they recorded had a firing rate which varied with the direction of hand movement [15]. Neural signals in M1 were recorded while a monkey made reaches from the center of a workspace out to eight peripheral targets. The relationship between firing rate and direction looked like a sinusoid and could be described by the equation

$$y = b + m\ \cos\left(\theta - \theta_{\vec{p}}\right) \tag{5.1}$$

where y is the firing rate of the neuron, b is its baseline firing rate (i.e., the mean firing rate), m is the modulation depth (i.e., the difference in firing rate between the baseline firing rate and the maximum firing rate), θ is the direction of movement, and $\theta_{\vec{p}}$ is the direction of movement that elicited the highest firing rate (i.e., the neuron's preferred direction). This study showed that, in addition to reflecting forces and muscle activity, M1 activity also reflects the direction of arm movement.

Since these experiments, a number of groups have reported a correlation between M1 firing rates and various kinematic variables, including direction and distance of targets, as well as direction, speed, and spatial path of hand displacement. Other groups have found M1 firing rates to be related to forces and even to muscle activity. It appears that M1 includes a heterogeneous representation of both the kinematics and kinetics of limb movements. The good news, from the perspective of designing an iBMI, is that either representation can be exploited as a BMI control signal, depending on the intended function of the device. If we can accurately extract information about the position, velocity, or acceleration of the desired movement from the neural activity, that type of control signal can be used to move the robotic arm. If we can accurately extract information about muscle activity from the neural activity, that control signal can be used to drive muscle stimulators.

The control signal for a BMI output device could conceivably be any of the aspects of movement with which neural signals are correlated. Most current BMIs utilize *kinematic* signals to control external actuators such as computer cursors or robotic limbs. We can drive robotic limbs because kinematics (i.e., position and velocity) can be directly decoded from neural activity and a kinematic signal could drive the endpoint position of the limb (see Sects. 5.3 and 5.4). BMIs could also take advantage of the *kinetic* (i.e., force-related) signals in M1. A demonstration of a kinetic BMI is cortically controlled stimulation of paralyzed muscles. We can reanimate the arm because muscle activity can be directly decoded from neural activity and a kinetic signal could control electrical stimulation of paralyzed muscle tissue (see Sect. 5.5).

The decision about which brain area one should record from should take into account which aspects of movement are best suited for the intended type of control. A kinetic BMI, like one to control stimulation of muscles, would benefit from a muscle-like input signal. Such a BMI is likely to be implanted in M1, which has shown strong correlations to muscle activity. On the other hand, a user who will be engaging primarily in computer cursor control might benefit more from an implant in an area of the brain that strongly encodes kinematic signals, such as the location of movement goals. Activity in premotor cortex (PMd and PMv; Fig. 5.7) reflects target positions [16, 17] and could function as the signal source for a BMI to be used for a communication interface involving target selection similar to typing [18]. For endpoint control of a robotic limb, it would be advantageous to decode a kinematic signal such as hand position or velocity. Although this has been most notably done with signals from M1 [19], areas PMd, SMA, and S1 also contain information about hand position and velocity [5].

5.3 Intracortical Spelling Devices

The goal of a communication BMI is to provide a means of communication for the user. This might hold particular value for locked-in patients, who are no longer able to speak. Ideally, we would record neural signals from the parts of the brain responsible for speech, decode the intended message, and use that to drive a speech synthesizer or a speech transcription program. However, the neural encoding of speech is only now beginning to be understood. Instead, current devices leverage our understanding of the neural representation of intended movements to design spelling devices through control of an onscreen keyboard. Users imagine reaching toward the letter they would like to type. By decoding the intended *movement* from motor and premotor cortex, it is possible to infer which letter the person is trying to type. Rather than solving the neural encoding of speech, we only need to solve a classification problem: of all the letters on the screen, which character is the person trying to select?

In this section, we will describe classification decoders, including an example of how to implement a classifier. We will discuss how to estimate the model parameters of the classifier and how to use the resulting classifier in a BMI context.

5.3.1 Classification Decoders

The goal of classification is to take an input and assign it to one of K discrete classes (Fig. 5.8). For an iBMI, the input is the spike counts across a population of neurons (in Fig. 5.8, two neurons are illustrated). We ask which of K discrete movements most likely corresponds to the user's intended movement. To begin, we need labeled training data. For example, we could record a user's neural activity while he or she imagines reaching to various letters displayed like keys on a keyboard, as instructed by the experimenter. As shown in Fig. 5.8a, the training data consists of the class label (i.e., the imagined letter, depicted as different colors) and the value of each data point (i.e., the activity of two neurons, y_1 and y_2). In the training phase, we fit a probability model to the training data, a process we will describe in detail below. This training phase defines a set of decision boundaries between the classes (Fig. 5.8b). Once the classifier has been trained, we can then use the classifier to predict the label of a new data point (Fig. 5.8c). We do this by comparing where the data point falls relative to the classifier's decision boundaries. In the example shown in Fig. 5.8c, the new data point would be assigned to class 3.

Before we describe how a classifier works in detail, let's start with a simplified example to build intuition. In this example, the iBMI user is typing one of three different letters, either "E," "Q," or "A," while we record activity from a single neuron (y). First, we ask the user to repeatedly imagine reaching to the letter "E" while we record the neuron's spiking activity. Because neurons are noisy, the neuron's spike counts will

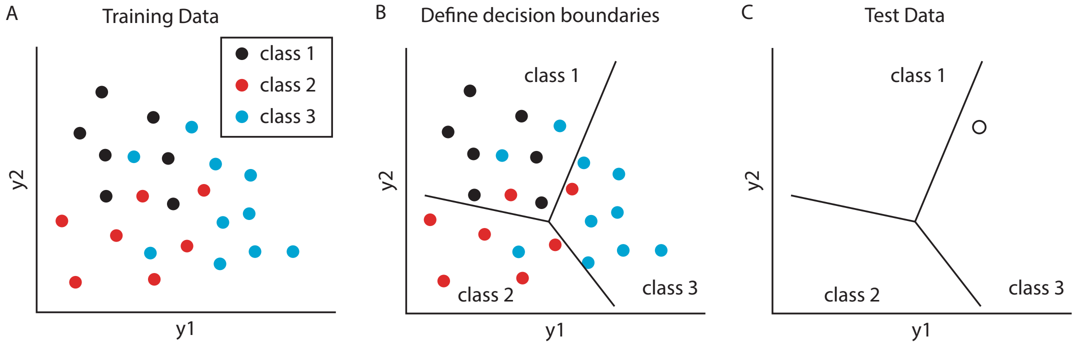


Fig. 5.8 (**a**) To train a classifier, we collect a set of labeled training data, consisting of a set of data points where each data point consists of values, y_1 and y_2, along with their corresponding class labels, depicted here as the color of each data point. (**b**) The classifier uses the training data to create a set of decision boundaries (black lines) that divide the different classes of data points into different regions. (**c**) Given a new data point (open circle) for which we do not know the true class label, the classifier will predict the class of that new data point using its decision boundaries. In this case, the new data point would be assigned to class 3

not be exactly the same every time (Fig. 5.9a). Instead, we obtain a distribution that reflects the conditional probability of measuring a particular spike count given that the person is intending to reach to the letter "E" (Fig. 5.9b), here idealized as a Gaussian distribution. We can repeat this process for the letter "Q" and again for the third letter, "A."

How can we use this spiking activity to build a classifier that will classify the letter a user intends from only the neural activity? First, let's suppose the neuron spiked 30 times. We would probably guess that the user was intending to reach to the "A" because that is the letter that is the most probable for that spike count. Similarly, if the neuron spiked five times, by comparing the probability distributions, we would guess that the user was intending to reach to the "E." In general, we would like our classifier to predict the most likely letter given the recorded spike counts by comparing the measurement to the conditional distributions of the neural activity. If we recorded 20 spikes, what letter would we guess the user intended? This time it is not obvious because intending to reach to either the "Q" or the "A" would be equally likely to generate that measurement. Now suppose we repeated this exercise for a second neuron. Neuron 2 will also have spike counts for which the classification will be unambiguous and spike counts for which the classification is ambiguous. In general, the range of ambiguous spike counts of the two neurons will not overlap because neurons have different preferred stimuli. Thus, adding even just one more neuron will likely enable our classifier to make a more accurate guess about the intended letter. Similar to the single-neuron example, we can estimate the distribution of spike counts of two (or more) neurons conditioned on the user's intentions to select each of the three letters (Fig. 5.9c). Now, the distribution of spike counts corresponding to each letter is a region in a plane (for two neurons) or in an N-dimensional space for N neurons. This looks very much like the scenario depicted in Fig. 5.8 for which classifiers are designed.

Now that we have established some intuition, let's talk about how to implement the classifier using a probability model. Here we'll suppose that the user is typing one of K different letters, which we'll refer to as c_1, c_2, ..., c_K. (In the above example, we had $K = 3$, with c_1 = "E," c_2 = "Q," and c_3 = "A"). While the user is intending to type these different letters, we record the spike counts, $y \in R^d$, from d different neurons. To build the classifier, we would like to be able to predict which letter, c_k, the user was most likely intending just from observing the neural activity, y. In other words, we want to know $P(c_k \mid y)$. What our training data provides, however, is the reverse: the distribution of spike counts given the

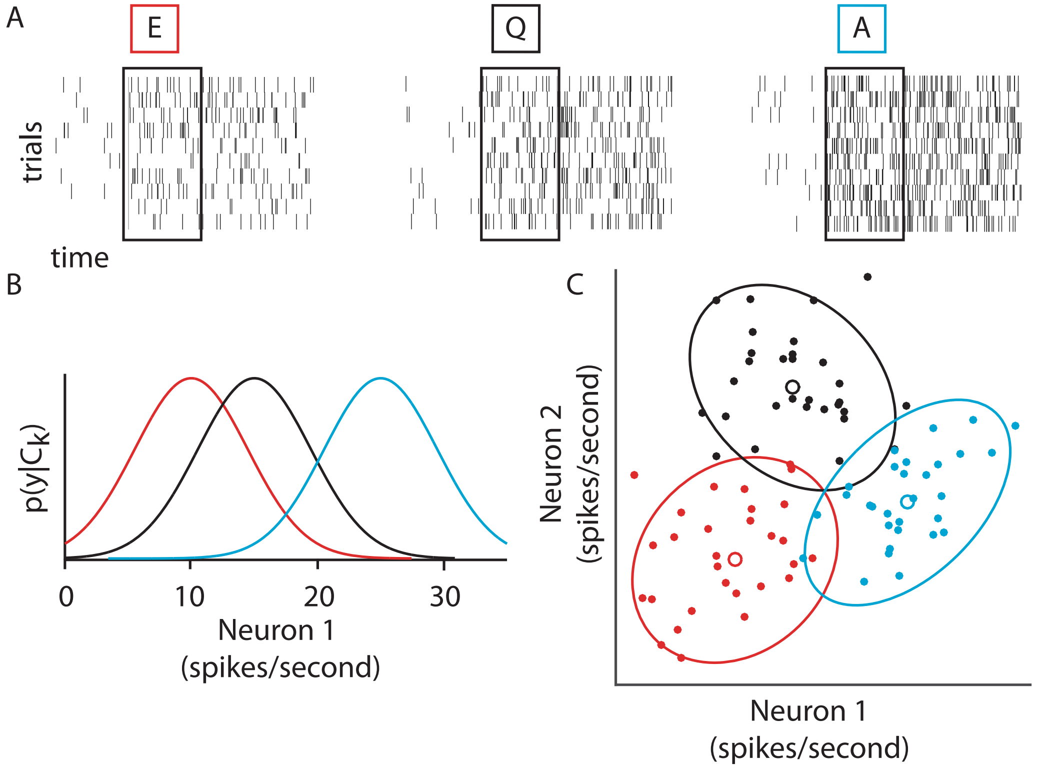


Fig. 5.9 (**a**) A user imagines typing the letter "E" many times while neural activity is recorded from one neuron. We can plot the neural activity across time as a raster plot, in which each row is a trial and each mark represents the time of a spike occurred. We count the spikes occurring within a given window (black box) and repeat this process for other letters (e.g., "Q" and "A"). (**b**) We can plot the conditional distribution of spike counts we recorded given that the user imagined typing the letter "E" (red), "Q" (black), or "A" (blue). For each letter, we will likely see a different distribution of spike counts, here idealized as Gaussians. (**c**) If we record from a second neuron's spike counts simultaneously, we can plot the joint distribution of spike counts in both neurons for each letter. For each set of spike counts, we can fit a multivariate Gaussian to the data by estimating the mean (unfilled circle) and covariance (ellipse). Adding additional neurons will likely improve our ability to discriminate which letter the user was imagining

intended letter, $P(y \mid c_k)$. Additionally, because we know the true letters the user was intending in the training data, we also know $P(c_k)$, the proportion of data points with a given class label. We can relate all of these terms using Bayes' rule:

$$P\left(c_k|y\right) = \frac{P\left(y|c_k\right)P\left(c_k\right)}{P\left(y\right)} \tag{5.2}$$

To predict the letter the subject was intending given the neural activity y, we will simply choose the class (k) that has the largest value of $P(c_k \mid y)$. This is similar to how in our single-neuron example, we chose the letter that had the highest probability in Fig. 5.9b. To write this mathematically, given y, the classifier will predict the class as follows:

$$\hat{k} = \underset{k}{\operatorname{argmax}} P\left(c_k|y\right) = \underset{k}{\operatorname{argmax}} \frac{P\left(y|c_k\right) P\left(c_k\right)}{P\left(y\right)} \tag{5.3}$$

We can ignore the denominator in Eq. 5.3 because $P(y)$ is the same for every class k, so it does not affect which k yields the maximum.

Equation 5.3 tells us how we can predict the letter the user was most likely intending, given only the spiking activity, y. The right-hand side of the equation includes two terms: the conditional probability of spiking given the intended letter,

$P(y|\, c_k)$, and the prior probability of each letter, $P(c_k)$. We now discuss how we can use our training data to estimate these two quantities.

Let's begin by discussing the first term. What we would like is to use the training data to describe the distribution of neural activity observed for each letter. In Fig. 5.9b, c, we note that a Gaussian captures the first and second moments (i.e., mean and covariance) of the distribution of neural activity. This means when the user is intending a particular letter (e.g., c_k = "E"), the distribution of observed spike counts (y) can be described by a Gaussian with a mean and covariance:

$$P\left(y|c_k\right) = N\left(\mu_k, \Sigma_k\right) \tag{5.4}$$

where $y \in R^d$ is a vector of spike counts from a population of d neurons, μ_k describes their mean spike counts, and the covariance matrix Σ_k describes any correlations that might exist among neurons. To estimate these mean and covariance parameters, we seek to find the parameters μ_k and Σ_k that maximize the probability of having observed the activity that we observed. This widely used procedure is known as maximum likelihood estimation (MLE). Let's suppose we have N examples (or trials) of recorded neural activity, $\{y_1, \ldots, y_N\}$ when the user was imagining typing the same letter c_k. Then the probability of recording a particular y_i (i.e., on a single trial) is

$$P\left(y_i\ |c_k\right) = (2\pi)^{-d/2}|\Sigma_k|^{-1/2}e^{-(y_i-\mu_\mathrm{k})^\top \Sigma_\mathrm{k}{}^{-1}(y_i-\mu_\mathrm{k})/2} \tag{5.5}$$

Assuming the neural activity recorded across trials is conditionally independent, the probability of observing $\{y_1, \ldots, y_N\}$ is the product of observing each individual trial:

$$P\left(y_1, \ldots, y_N\ |c_k\right) = \prod\nolimits_{i=1}^{N}(2\pi)^{-d/2}|\Sigma_k|^{-1/2}\cdot e^{-(y_i-\mu_\mathrm{k})^\top \Sigma_\mathrm{k}{}^{-1}(y_i-\mu_\mathrm{k})/2} \tag{5.6}$$

This joint probability indicates the "likelihood" of observing the spike counts given that the true parameters were μ_k and Σ_k. For this, we write

$$L\left(\mu_k, \Sigma_k; y_1, \ldots, y_{N,}c_k\right) = P\left(y_1, \ldots, y_N\ |c_k\right) \tag{5.7}$$

The approach of maximum likelihood estimation is to choose the parameters most consistent with the observed data. In other words, we will choose the parameters that maximize the probability of the neural activity that we observed:

$$\mu_k, \Sigma_k = \underset{\mu,\Sigma}{\operatorname{argmax}}\ L\left(\mu, \Sigma; y_1, \ldots, y_N, c_k\right) \tag{5.8}$$

By maximizing the log likelihood function for each class k, we can find μ_k, Σ_k for each of the distributions $P(y|\, c_k)$. We would find (with a few lines of math, omitted here) that the parameters μ_k, Σ_k are the sample mean and sample covariance of the spike counts recorded with class k:

$$\mu_k = \frac{1}{N}\sum\nolimits_{i=1}^{N} y_i \tag{5.9}$$

$$\Sigma_k = \frac{1}{N}\sum\nolimits_{i=1}^{N}\left(y_i - \mu_k\right)\left(y_i - \mu_k\right)^\top \tag{5.10}$$

If we had assumed that the covariance of neural activity was the same across classes (i.e., $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_k$), it can be shown that the resulting decision boundaries between classes are linear, as shown in Fig. 5.8c. As a variant of this Gaussian classifier, we could instead describe the distributions of neural activity $P(y|\, c_k)$ using a Poisson distribution and perform the same MLE procedure to estimate its parameters.

The other term we need to know in order to implement the classifier is $P(c_k)$, the prior probability of each class. This is the probability that a user is likely to want to type each letter without having observed any neural activity. For example, in the English language, "E" is a much more common letter than "Q," which means we should expect to observe "E" more often than "Q." Or, imagine that instead of typing letters, the goal was to select among different icons on a computer screen. It might be that each icon is expected to get the same amount of use. In this case, $P(c_k) = 1/K$ for each $k = 1, \ldots, K$. Using Bayes' rule, the classifier accounts for the prior probability when making its predictions (see Eqs. 5.2 and 5.3).

As an example of using a classification decoder in a BMI setting, researchers recorded neural activity from premotor cortex in monkeys to predict the intended reach target from neural activity while a monkey planned a reach [20]. The researchers assumed that neurons were Poisson and conditionally independent of one another. They then decoded the most likely target given the observed neural activity using a method similar to the approach described above. In this case, the monkey made arm reaches to each of eight targets that were equally likely by design. The classifier successfully decoded the correct target from the neural activity on 90% of the trials. This work was the first demonstration that neural activity recorded during the movement planning period could be decoded as a useful control signal for a classification iBMI.

Performance of classification decoders is assessed based on the speed and accuracy of target selection. In general, fast and accurate classification is difficult because neural activity is variable (cf. Fig 5.9a). The approach is to average the neural activity over a longer window of time. Because neural variability is Poisson-like, averaging over a longer window reduces the "noise" and results in a more accurate prediction of the target. However, with longer time windows, fewer predictions are made each second, resulting in slower decoding. There is a speed accuracy trade-off that makes the choice of the particular duration and placement of the time window an important design choice.

Rather than evaluating BMI performance on accuracy alone, one should include some measure of speed as well. One metric that is often used for this purpose is the information transfer rate (ITR; [18]). ITR measures how much information is conveyed per unit time. ITR increases with window duration but then decreases (Fig. 5.10). This is because ITR takes into account both how accurately and how quickly each target is selected. Accuracy fails to increase rapidly enough to overcome the slowdown in target selection rate with longer window durations. An intracortical classification decoder can convey 6.5 bits per second or 2–3 targets per second with greater than 90% accuracy. This would allow users to type at a speed of 15 words per minute. While this is an improvement over noninvasive BMIs, this is not yet up to the average typing speed of 40 words per minute. State-of-the-art communication BMIs, such as the one shown in Fig. 5.3, combine discrete and continuous decoders. We will discuss continuous decoders in the next section.

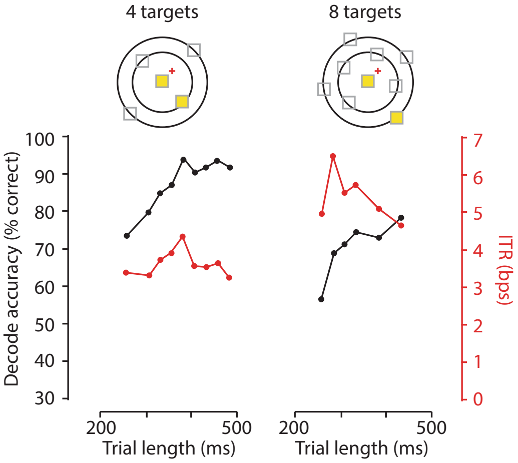


Fig. 5.10 The relationship between single-trial decoding accuracy and information transfer rate (ITR). Performance was measured during iBMI experiments for a four-target configuration and an eight-target configuration across varying trial lengths. Each data point represents performance calculated from one experiment (hundreds of trials). (Adapted from Santhanam et al. [18])

5.4 Intracortical Control of Continuous Effectors

In the spelling device, and in classification decoders in general, we decode the intended target (or letter) directly from the neural activity. However, if we want to control a robotic arm, we need to specify the path that the arm will take so that, for example, the user can prevent the arm from bumping into objects in the workspace. To specify the reach trajectory, we need to decode the evolution of the desired movement at progressive time steps. In other words, we need a continuous decoder. Accurate decoding of a continuous control signal is necessary for controlling not only robotic arms but also computer cursors or a patient's own paralyzed limb. In this section, we discuss three continuous decoders for BMI control: the population vector algorithm, the optimal linear estimator, and the Kalman filter. Variations on the Kalman filter are the current state of the art for continuous decoders.

5.4.1 Population Vector Algorithm

One of the first continuous decoders was the population vector algorithm (PVA). The PVA was proposed by Apostolos Georgopoulos in the 1980s as a way of decoding movement direction from a population of neurons. As mentioned in Sect. 5.2, the firing rates of neurons in motor cortex reflect the direction of a reach, as the relationship between a neuron's firing rate and the arm's reach direction is approximately cosine tuned (Fig. 5.11). This means that there is a reach direction for which the neuron fires maximally. We refer to this direction as the neuron's "preferred direction," $\theta_{\vec{p}}$. The neuron's firing rate decreases gradually as the reach direction moves away from this preferred direction. Different neurons have different preferred directions, so together the activity of a population of neurons can uniquely specify the arm's direction of movement. Specifically, when the arm is moving in a particular direction, θ, we can describe the firing rate, y, of one neuron as shown in Eq. 5.1. The firing rate is linearly related to $\cos\left(\theta - \theta_{\vec{p}}\right)$, and this relationship is the basis of PVA. In motor cortex, studies have used cosine tuning to describe movement direction, velocity, speed, position, force, and torque. Cosine tuning has also been used to describe neural activity in other nonmotor brain areas.

The fact that a neuron's firing rate has a systematic relationship with reach direction suggests that we can accurately decode a subject's intended reach direction from the activity of a single neuron. However, it is not easy to estimate direction of movement from one cosine-tuned neuron because there are multiple reach directions associated with a given firing rate. In Fig. 5.11, suppose that the neuron is firing at 30 spikes per second. This could correspond to the subject reaching at 0° (yellow) or 135° (cyan). As we saw in the classification example above, the activity of just one neuron can be ambiguous, but we can solve this problem by recording from a population of neurons in order to reduce uncertainty in our estimate of the reach direction.

As its name suggests, the population vector algorithm (PVA) utilizes the activity of a population of neurons to estimate the desired movement. Each neuron contributes a "push" in the direction of its preferred direction. This push is weighted by the neuron's normalized firing rate, given by $w_i = \frac{y_i - b_i}{m_i}$, where y_i is the measured firing rate of neuron i, b_i is the neuron's baseline firing rate, and m_i is its modulation depth. The algorithm then averages all of the neurons' contributions together to yield the resulting command. Mathematically, the PVA decoder is a weighted vector sum of each of the recorded neurons. Taken together, the prediction of the intended movement direction is the resulting population vector.

Figure 5.12 shows a simple example of the PVA using two neurons at one time point. Each of the neurons has a different preferred direction (Fig. 5.12a). The preferred direction of the neuron determines the direction of its push. The red neuron will push "up," while the blue neuron will push "left" (Fig. 5.12b). The measured firing rates determine the magnitude of the pushes (Fig. 5.12c). Let's consider the firing rates specified by the gray shaded box. The red neuron has a firing rate of $y_{\mathrm{red}} = 25$ spikes per second. Given that this neuron's tuning parameters are $b_{\mathrm{red}} = 35$ spikes per second and $m_{\mathrm{red}} = 15$ spikes per second, the magnitude of the red neuron's push is then $w_{\mathrm{red}} = -0.66$. This contribution is negative, so its push is now "down." Let's repeat this process for the blue neuron. The blue neuron has a firing rate $y_{\mathrm{blue}} = 45$ spikes per second. Given that this neuron's tuning parameters are $b_{\mathrm{blue}} = 35$ spikes per second and $m_{\mathrm{blue}} = 15$ spikes per second, the magnitude of the blue neuron's push is then $w_{\mathrm{blue}} = 0.66$. This contribution is positive, so its push stays "left." Taking the weighted sum of the two pushes, the population vector points down

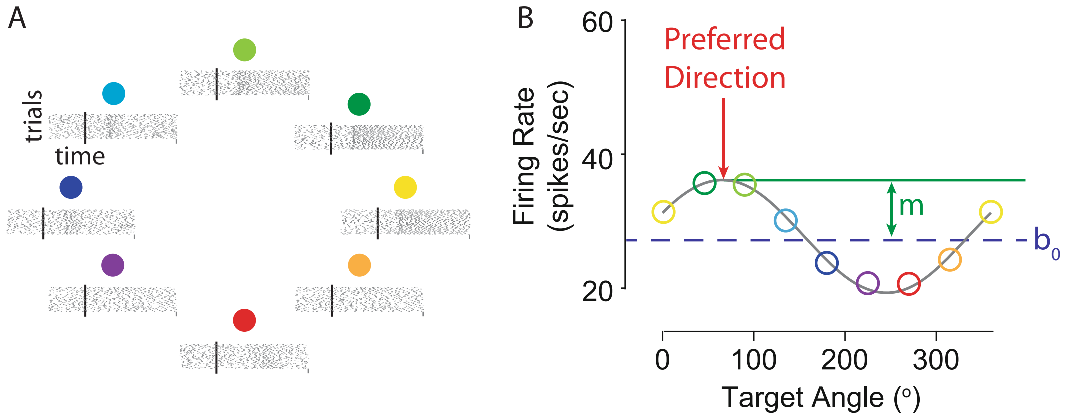


Fig. 5.11 (**a**) Spikes trains for repeated reaches to each of eight reach directions. (**b**) A cosine tuning curve describes the relationship between the neural firing rate and the reach directions. The preferred direction is the direction for which the neuron shows the maximal firing rate. The baseline firing rate, b_0, is the mean firing rate. The modulation depth, m, is the difference in firing rate between the baseline firing rate and the maximum firing rate

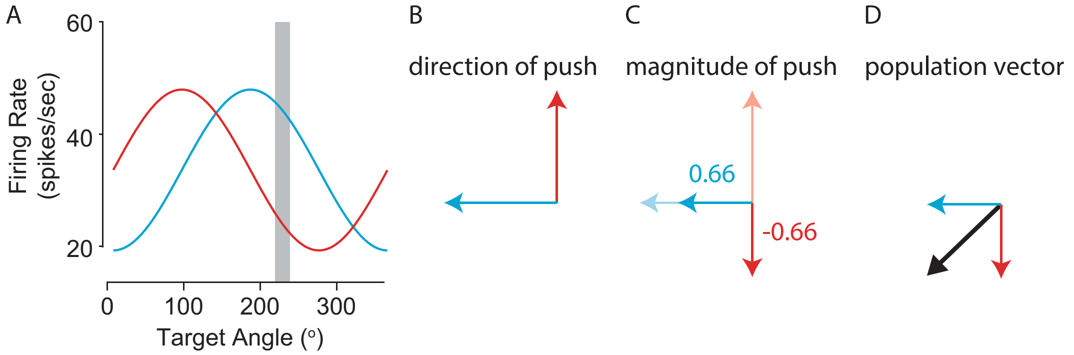


Fig. 5.12 A two-neuron example of the population vector algorithm. (**a**) Tuning curves of the two neurons (red and blue). (**b**) Each neuron contributes a push in the direction of its preferred direction. (**c**) The magnitude of each neuron's push is determined by its firing rate. (**d**) The population vector algorithm outputs a resultant vector represented by the black arrow that is the weighted sum of each neuron's push

and to the left, corresponding to a movement of around 225° (Fig. 5.12d). This procedure repeats at each time point, as the activity of the neurons varies over time.

In a BMI use scenario, the goal is to record neural activity from a population of neurons and convert it into the position of a cursor on a screen over time. In contrast to the classification decoders of the previous section, here we decode the cursor position at each point in time. This provides the user with continuous control over the trajectory that the cursor takes.

The PVA is a biologically inspired decoder, where the idea is that perhaps neurons in motor cortex cause slight contractions of muscles that push the arm in their preferred directions. However, PVA suffers from statistical biases: if there is a nonuniform distribution of preferred directions in the recorded neural population, the PVA will systematically misestimate the intended reach direction. In practice, it is rare to record a population of neurons with a uniform distribution of preferred directions. While this bias can be mitigated by recording from a large number of neurons or by sub-selecting neurons that have a uniform distribution of preferred directions, a better approach is to use an unbiased decoding algorithm. The optimal linear estimator and the

Kalman filter, both of which we will discuss shortly, are examples of unbiased decoders.

5.4.2 Optimal Linear Estimator

As discussed above, if we do not have a uniform distribution of preferred directions, we do not want to use a PVA decoder. Instead, we should specify a statistical model that describes the relationship between the intended movement and the activity of each neuron. This *encoding* model, a probabilistic description of how neural activity (y) varies based on the intended movement (x), is written as $P(y|x)$. The encoding model has parameters which are estimated during a decoder calibration phase. Applying Bayes' rule, we can then use this encoding model to create a *decoding* model, $P(x|y)$, which is our estimate of the intended movement given the observed neural activity.

An example of an unbiased continuous decoder is the optimal linear estimator (OLE). The OLE makes two assumptions: that firing rates are *linearly* related to intended movement and that neural variability is described by a Gaussian distribution. We can rewrite the cosine tuning model in matrix form:

$$y_t = b_0 + Bv_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, \Sigma), \tag{5.11}$$

where y_t is the $n \times 1$ vector of firing rates from n neurons, b_0 is the $n \times 1$ vector of baseline firing rates, B is the $n \times 2$ matrix of tuning coefficients, v_t is the 2×1 intended velocity, and ε_t is the $n \times 1$ noise vector.

We can now ask a question that is very similar to the classification problem we solved in Sect. 5.3: What is the most likely velocity given a measurement of firing rates from our population? From our encoding model (Eq. 5.11), we know the probability of the firing rates given the intended movement direction:

$$P(y_t \mid v_t) = (2\pi)^{-k/2}|\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(y_t - Bv_t - b_0)^\top \Sigma^{-1}(y_t - Bv_t - b_0)\right) \tag{5.12}$$

In the OLE, our estimate of velocity is the velocity that maximizes the above probability with respect to the observed neural activity. This velocity is

$$\hat{v}_t = \left(B^T \Sigma^{-1} B\right)^{-1} B^T \Sigma^{-1} (y_t - b_0) \tag{5.13}$$

Note that this estimate of velocity is a linear function of the recorded firing rates. Further, the OLE decoder corrects for any nonuniformity in the distribution of preferred directions, resulting in an unbiased estimate of intended velocity. This is why this decoder is called the optimal linear estimator.

Which decoder should we implement, a PVA decoder or an OLE decoder? The PVA decoder is simpler than statistical approaches like OLE, but the OLE decoder is optimal given the specified encoding model. Empirically, if we were to use each decoder to reconstruct arm trajectories, we would see that OLE performs significantly better than PVA and with fewer neurons [21]. However, both decoders do comparably well in a BMI use scenario because the users can incorporate feedback and correct errors quickly enough to compensate for any theoretical differences in system performance.

5.4.3 Kalman Filter

Is it possible to do even better? Both PVA and OLE estimate movement velocity given only the neural activity. But there is other information we could also incorporate into our estimates. For instance, we know that the cursor or arm should move smoothly. During arm reaches, the arm cannot teleport from one location to another instantaneously. Rather, there are finite constraints to the accelerations and decelerations that muscles can produce. We can use this information about the kinematics of the arm during natural reaching to influence how we allow our estimate of the desired trajectory to change with time. As a simple example, if we know where the arm is currently (current state), and how fast the arm is moving (state dynamics), we can predict where the arm will go next (future state). To use this information to improve our ability to decode arm velocity, we need to combine it with the information we have from the neural activity.

Consider trying to track a satellite. What sources of information might we use to do so? We could simply measure its position. But what if it goes behind a cloud or over a region of Earth with no sensors? We could potentially use Newton's laws to predict the satellite's trajectory. But what if something collides with our satellite and changes its course? Intuitively, the best way to track a satellite would be to combine our measurements and our predictions, weighting each source of information according to how reliable it tends to be.

These are the intuitions captured by the Kalman filter: it predicts the current state of a system based on our estimate at the previous state combined with new observations of data. The two key components of a Kalman filter are a state model that describes how the movement evolves over time and an observation model that describes how the observations relate to the movement. The way the state and observation models combine can be visualized graphically as in Fig. 5.13, with the red arrows indicating the state model and the black arrows indicating the observation model.

We outline below how these two models are combined to update our predictions at each time step t. The Kalman filter is based on linear-Gaussian relationships. First, we define the observation model:

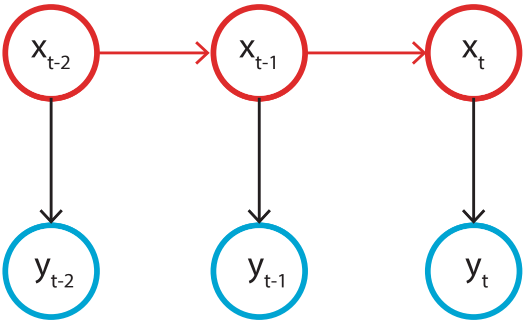


Fig. 5.13 Graphical model of a Kalman filter. Each vertical slice represents a time step. The nodes in red represent the state (e.g., movement velocity), while the nodes in blue represent the observations (e.g., spike counts). Each arrow represents a probabilistic relationship between the nodes

$$y_t = Bx_t + \varepsilon_t, \varepsilon_t \sim N(0, \Sigma) \tag{5.14}$$

where $y_t \in R^{n \times 1}$ is the vector of spike counts measured from all n neurons at time step t, $B \in R^{n \times d}$ is the matrix of tuning coefficients, $x_t \in R^{d \times 1}$ is the vector of the intended movement kinematics (e.g., cursor velocity) at time step t, d is the number of kinematic variables (e.g., $d = 2$ for a two-dimensional velocity), and $\varepsilon_t \in R^{n \times 1}$ is the vector of additive Gaussian noise, drawn from a distribution with mean 0 and covariance matrix $\Sigma \in R^{n \times n}$. Note that this is the same observation model that the OLE uses (Eq. 5.11), with two exceptions. First, we have assumed that the baseline, b_0 from Eq. 5.11, is already subtracted from the spike counts. This just simplifies the derivations below. Second, we denote the state as x instead of v, as it is common to incorporate other kinematic variables in addition to velocity, such as the position and acceleration, in a Kalman filter. Though for simplicity, we will assume that x contains only velocity in what follows.

The state model is defined as

$$x_t = Ax_{t-1} + \omega_t, \omega_t \sim N(0, Q) \tag{5.15}$$

where $A \in R^{d \times d}$ describes how the velocity evolves from one time step to the next, $\omega_t \in R^{d \times 1}$ is additive Gaussian noise to the velocity, and $Q \in R^{d \times d}$ is the covariance matrix of the velocity noise. Notice how in the observation model (Eq. 5.14), the current state is linearly related to the observed neural activity and in the state model (Eq. 5.15), the state at time t is linearly related to the state at time $t - 1$.

To calibrate the decoder, we estimate the parameters B, Σ, A, and Q in the observation and state models. Typically, in a decoder calibration session, the neural activity is recorded, while the states of the arm or cursor are known. This can be done in a number of ways. Some decoders are calibrated based on arm movements. Other decoders are calibrated by moving a cursor on the screen and having the user intend or imagine that they are moving the cursor. In this way, the state of the cursor is known or assumed during decoder calibration. Because Eqs. 5.14 and 5.15 are linear-

Gaussian, when both the states and neural activity are known, we can find the parameters using multivariate linear regression on the calibration data. We leave the derivation of the equations for the parameters from the observation and state models to homework problem #7.

How do we use these models to decode movement trajectories from neural activity for a BMI? What we would like to know is $P(x_t|\, y_1,\ \ldots, y_t)$, which describes the probability of the intended movement velocity at a particular time step given all of the recorded neural activity up to that time step. The decoded movement is the movement that maximizes this probability.

To decode the movement velocity at all time steps, we will compute $P(x_t|\, y_1,\ \ldots, y_t)$ sequentially starting from $t = 1$. In order to do this, we will first need to find $P(x_t|\, y_1,\ \ldots, y_{t-1})$. This is called a “one-step prediction” and can be found from the previous time step and the state model as follows:

$$\begin{aligned} &P\left(x_t|y_1,\ldots,y_{t-1}\right) \\ &= \int P\left(x_t|x_{t-1}\right) P\left(x_{t-1}|y_1,\ldots,y_{t-1}\right) dx_{t-1} \end{aligned} \tag{5.16}$$

This equation describes our current estimate of the movement velocity at time t given all of our observations up to time $t-1$, along with our knowledge of how the movement kinematics evolve over time. That is, it’s a prediction of where the state may have gone since our last measurement. We then augment this prediction with a “measurement update” that describes how this prediction changes when we observe y_t:

$$P\left(x_t|y_1,\ldots,y_t\right) = \frac{P\left(y_t|x_t\right) P\left(x_t|y_1,\ldots,y_{t-1}\right)}{P\left(y_t|y_1,\ldots,y_{t-1}\right)} \tag{5.17}$$

The one-step prediction and measurement update (Eqs. 5.16 and 5.17) are general. They can be used for any state and observation model as long as the graphical model is as shown in Fig. 5.13.

For the particular state and observation models defined in our example (Eqs. 5.14 and 5.15), we can simplify Eqs. 5.16 and 5.17. Because the relationships in our state and observation models (Eqs. 5.14 and 5.15) are linear-Gaussian, this means that all of the relevant marginal, conditional, and joint distributions are also Gaussian. Thus, all we need to do is compute the mean and covariance of each distribution.

We start with the state estimate at the previous time step $t-1$, $P(x_{t-1}|\, y_1, \ldots, y_{t-1})$. Let its mean and covariance be μ_{t-1} and Φ_{t-1}, respectively. The mean and covariance of the one-step prediction distribution are $\mu_t^- = E\left[x_t|y_1,\ldots,y_{t-1}\right]$ and $\Phi_t^- = \mathrm{Var}\left[x_t|y_1,\ldots,y_{t-1}\right]$. We can solve for these by plugging the state model into Eq. 5.16:

$$\begin{aligned} \mu_t^- =& A E\left[x_{t-1}|y_1,\ldots,y_{t-1}\right] \\ &+ E\left[\omega_t|y_1,\ldots,y_{t-1}\right] = A\mu_{t-1} \end{aligned} \tag{5.18}$$

Similarly, for the covariance:

$$\begin{aligned} \Phi_t^- &= \mathrm{Var}\left[Ax_{t-1} + \omega_t|y_1,\ldots,y_{t-1}\right] \\ &= A\Phi_{t-1}A^{\mathrm{T}} + Q \end{aligned} \tag{5.19}$$

In the measurement update, we use the new observation y_t to update the one-step prediction to compute the state estimate at the current time step t, $P(x_t|\, y_1,\ \ldots, y_t)$. Let its mean and covariance be μ_t and Φ_t, respectively. To compute μ_t and Φ_t, we first obtain the joint distribution of x_t and y_t given $y_1,\ \ldots, y_{t-1}$. Using the one-step prediction and the observation model, we find:

$$\begin{aligned} &\begin{bmatrix} y_t \mid y_1,\ldots,y_{t-1} \\ x_t \mid y_1,\ldots,y_{t-1} \end{bmatrix} \\ &\sim N\left(\begin{bmatrix} B\mu_t^- \\ \mu_t^- \end{bmatrix}, \begin{bmatrix} B\Phi_t^- B^{\mathrm{T}} + \Sigma & B\Phi_t^- \\ \Phi_t^- B^{\mathrm{T}} & \Phi_t^- \end{bmatrix}\right) \end{aligned} \tag{5.20}$$

Then, using the theorem of conditioning for jointly Gaussian random variables, we can solve for

$$\mu_t = \mu_t^- + K_t \left(y_t - B\mu_t^- \right) \tag{5.21}$$

$$\Phi_t = (I - K_t B)\,\Phi_t^- \tag{5.22}$$

where the Kalman gain K_t is the $d \times n$ matrix:

$$K_t = \Phi_t^- B^{\mathrm{T}} \left(B\Phi_t^- B^{\mathrm{T}} + \Sigma \right)^{-1} \tag{5.23}$$

The Kalman gain indicates how much the measurement influences the update. When the uncertainty in the measurement is large compared to uncertainty in the state estimate, the Kalman gain is small. On the other hand, when uncertainty in the measurement is small compared to uncertainty in the state estimate, the Kalman gain is large.

To summarize, we implement the Kalman filter by iterating between the one-step prediction (Eqs. 5.18 and 5.19) and the measurement update (Eqs. 5.21 and 5.22) for time steps $t = 1, \ldots, T$. Using this procedure, we obtain the estimated kinematics μ_t that is used to move the computer cursor or robotic limb at each time step t. Φ_t Is the uncertainty around that estimate. These steps are illustrated in Fig. 5.14.

The Kalman filter and OLE have advantages over methods such as PVA because their assumptions are made explicitly and they provide an uncertainty around the state estimate. Having explicit assumptions means that we can easily change our assumptions and derive a different continuous decoder. In practice, the leading iBMI decoders in the field today are variants of the Kalman filter.

An example of a high-performance closed-loop iBMI [22] using a Kalman filter is illustrated in Fig. 5.15. This approach involved two key modifications to the basic Kalman filter. First, the experimenters assumed that the user intended to produce velocities straight to the instructed target at every time step, rather than produce the cursor velocities that were actually decoded (red vectors in Fig. 5.15) following the original calibration. They used this information to improve the de-

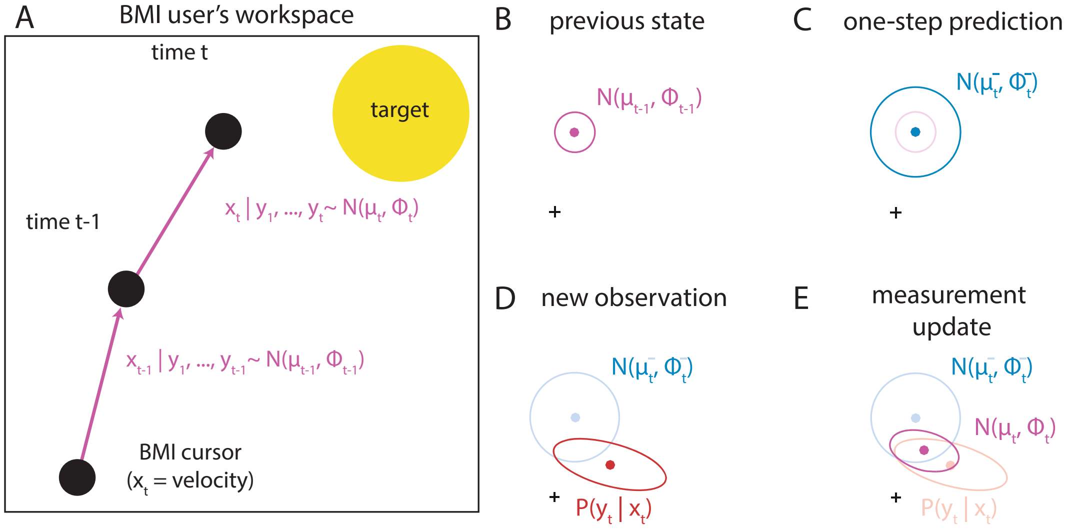


Fig. 5.14 Implementing a Kalman filter. (**a**) At each time step t, we update the cursor's position (black circles) by adding to it the cursor velocity, x_t, which we estimate using a Kalman filter (panels **b**–**e**). (**b**) At time step $t - 1$, we have an estimate of the previous state of the cursor velocity, a Gaussian with mean μ_{t-1} (pink dot), and a covariance Φ_{t-1} (pink ellipse). (**c**) At time step t, we first update the estimated velocity distribution using our Kalman state model, according to the one-step prediction. (**d**) When we observe a new measurement of neural activity, y_t, the Kalman observation model provides us with additional information about the likelihood of the intended cursor velocity. (**e**) In the final step, we use the measurement update to combine the two sources of information about cursor velocity to arrive at our final estimate of the cursor velocity μ_t, which is then used to update the position of the cursor in panel **a**

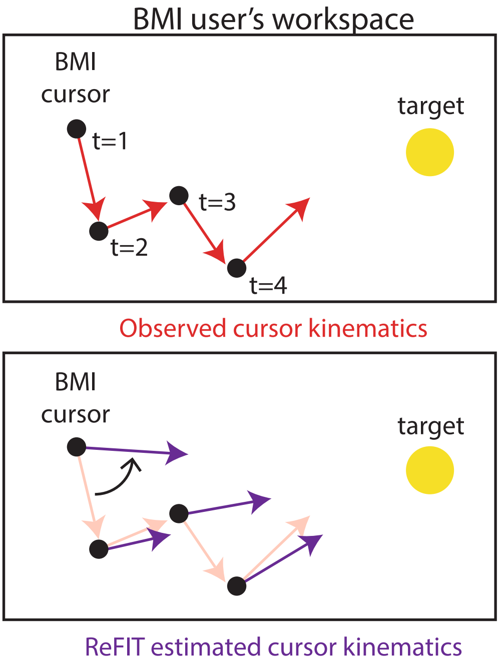


Fig. 5.15 Top: A series of cursor positions (black circles) and cursor velocities (red vectors) from an example trial on which the user was trying to navigate a cursor toward the target (yellow circle). Axes are the horizontal and vertical position in the BMI user's workspace (e.g., as seen on a computer screen). Bottom: Decoding with a Kalman filter can be improved by assuming that the user was trying to produce a velocity straight to the instructed target at every time step. Rather than calibrating the decoder on the observed cursor kinematics (red vectors), the assumed kinematics are obtained by rotating the observed velocities toward the instructed target (purple vectors). The estimate of intended kinematics is regressed against neural activity to obtain the parameters of the ReFIT-KF

coder by rotating the decoded cursor velocities to point toward the target (purple arrows in Fig. 5.15) and using the resulting velocities in a second round of calibration. This requires knowledge of the intended target during the decoder calibration phase. The second change was a causal intervention in which the feedback the user received about the cursor position was taken to be known with no uncertainty. Doing so meant that the user's estimate and the algorithm's estimate of the cursor position were the same, effectively removing the uncertainty in the cursor position. Taking these changes together, they called their extension of the Kalman filter the recalibrated feedback intention-trained Kalman filter or ReFIT-KF [22]. Compared to a standard-velocity Kalman filter, the ReFIT-KF increased performance by reducing the time required to move a computer cursor to hit a target. The ReFIT-KF is currently being used in clinical trials (Fig. 5.3).

5.5 Reanimating Paralyzed Limbs

Our vision is that one day paralyzed people will walk, shake hands, interact with objects, and overall behave in a manner that is virtually indistinguishable from healthy individuals. Although this goal is far from realized, ongoing research is promising. Consider the development of the pacemaker over the past 50 years. The original pacemaker recipient was confined to a wheelchair due to the extensive externalized devices and required daily maintenance from trained care givers. With the assistance of the pacemaker, he lived another 43 years and passed away at 86 from causes unrelated to his heart. These days one would be hard-pressed to determine who has a pacemaker and who does not without an X-ray machine. We anticipate a similar development for iBMI systems.

Most current iBMIs decode kinematic control signals, such as desired velocity, in order to control a computer cursor or a robotic arm. As an output device, robotic arms are most appropriate for amputees, but many potential iBMI users have intact limbs. If we could decode desired muscle activity directly from the brain, we could use functional electrical stimulation (FES) to directly activate a patient's muscles to reanimate their own limbs. This would allow a person with paralysis to regain the ability to interact with the world with their own limbs. Although FES applications have been developed for upper and lower extremity function, bowel and bladder control, and respiratory function, here we will focus on FES for grasping. In this section, we will describe how to decode desired muscle activity and how to deliver electrical stimulation to the muscles in order to match that desired activity.

5.5.1 Functional Electrical Stimulation

Functional electrical stimulation (FES) is neuromuscular stimulation used to restore motor function to paralyzed limbs. This is possible because neurons are electrically excitable. There is an electric potential maintained across the cell membrane. Physiologically, synaptic inputs to a neuron cause a change in the membrane potential, and action potentials are generated when the membrane is depolarized past a certain threshold. Electrical stimulation can artificially depolarize the membrane in a similar way to generate action potentials.

FES electrically stimulates the neurons that are responsible for generating movement, called alpha motor neurons. Alpha motor neurons communicate directly with muscles and are ultimately responsible for generating movement. They cause muscle contractions by releasing the neurotransmitter acetylcholine at the synapse of the alpha motor neuron onto skeletal muscle. This synapse is termed the neuromuscular junction. Acetylcholine binds to receptors on the muscle fiber and generates a muscular action potential that causes the muscle to contract. Although the muscle tissue itself is electrically excitable, most FES systems target the alpha motor neurons because they require less current to generate action potentials than activating the muscle fibers directly (Fig. 5.16). Thus, FES requires the alpha motor neuron to be intact and the neuromuscular junction and muscle to be healthy. These requirements exclude patients with polio, amyotrophic lateral sclerosis, peripheral nerve injuries, and muscular

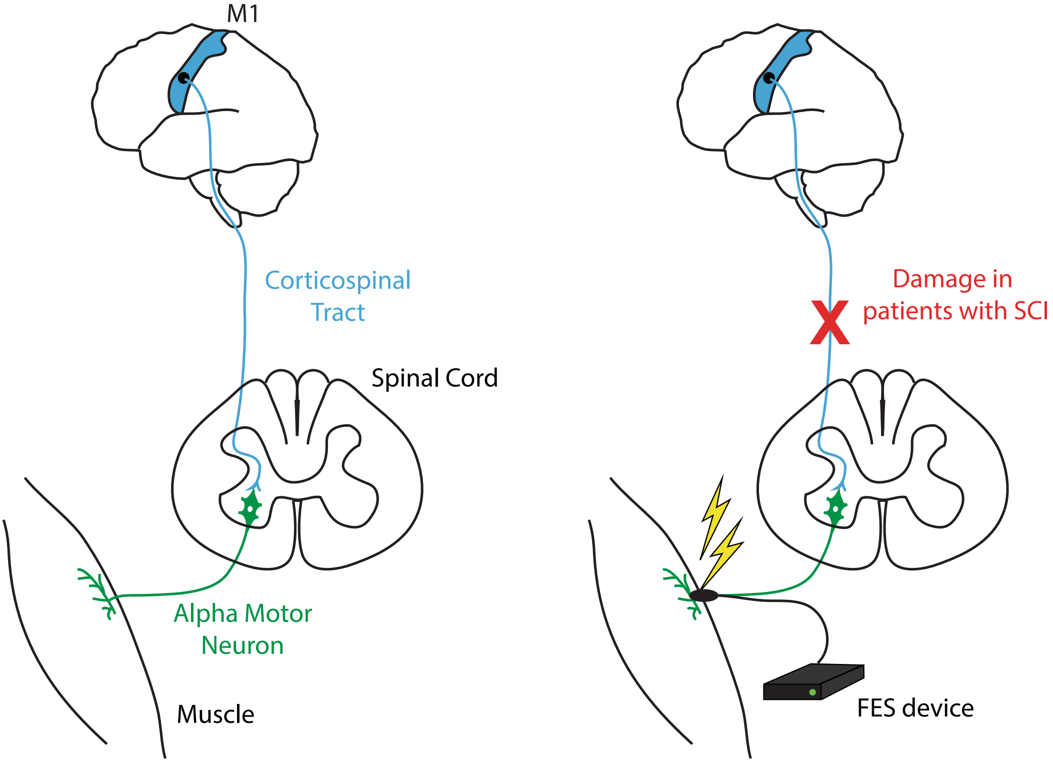


Fig. 5.16 Many neurons in M1 extend down the corticospinal tract in the spinal cord and synapse either directly onto alpha motor neurons or onto interneurons that in turn synapse onto alpha motor neurons. Alpha motor neurons synapse onto muscle fibers at a specialized contact known as the neuromuscular junction. Spinal cord injury (SCI) interrupts the connection between M1 and the muscles. Functional electrical stimulation (FES) artificially generates action potentials at the alpha motor neuron to generate movement in people who are otherwise paralyzed

dystrophies. Patients who can benefit from FES include those with spinal cord injury, stroke, head injuries, cerebral palsy, or multiple sclerosis.

A single alpha motor neuron makes synapses onto several (10–100) muscle fibers. These muscle fibers are driven only by that single motor neuron. Small motor neurons innervate slow-twitch, fatigue-resistant muscle fibers that produce low forces. Large motor neurons innervate fast-twitch, fatigable muscle fibers that produce large forces. Together, a motor neuron and the muscle fibers it innervates are known as a motor unit.

5.5.2 FES Systems

An FES system consists of a controller, electrodes, and a stimulator. The controller regulates the timing and intensity of the delivered stimulation. Stimulation is delivered in the form of pulses of current with waveform patterns such as a square wave or a sine wave. These waveform patterns are described by their frequency, duration, and amplitude (Fig. 5.17). Frequency refers to the number of pulses per second. For FES applications, typically low frequencies are used to produce a smooth contraction at low force levels while minimizing muscle fatigue. The time span of a single pulse is the pulse duration or width. Increasing pulse duration tends to recruit more motor units. Stimulation amplitude describes the strength of the current applied. The higher the amplitude, the stronger the depolarizing effect. This recruits more neurons and results in a stronger muscle contraction. Adjusting these parameters changes the strength of the evoked muscle contraction. Regardless of the particular parameters, FES stimulation typically consists of biphasic, charge-balanced pulses (i.e., the amount of charge injected into the tissue is balanced by the amount of charge drawn out of the tissue) to minimize adverse effects on the tissue and the electrodes.

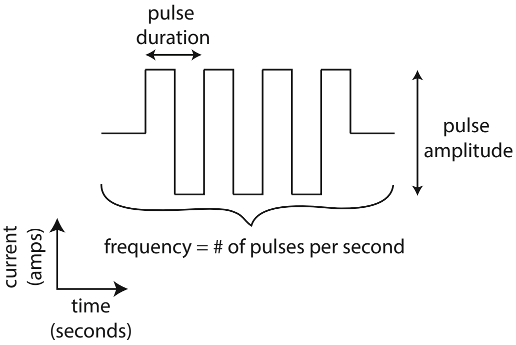


Fig. 5.17 Square pulse waveform

FES electrodes are broadly of two classes: surface electrodes and intramuscular electrodes. Surface electrodes are positioned on the skin over the targeted muscles. Intramuscular electrodes are implanted near the neurons that innervate the targeted muscles. Implanted electrodes have the benefit of being able to recruit muscle fibers more selectively, because they are positioned closer to the neuromuscular junctions. However, surface electrodes are less invasive and easier to replace.

The majority of FES systems in use today do not rely on cortical control signals but rather rely on signals from intact, residual movements. For example, quadriplegics can use a sip/puff tube to control the initiation of a preprogrammed stimulation pattern. Patients who have spinal cord injuries at the level of the fifth to sixth vertebrae of the cervical spinal column retain voluntary control of the muscles above the injury and can shrug their shoulders. Some systems detect the electrical activity when these muscles contract and use it as a control signal for the FES. This is known as myoelectric control. One-dimensional control signals such as these allow the user to control only one degree of freedom. For example, shrugging the shoulder might control the stimulation to open or close the hand and control the degree to which the hand opens and closes. This ultimately limits the number of movements to a few preprogrammed grasps. The first FES system for grasp was developed in the 1960s. It consisted of surface stimulation to open and close the hand [23, 24]. Since then, advances in the electrodes and stimulation paradigms have led to implantable systems.

To access a wider repertoire of movements and ideally greater dexterity, a higher-dimensional control signal is necessary. Using a cortical control signal for FES would enable higher degree-of-freedom control and thus more complex movements. A cortical control signal is also more natural because it taps into the neural activity that controls muscle activity in normal reaching. The goal is that, when the user thinks about reaching, the brain-controlled FES would generate a movement as seamlessly as a normal reach.

5.5.3 Brain-Controlled FES

For a brain-controlled FES system, neural activity is mapped to the stimulation of paralyzed muscles. A simple way to do this is to have the firing rate of a neuron directly control the intensity of the simulation. A group at the University of Washington showed that monkeys could modulate the firing rate of one or two neurons to control the stimulation of temporarily paralyzed wrist muscles to flex and extend the wrist. In this demonstration, when the firing rate of the neuron crossed a certain threshold, current was delivered through the FES in proportion to the neuron's firing rate, allowing the monkey to produce graded muscle contraction force [25]. Brain control of more complex behaviors requires more muscles and more neurons. However, it is not as straightforward as controlling each muscle with a different neuron because the activity of neurons in primary motor cortex is correlated. Instead populations of neurons are used to drive the coordinated activity of the muscles.

We can measure the electrical activity in a muscle while it is contracting. This technique is called electromyography or EMG. The amplitude of the EMG signal is a measure of motor unit activity during muscle activation and is proportional to the magnitude of muscle force. The more active motor units, the higher the measured EMG amplitude and the greater the resulting force. For brain-controlled FES, the goal is to decode the EMG signal that would have naturally resulted from the activity of the recorded M1 neurons and then stimulate the muscles to artificially generate that EMG.

As a proof of concept, researchers at Northwestern University simultaneously recorded EMG activity and neural activity in M1 from an able-bodied monkey while the monkey performed a reaching task [26]. They then used a linear filter (Fig. 5.18a) with multiple inputs (i.e., the recorded neural activity) to predict a single output (i.e., EMG activity from one muscle). The filter can be fit by minimizing the squared error of the predicted EMG. The predicted EMG is a weighted linear combination of the recent history of neural responses from many neurons:

$$\mathrm{EMG}_{\mathrm{linear}}(t) = \sum_{k=1}^{N}\sum_{l=0}^{L} w_{k,l}\, y_k\,(t-l) \qquad (5.24)$$

where l is a time lag, y_k is the firing rate of neuron k, N is the number of neurons in the population, and $w_{k,\,l}$ is the weight that characterizes the effect of neuron k's firing rate at time $(t-l)$ on the EMG signal at time t. Typically, the time history is a few hundred milliseconds in length.

A linear filter of this type does quite well at predicting force signals and muscle activity but often fails to capture specific features of EMG signals. In particular, linear filters often fail to capture the peaks of activity and adequately characterize the quiescent periods between movements. A nonlinear decoder can address these issues, improving predictions by up to 10%. One such option is a Wiener cascade, which is a linear combination of the neural activity passed through a static nonlinearity:

$$\mathrm{EMG}(t) = P\left(w_0 + \sum_{k=1}^{N}\sum_{l=0}^{L} w_{k,l}\, y_k\,(t-l)\right) \qquad (5.25)$$

Here, P is a nonlinear function (e.g., often a polynomial) and w_0 is a bias term. The polynomial is fit between the output of the linear filter and the EMG activity (Fig. 5.18b). The static nonlinearity acts to increase the gain of the peaks and decrease

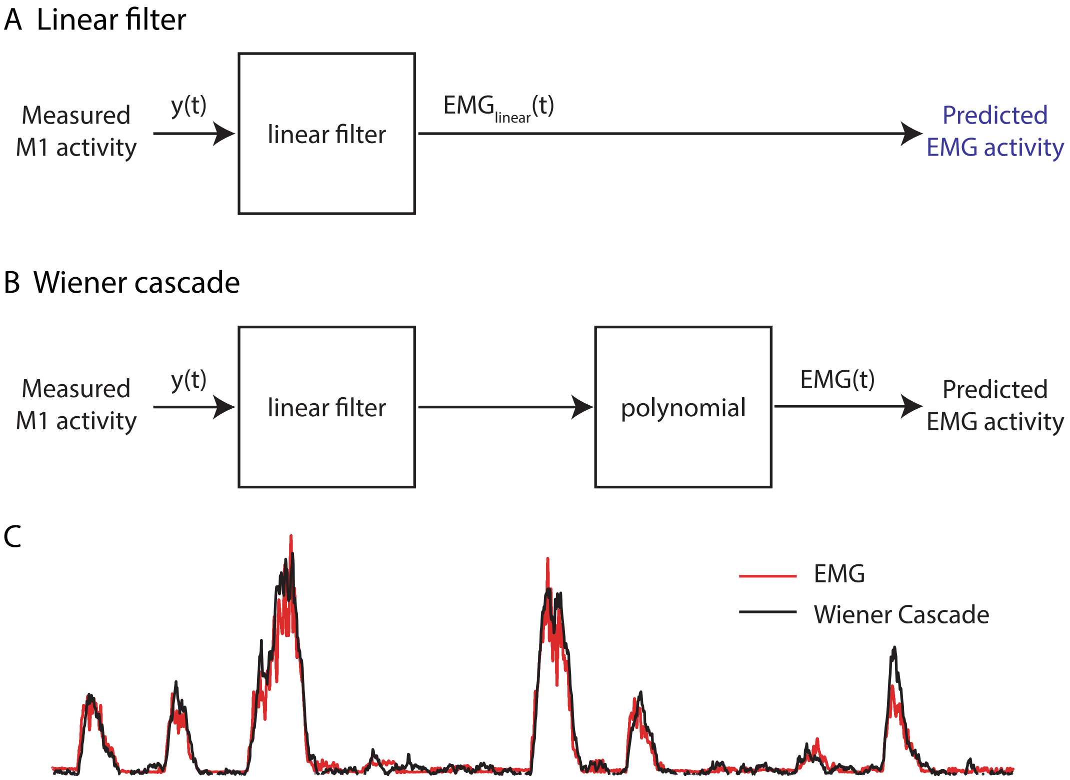


Fig. 5.18 Decoding muscle activity from neural activity. (**a**) Block diagram for a linear filter. (**b**) Block diagram of a Wiener cascade. (**c**) Using a Wiener cascade to predict EMG activity

low-level noise. This is particularly important in an FES application because it reduces unnecessary stimulation during the quiescent periods.

The group at the Northwestern University further showed that the approach of predicting EMG activity from neural population activity could be used in a closed-loop iBMI-FES system in monkeys. They simultaneously recorded neural activity in M1 and EMG activity while the monkeys performed wrist movements or grasping movements. They then temporarily paralyzed the monkeys by injecting lidocaine around the nerves innervating the forearm and hand muscles. Temporarily paralyzed monkeys could use the iBMI-FES system to control stimulation of muscles in the forearm to flex and extend the wrist and to grasp and release a ball (Fig. 5.19; [27]). This demonstration showed that a brain-controlled FES system could allow for more flexible and dexterous movements than was possible with the preprogrammed grasps available through existing FES systems.

Because people who are paralyzed cannot generate EMG activity, the aforementioned methods (which require knowing the intended EMG activity) cannot be directly applied to train a brain-controlled FES for these people. One way to overcome this issue is to take advantage of the fact that the patterns of muscle activity produced in a given task (e.g., grasping) are stereotyped between different individuals. To train a decoder that predicts EMG, it is possible to use the EMG activity measured from a healthy individual as a template and record appropriate neural activity by cueing the user to attempt to generate forces that correspond to the EMG activity. This approach was successful in monkeys [28], suggesting it would be possible to use a similar approach to train a brain-controlled FES for paralyzed people.

Indeed, a group from Case Western University showed that a decoder for FES in a person

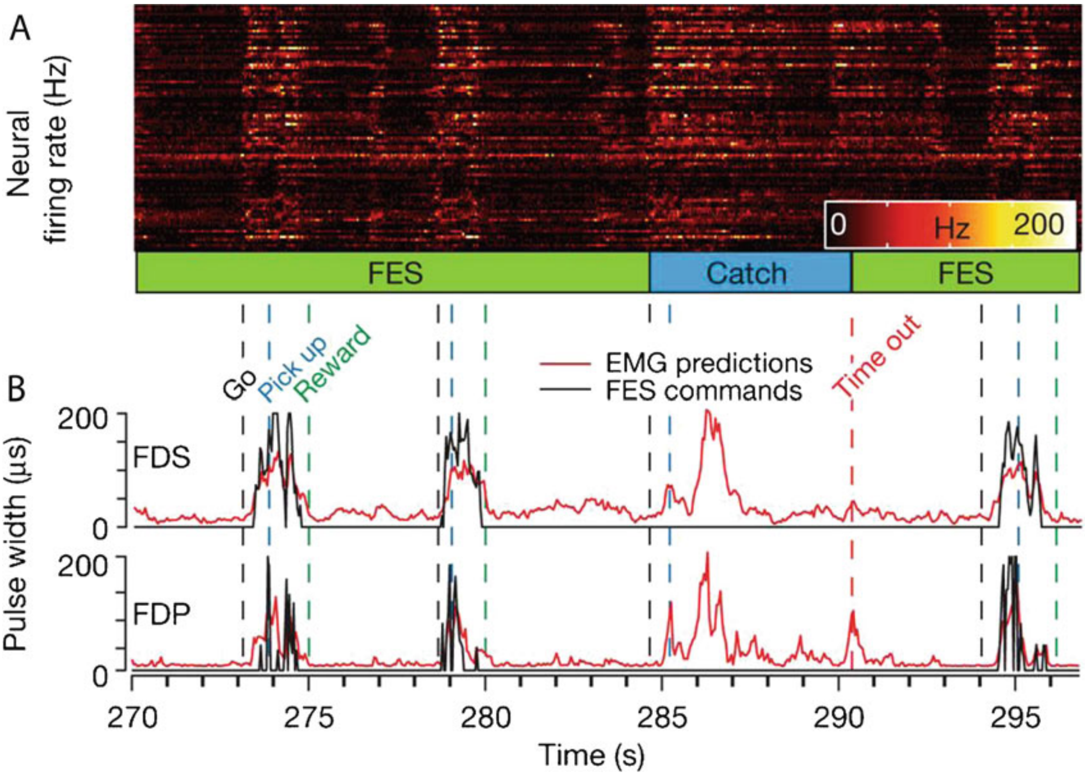


Fig. 5.19 Grasp performance during four consecutive trials in a iBMI-controlled FES ball grasp-and-release task. (**a**) Neural activity plotted as a raster colored by firing rate. (**b**) Predicted muscle activity (red) for two muscles involved in flexing the fingers (flexor digitorum superficialis, FDS; flexor digitorum profundus, FDP). The predicted muscle activity was translated into stimulus commands (black) executed by the stimulator. The vertical dashed lines indicate the progression of successful trials: a go cue (black dashed), the ball was picked up (blue dashed), and the ball was released and the monkey was rewarded (green dashed). When the iBMI-controlled FES system is working well, the monkey modulates his neural activity to drive the stimulation of his muscles, successfully completing the grasp to earn a reward (green dashed). In addition, when the FES system is turned off during "catch" trials, the monkey is unable to complete the trial in the allotted time (red dashed). Note that during this trial, the neurons are firing (**a**) and there is a prediction of EMG activity (red), but no commands (black) are sent to the FES stimulator and the monkey fails to complete the task within 5 seconds (red dashed). (Adapted from Ethier et al. [27])

with spinal cord injury can be trained from the neural activity evoked during attempted movements (Fig. 5.20; [9]). An initial decoder was trained from the neural activity recorded while the participant watched a virtual arm make goal-directed movements and simultaneously attempted to make the same movements. This initial decoder was refined during a virtual reality condition in which his neural activity controlled the movements of a virtual arm. Once the decoder parameters were fixed, the participant performed volitional multi-joint movements of his own FES-actuated arm under brain control (Fig. 5.2). He could perform point-to-point movements with 80–100% accuracy and, in one session, was successful in 11 of 12 attempts at reaching to grab a mug of coffee.

5.5.4 Challenges for FES

An important challenge in the development of FES systems relates to the way motor units are recruited by electrical stimulation. Henneman's size principle states that the natural physiological order of motor unit recruitment is from small

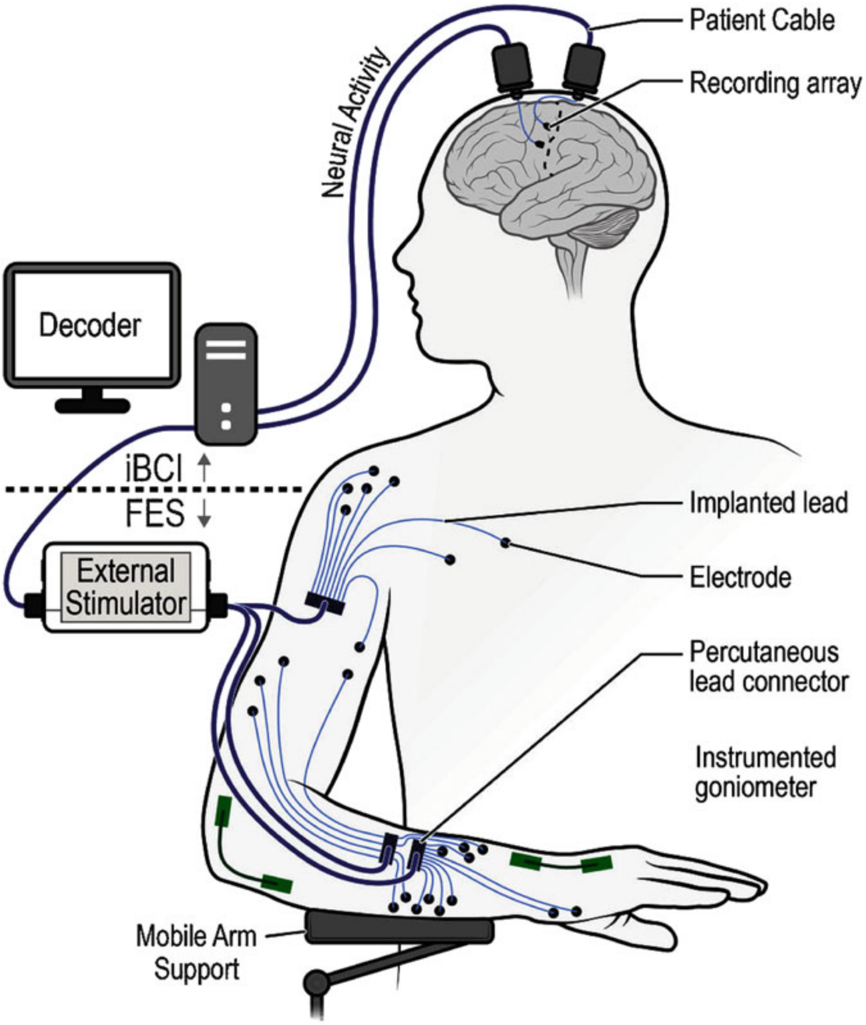


Fig. 5.20 A iBMI-controlled FES system for reaching and grasping. Neural activity was recorded from two arrays implanted into motor cortex. These neural signals were used to control stimulation of intramuscular electrodes implanted in the biceps, triceps, forearm, and hand muscles. The neural signals also actuated the mobile arm support. (Reprinted from Ajiboye et al. [9] with permission)

to large units (Fig. 5.21). That is, motor units that generate small amounts of force are recruited first (allowing for precise control of small movements), and as the force requirements grow larger, the units that generate larger forces (but are consequently not as finely controlled) are recruited. This natural recruitment order arises from Ohm's law, $V = I_{\mathrm{synaptic}} R_{\mathrm{input}}$, where V is voltage, I is the synaptic current, and R is the input resistance. Smaller neurons have smaller membrane surface area, which means they have fewer ion channels and a correspondingly larger input resistance. Thus, they require less synaptic current to change the membrane potential enough to fire action potentials. Similarly, larger neurons have more membrane surface area, more ion channels, and a lower input resistance. So, they require more synaptic current to change the membrane potential enough to fire action potentials. Given a common synaptic drive, the smaller motor units will be recruited before the larger motor units. This natural recruitment order minimizes fatigue

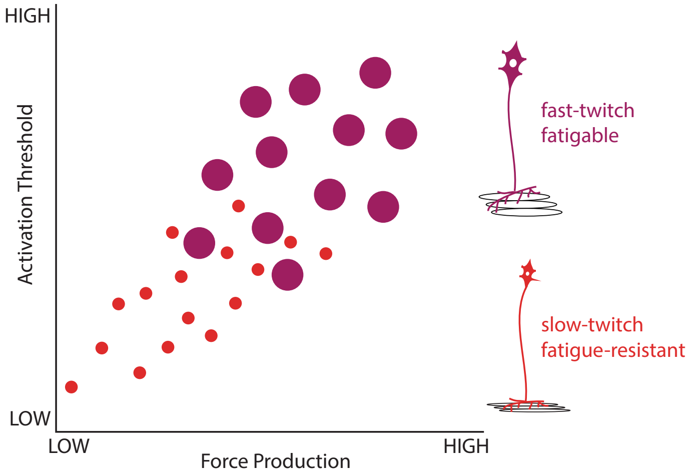


Fig. 5.21 Motor units are naturally recruited from smallest to largest according to Henneman's size principle. This means that slow-twitch, fatigue-resistant motor units (red) are recruited at lower activation thresholds than fast-twitch, fatigable motor units (purple)

by recruiting fatigue-resistant muscle fibers first and only recruiting fatigable fibers when high forces are necessary. This is why it is possible to walk for hours but only sprint for minutes at a time.

By contrast, in an FES system, electrical stimulation recruits motor units in the reverse order of Henneman's size principle. With electrical stimulation, the injected current creates an electric potential across the membrane. The larger neurons have more ions and are easier to depolarize. Thus, in FES the larger motor units are recruited before the smaller motor units. This means that the fatigable motor units are recruited before the fatigue-resistant motor units, which is the reverse of the natural physiological order. The reverse recruitment order limits dexterity and it also causes fatigue. A muscle that is fatigued will produce less force for the same stimulation than a muscle that is not fatigued.

There are some ways to counteract the reverse recruitment order. One such proposal is the utilization of a pre-pulse. A pre-pulse is a pulse several hundred microseconds in duration which precedes the stimulation and hyperpolarizes the neurons, making them less easily excitable. As with other stimulation, a pre-pulse preferentially affects the larger-diameter motor units. With the appropriate parameters, the pre-pulse can be selectively applied so that only the large-diameter axons are hyperpolarized, leaving the small-diameter, fatigue-resistant motor units to be activated by the subsequent stimulation pulse. This type of pre-pulse paradigm has the potential to reduce fatigue under FES conditions by recruiting fatigue-resistant fibers earlier.

Another attempt to mitigate the fatigue problems encountered with FES is to pretreat the muscles with low-level stimulation. This has two benefits. First, the low level of stimulation acts as exercise for the muscles, counteracting the observed increase in fatigability of chronically paralyzed muscles due to disuse [29]. The second benefit is that motor units can actually be changed from fatigable to fatigue-resistant through exercise. Indeed, fatigability profiles can also be changed with low levels of electrical current [30]. Patients implanted with FES systems are often pretreated with low-level stimulation to change muscle fibers toward fatigue-resistant fibers.

5.6 The Future of iBMIs

Brain–machine interfaces have shown promise for restoring motor function to patients with neurological injury or disease. However, there are still many improvements before iBMIs become a widespread treatment for paralysis. In this section, we will discuss ongoing work toward making iBMIs a clinical reality, such as including somatosensory feedback and building better electrodes. Finally, we will end by discussing an emerging new field in which iBMIs are used to answer basic science questions about motor learning and motor control that are currently too difficult to tackle any other way.

5.6.1 Restoring Somatosensory Feedback

Thus far, we have discussed motor control from the perspective of controlling the movements of our body. A critical component of motor control is sensory feedback or sensory information about ongoing movements [31]. All of the iBMIs we have discussed to this point have relied solely on visual feedback. For example, when using an iBMI to control a robotic limb, the user can see where the limb moves. Another equally (if not more) important source of sensory information which we have not yet discussed is somatosensation, which is the sensation of touch, temperature, and proprioception (e.g., body position).

Without somatosensation, the everyday movements that many of us take for granted would be much more difficult. For example, when we carry a heavy box, the texture receptors in our fingertips let us know when the box starts to slip, allowing us to adjust our grip. When we reach to grab a cup of coffee, the temperature receptors in our fingertips tell us that the coffee is too hot to drink. And when we get ready to go outside, our sense of proprioception lets us slide our arms into the sleeves of our jacket without having to turn around. Given the importance of somatosensation during movements such as these, it makes sense that a BMI would benefit from incorporating somatosensory feedback.

In principle, sensory percepts can be restored by electrically stimulating the neural structures responsible for sensation and perception [32]. For amputees, somatosensory feedback could be provided by stimulating the peripheral sensory nerves. However, for patients with quadriplegia due to spinal cord injury, stimulating the nerves would not work because the pathway between the brain and the limb has been disrupted. In this case, the somatosensory cortex could be stimulated directly with intracortical electrodes [33].

One type of movement that is particularly aided by somatosensory feedback is grasping. Grasping an object requires information about contact forces that is difficult to get from visual feedback alone. For example, consider the difference in the forces on the hand when lifting an egg versus a suitcase. Our hands have a variety of touch sensors providing information about shape, weight, size, and texture – information critical for effective grasping. Incorporating similar information into a BMI could improve the degree to which a user could reach and grab a wide variety of objects. A bidirectional BMI (i.e., one that incorporates both motor output and sensory input) could potentially both improve motor function and restore the sense of touch.

5.6.2 Building Better Electrodes

A major obstacle to building clinically viable iBMIs is that the signals recorded from chronically implanted electrode arrays degrade over time. This happens because the electrodes trigger an inflammatory response in the brain that eventually encapsulates the electrodes with a protective layer of glia, forming a “glial scar” [34]. This encapsulation reduces the quality of the recorded signals because the neurons are pushed farther away from the electrode tips. Typically, it is possible to record neural signals from chronically implanted electrodes for months to a few years before the signal degrades. However, it would be unreasonable to expect patients to replace an iBMI (a process involving brain surgery and the associated risks) every few years for the rest of their lives.

There have been a number of attempts to minimize this problem of signal degradation. One such approach is to coat the electrodes with a chemical to minimize scar formation. L1, a neuronal specific cell adhesion molecule, has been shown to minimize glial scar formation [35]. Other neurotrophic chemicals are also being tested. Another approach is to make electrodes that have mechanical characteristics that are more similar to brain tissue. Most electrode arrays today are made with rigid materials, such as tungsten or silicon. The mismatch in the stiffness of the electrodes and the soft brain tissue can induce damage and exacerbate the inflammatory response. To minimize this problem, electrodes can be made from soft materials that are a closer match to the mechanical properties of the brain. Yet another approach is to reduce the diameter of the electrodes. For example, Neuralink is developing electrodes the size of neurons that can be sewn into the cortex with a robot that acts like a sewing machine [36]. Although this work is at its infancy, the hope is that with enough investment, these approaches will lead to longer lasting, more information-rich neural recordings.

5.6.3 iBMIs for Basic Science

As discussed in Sect. 5.2, the more we understand about natural motor control, the better BMI systems will be. In turn, the inverse is true: by studying how the brain functions during control of a BMI, we can gain new insights into the natural processes of motor planning, control, and learning. A BMI is a simplified motor control system compared to arm movement control. When we move our arms, there are hundreds of thousands of output neurons; the mapping from these neurons to movement is unknown; and the arm has nonlinear dynamics that are difficult to measure. All of these characteristics make it difficult to study sensorimotor control during arm reaching. By comparison, an iBMI simplifies all of the features of natural arm reaching, and this makes motor control easier to study: all the output neurons are recorded, the mapping from these neurons to the movement is specified by the experimenter, and the dynamics of the cursor or robotic limb are known and can be made to be simple (e.g., linear). As a result, it is possible to make scientifically causal statements about the relationship between neural activity and behavior (in this case, cursor or robotic limb movements) that are not currently possible when studying arm movements. Together these features make iBMIs a powerful tool for studying sensorimotor control (Table 5.1).

A key feature of sensorimotor control is the ability to learn, adapt, and refine motor skills over time. Our understanding of learning is grounded in concepts of synaptic plasticity and cortical map plasticity. However, we lack an explanation for how such changes give rise to new behavioral capacities. We can leverage an iBMI to establish a causal link between learning-related changes in the brain and new behavioral capacities, because in an iBMI, we record from all of the neurons that drive the behavior, and we as the experimenter define the relationship between the activity of those neurons and the behavior. As discussed in the previous sections, we can begin with a BMI decoder that relates neural activity patterns to cursor velocities in a way that provides proficient control without requiring the user to learn. We can then induce learning by presenting a novel decoder from neural activity to behavior (i.e., cursor velocity). This is akin to giving somebody a flipped computer mouse and asking them to learn to control the cursor.

Table 5.1 Comparison of BMI control to arm reaching

	Arm reaching	iBMI
Effector	Arm	Cursor or robotic limb
Number of non-output neurons	Millions	Millions
Number of output neurons	Thousands (only a subset are recorded)	Tens to hundreds **(all are recorded)**
Neuron-to-movement mapping	Unknown	**Known**
Effector dynamics	Difficult to measure, nonlinear	**Known, can be linear**
Sensory feedback	Tied to the arm	**Flexibly manipulable**

From Golub et al. [37]

Entries in bold indicate components of an iBMI that make it a simplified, well-defined, and easily manipulated system for studying sensorimotor control

Through trial and error, the person can learn to use a flipped mouse. Similarly, through trial and error, the user can learn to use a novel BMI decoder [38]. Because we know exactly how neural activity relates to cursor movement in the iBMI, any observed improvement in behavior (e.g., accuracy of cursor movements) can be attributed to an observed change in the neural activity. We have found that the way in which neurons are interconnected can shape learning that occurs on a timescale of hours [39]. In particular, it is easier to learn tasks requiring population activity patterns that are consistent with the underlying network constraints than tasks requiring novel population activity patterns. On a time scale of days to weeks, populations of neurons can produce new patterns of activity to enable new behavioral capacities [40]. These findings can inform the design of future iBMIs in which we can leverage the user's ability to learn to create even higher-performance iBMI systems [41].

Homework

1. Consider designing a BMI to classify movement to the right or left, and we want to test how well it works with one neuron. If the BMI user intends to move right, the neuron's firing rate is drawn from a Gaussian distribution with mean $\mu_{\text{right}} = 8$ spikes/second and standard deviation $\sigma_{\text{right}} = 5$ spikes/second. If the BMI user intends to move left, the neuron's firing rate is drawn from a Gaussian distribution with mean $\mu_{\text{left}} = 12$ spikes/second and standard deviation $\sigma_{\text{left}} = 6$ spikes/second.
   (a) Suppose we make one measurement of the firing rate, y, and we assume the prior probability of "left" and "right" are equal. For each of the cases below, would we classify "left" or "right"?

y (Spikes/second)	2	5	8	11	14	17
Classification						

   (b) Suppose we now assume that the BMI user moves "left" twice as often as "right" (i.e., $P(\text{left}) = 2/3$ and $P(\text{right}) = 1/3$. For each of the cases below, would we classify "left" or "right"?

y (Spikes/second)	2	5	8	11	14	17
Classification						

2. In Sect. 5.3 we showed how to implement a classifier with Gaussian firing statistics, where the neural activity for class k is modeled as $\boldsymbol{y} \sim N(\boldsymbol{\mu}_k, \Sigma_k)$, where $\boldsymbol{\mu}_k \in R^d$ and $\Sigma_k \in R^{d \times d}$ are the mean and covariance of the activity of a population of d neurons. Here we will assume that the covariance matrix is the same for each class $k = 1, \ldots, K$ (i.e., $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_K$).
   (a) First, suppose we have a new recording of neural activity, $\boldsymbol{y}$. Also, suppose that $P(c_k) = \pi_k$. Using Bayes'rule, find $\log P(c_k | \boldsymbol{y})$, up to the normalizing constant.
   (b) Find the decision boundary used for determining whether the point $\mathbf{y}$ came from class j or class k, and simplify the expression.
   (c) Is the decision boundary linear?
3. In this problem we will derive the equations to implement a classifier based on Poisson spike counts. The spike count of neuron i given class k is Poisson-distributed with parameter λ_{ik}. We will assume that the D neurons, $y_1, \ldots, y_D$ are conditionally independent given the class j. In other words, given neural activity $\boldsymbol{y} \in \mathrm{R}^D$, the probability that $\boldsymbol{y}$ came from class k is as follows:

$$P(\mathbf{y}|c_k) = \prod_{\mathrm{i}=1}^{D} P(y_i|c_k), \quad \text{where}$$

$$P(y_i|c_k) = \exp(-\lambda_{ik})\, \lambda_{ik}^{y_i} / y_i!$$

(a) Let $P(c_k) = \pi_k$. Find $P(c_k|\boldsymbol{y})$ using Bayes rule.

Now simplify the expression above by taking the log: $\log P(c_k \mid \boldsymbol{y})$.

(b) Given a new point $\boldsymbol{y}$, we want to determine to which class this point belongs. Derive the decision boundary that determines whether we classify a new point $\boldsymbol{y}$ as belonging to either class j or class k. Use the expression that you derived in part a.

(c) Is the decision boundary linear?

4. In Sect. 5.3 we provided the following expressions for the training phase of a classifier: $\boldsymbol{\mu}_k = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{y}_i$ (Eq. 5.9) and $\Sigma_k = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_k\right)\left(\boldsymbol{y}_i - \boldsymbol{\mu}_k\right)^\top$(Eq. 5.10), where $\boldsymbol{y}_i \in R^d$ for all $i = 1, \ldots, N$ is the neural activity recorded with class k, $\boldsymbol{\mu}_k \in R^d$, and $\Sigma^k \in R^{d\times d}$. Show that these values of $\boldsymbol{\mu}_k$ and Σ_k maximize the following equation for the likelihood:

$$L\left(\boldsymbol{\mu}_k, \Sigma_k \,|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{N,}c_k\right) = P\left(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N \,|c_k\right)$$

$$= \prod_{i=1}^{N}(2\pi)^{-d/2}|\Sigma_k|^{-1/2}$$

$$\exp\left(-\frac{1}{2}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_k\right)^\top \Sigma_k{}^{-1}\left(\boldsymbol{y}_i - \boldsymbol{\mu}_k\right)\right)$$

5. In Sect. 5.4, we considered a two-neuron example of the PVA decoder where the neurons had orthogonal preferred directions (e.g., one neuron preferred 90°, while the other neuron preferred 180°). Show that if the two neurons do not have orthogonal tuning directions, the directions decoded by PVA will be biased.

6. Show that a neuron that exhibits cosine tuning also shows linear tuning to velocity. That is, suppose that given a reach in the θ direction with speed s, a neuron's firing rate can be written as $y = b_0 + ms\,\cos\left(\theta - \theta_{\vec{p}}\right)$, where b_0 is the neuron's baseline firing rate, m is its modulation depth, and $\theta_{\vec{p}}$ is the neuron's preferred direction. Show that this means we can also write $y = \mathrm{b}_0 + \mathbf{b}^T\mathbf{v}$, where $\mathbf{b}$ and $\mathbf{v}$ are both 2D vectors.

7. Derive the expressions for the training phase of the Kalman filter in Sect. 5.4:

$$B = \left(\sum_{t=1}^{T}\mathbf{y}_t\mathbf{x}_t^T\right)\left(\sum_{t=1}^{T}\mathbf{x}_t\mathbf{x}_t^T\right)^{-1}$$

$\Sigma = \frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{y}_t - B\mathbf{x}_t\right)\left(\mathbf{y}_t - B\mathbf{x}_t\right)^T$ (Note that here we use the B found above.)

$$A = \left(\sum_{t=2}^{T}\boldsymbol{x}_t\boldsymbol{x}_{t-1}^T\right)\left(\sum_{t=2}^{T}\boldsymbol{x}_{t-1}\boldsymbol{x}_{t-1}^T\right)^{-1}$$

$Q = \frac{1}{T-1}\sum_{t=2}^{T}\left(\boldsymbol{x}_t - A\boldsymbol{x}_{t-1}\right)\left(\boldsymbol{x}_t - A\boldsymbol{x}_{t-1}\right)^T$ (Note that here we use the A found above.)

8. Consider using a BMI to play Pong with one neuron. That is, we will use a Kalman filter to decode position along a one-dimensional axis from the firing rate of a single neuron. Let the state model be $x_t = x_{t-1} + \omega_t$, $\omega_t \sim N(0, q)$ and the observation model be $y_t = \mathrm{b}x_t + \varepsilon_t$, $\varepsilon_t \sim N(0, \sigma)$.

(a) Show that the estimate of the position on time step t, μ_t, can be written in the form

$$\mu_t = (1 - \alpha)\,\mu_{t-1} + \alpha\left(\frac{\mathrm{y}_t}{\mathrm{b}}\right)$$

(b) Prove that $0 \leq \alpha \leq 1$.

(c) When does α approach 0? Under this case, why does it make sense for $\mu_t = \mu_{t-1}$?

(d) When does α approach 1? Under this case, why does it make sense for $\mu_t = y_t/b$?

9. You decide to speed up the implementation of your Kalman filter by skipping the one-step prediction. Whereas normally you would solve the measurement update (Eq. 5.17) and one-step predictions iteratively on each time step (Eq. 5.16).

You instead decide to just iterate the measurement update step, by directly plugging in the velocity estimate from the previous time step, $P\left(\boldsymbol{x}_{t-1}|\{\boldsymbol{y}\}_1^{t-1}\right)$, without making a one-step prediction:

$$P\left(\boldsymbol{x}_t|\{\boldsymbol{y}\}_1^t\right) = \frac{P\left(\boldsymbol{y}_t|\boldsymbol{x}_t\right)P\left(\boldsymbol{x}_{t-1}|\{\boldsymbol{y}\}_1^{t-1}\right)}{P\left(\boldsymbol{y}_t|\{\boldsymbol{y}\}_1^{t-1}\right)}$$

Describe qualitatively what will happen to the velocity estimate over time.

(Hint: when in doubt, try simulating it or solving the 1D case.)

10. The goal of the measurement update of the Kalman filter is to find $P(\boldsymbol{x}_t|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t)$. To do so, we adopted the strategy in Sect. 5.4 whereby we would first find the joint distribution $P(\boldsymbol{x}_t, \boldsymbol{y}_t|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1})$, and then use the theorem of conditioning for jointly Gaussian random variables to find $P(\boldsymbol{x}_t|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t)$. Here we will derive the means and covariances of the joint distribution

$$\begin{bmatrix} \boldsymbol{y}_t \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1} \\ \boldsymbol{x}_t \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1} \end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix} B\mu_{\mathrm{t}}^{-} \\ \mu_{\mathrm{t}}^{-} \end{bmatrix}, \begin{bmatrix} B\Phi_t^{-}B^{\mathrm{T}} + \Sigma & B\Phi_t^{-} \\ \Phi_t^{-}B^{\mathrm{T}} & \Phi_t^{-} \end{bmatrix} \right)$$

(a) Find the mean of $y_t \mid y_1, \ldots, y_{t-1}$.
(b) Find the variance of $y_t \mid y_1, \ldots, y_{t-1}$.
(c) Find the covariance of $\boldsymbol{x}_t$, $\boldsymbol{y}_t$ when both are conditioned on $\boldsymbol{y}_t, \ldots, \boldsymbol{y}_{t-1}$.

11–12. We have provided a dataset (https://github.com/emilyoby/bmi-data-set) consisting of center-out arm reaches and neural activity recorded from a Utah electrode array implanted in M1. The following describes the data format. The .mat file has two data structures: ‘trainTrials’ contains 180 trials to be used as training data, and testTrials contains 8 trials to be used as test data. Each data structure contains ‘spikes’, ‘handPos’, and ‘handVel’ variables, representing the spiking activity, hand position, and hand velocity, respectively, on each trial in which a monkey reached to one of eight different targets. The ‘spikes’ variable contains, for each trial, the number of threshold crossings in 50 ms bins recorded simultaneously from the 91 electrodes and has dimensions (n time steps) × (91 electrodes), where n is the number of time steps within a particular trial. For example, ‘trainTrials.spikes{i}(n,k)’ contains the number of threshold crossings recorded on the kth electrode in the nth time step of the ith trial. The ‘handPos’ and ‘handVel’ variables are structured similarly and contain the 2D hand position (in mm) and velocity (in mm/sec), respectively, for the same time steps as in the ‘spikes’ variable.

For the problems below, use the provided neural and kinematic data to implement the continuous decoders discussed in Sect. 5.4.

11. Use PVA decoder to estimate the movement velocity during a center out task.
    (a) Fit the parameters of the decoder using the 180 trials of training data.
    (b) Test the decoder on the eight test trials. Plot the decoded trajectories and the actual movement trajectories on the same plot.
    (c) Try improving the decoding by smoothing the firing rates by using a running average of the firing rates during the previous 250 ms.
12. Use a Kalman filter decoder to estimate the movement trajectory for each trial.
    (a) Fit the parameters of the decoder using the 180 trials of training data.
    (b) Test the decoder on the eight test trials. Plot the decoded trajectories and the actual movement trajectories on the same plot.

References

1. J.K. Chapin, K.A. Moxon, R.S. Markowitz, M.A. Nicolelis, Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. Nat. Neurosci. **2**(7), 664–670 (1999). https://doi.org/10.1038/10223
2. J. Wessberg, C.R. Stambaugh, J.D. Kralik, P.D. Beck, M. Laubach, J.K. Chapin, J. Kim, S.J. Biggs, M.A. Srinivasan, M.A. Nicolelis, Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature **408**(6810), 361–365 (2000). https://doi.org/10.1038/35042582
3. M.D. Serruya, N.G. Hatsopoulos, L. Paninski, M.R. Fellows, J.P. Donoghue, Instant neural control of a movement signal. Nature **416**(6877), 141–142 (2002). https://doi.org/10.1038/416141a

4. D.M. Taylor, S.I. Helms Tillery, A.B. Schwartz, Direct Cortical Control of 3D Neuroprosthetic Devices. Science (New York, N.Y.) **296**(5574), 1829–1832 (2002). https://doi.org/10.1126/science.1070291
5. J.M. Carmena, M.A. Lebedev, R.E. Crist, J.E. O'Doherty, D.M. Santucci, D.F. Dimitrov, P.G. Patil, C.S. Henriquez, M.A.L. Nicolelis, Learning to control a brain-machine interface for reaching and grasping by primates. PLoS Biol. **1**(2), E42 (2003). https://doi.org/10.1371/journal.pbio.0000042
6. J.L. Collinger, B. Wodlinger, J.E. Downey, W. Wang, E.C. Tyler-Kabara, D.J. Weber, A.J.C. McMorland, M. Velliste, M.L. Boninger, A.B. Schwartz, High-performance neuroprosthetic control by an individual with tetraplegia. Lancet (London, England) **381**(9866), 557–564 (2013). https://doi.org/10.1016/S0140-6736(12)61816-9
7. B. Wodlinger, J.E. Downey, E.C. Tyler-Kabara, A.B. Schwartz, M.L. Boninger, J.L. Collinger, Ten-dimensional anthropomorphic arm control in a human brain-machine interface: Difficulties, solutions, and limitations. J. Neural Eng. **12**(1), 016011 (2015). https://doi.org/10.1088/1741-2560/12/1/016011
8. C.E. Bouton, A. Shaikhouni, N.V. Annetta, M.A. Bockbrader, D.A. Friedenberg, D.M. Nielson, G. Sharma, P.B. Sederber, B.C. Glenn, W.J. Mysiw, A.G. Morgan, M. Deogaonkar, A.R. Rezai, Restoring cortical control of functional movement in a human with quadriplegia. *Nature***533**(7602), 247–250 (2016). https://doi.org/10.1038/nature17435
9. A.B. Ajiboye, F.R. Willett, D.R. Young, W.D. Memberg, B.A. Murphy, J.P. Miller, B.L. Walter, et al., Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: A proof-of-concept demonstration. Lancet (London, England) **389**(10081), 1821–1830 (2017). https://doi.org/10.1016/S0140-6736(17)30601-3
10. V. Gilja, C. Pandarinath, C.H. Blabe, P. Nuyujukian, J.D. Simeral, A.A. Sarma, B.L. Sorice, et al., Clinical translation of a high-performance neural prosthesis. Nat. Med. **21**(10), 1142–1145 (2015). https://doi.org/10.1038/nm.3953
11. B. Jarosiewicz, A.A. Sarma, D. Bacher, N.Y. Masse, J.D. Simeral, B. Sorice, E.M. Oakley, et al., Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. Sci. Transl. Med. **7**(313), 313ra179 (2015). https://doi.org/10.1126/scitranslmed.aac7328
12. C. Pandarinath, P. Nuyujukian, C.H. Blabe, B.L. Sorice, J. Saab, F.R. Willett, L.R. Hochberg, K.V. Shenoy, J.M. Henderson, High performance communication by people with paralysis using an intracortical brain-computer interface. elife **6** (2017). https://doi.org/10.7554/eLife.18554
13. S. Perel, P.T. Sadtler, E.R. Oby, S.I. Ryu, E.C. Tyler-Kabara, A.P. Batista, S.M. Chase, Single-unit activity, threshold crossings, and local field potentials in motor cortex differentially encode reach kinematics. J. Neurophysiol. **114**(3), 1500–1512 (2015). https://doi.org/10.1152/jn.00293.2014
14. E.V. Evarts, Relation of pyramidal tract activity to force exerted during voluntary movement. J. Neurophysiol. **31**(1), 14–27 (1968). https://doi.org/10.1152/jn.1968.31.1.14
15. A.P. Georgopoulos, J.F. Kalaska, R. Caminiti, J.T. Massey, On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. J. Neurosci. Off. J. Soc. Neurosci. **2**(11), 1527–1537 (1982)
16. N. Hatsopoulos, J. Joshi, J.G. O'Leary, Decoding continuous and discrete motor behaviors using motor and premotor cortical ensembles. J. Neurophysiol. **92**(2), 1165–1174 (2004). https://doi.org/10.1152/jn.01245.2003
17. S. Kakei, D.S. Hoffman, P.L. Strick, Direction of action is represented in the ventral premotor cortex. Nat. Neurosci. **4**(10), 1020–1025 (2001). https://doi.org/10.1038/nn726
18. G. Santhanam, S.I. Ryu, B.M. Yu, A. Afshar, K.V. Shenoy, A high-performance brain-computer interface. Nature **442**(7099), 195–198 (2006). https://doi.org/10.1038/nature04968
19. M. Velliste, P. Sagi, M. Chance Spalding, A.S. Whitford, A.B. Schwartz, Cortical control of a prosthetic arm for self-feeding. Nature **453**(7198), 1098–1101 (2008). https://doi.org/10.1038/nature06996
20. K.V. Shenoy, D. Meeker, S. Cao, S.A. Kureshi, B. Pesaran, C.A. Buneo, A.P. Batista, P.P. Mitra, J.W. Burdick, R.A. Andersen, Neural prosthetic control signals from plan activity. Neuroreport **14**(4), 591–596 (2003). https://doi.org/10.1097/00001756-200303240-00013
21. S.M. Chase, A.B. Schwartz, R.E. Kass, 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 (2009). https://doi.org/10.1016/j.neunet.2009.05.005
22. V. Gilja, P. Nuyujukian, C.A. Chestek, J.P. Cunningham, B.M. Yu, J.M. Fan, M.M. Churchland, et al., A high-performance neural prosthesis enabled by control algorithm design. Nat. Neurosci. **15**(12), 1752–1757 (2012). https://doi.org/10.1038/nn.3265
23. C. Long, An electrophysiologic splint for the hand. Arch. Phys. Med. Rehabil. **44**(September), 499–503 (1963)
24. L. Vodovnik, C. Long, J.B. Reswick, A. Lippay, D. Starbuck, Myo-electric control of paralyzed muscles. I.E.E.E. Trans. Biomed. Eng. **12**(3), 169–172 (1965). https://doi.org/10.1109/tbme.1965.4502374
25. C.T. Moritz, E.E. Fetz, Volitional control of single cortical neurons in a brain-machine interface. J. Neural Eng. **8**(2), 025017 (2011). https://doi.org/10.1088/1741-2560/8/2/025017
26. E.A. Pohlmeyer, S.A. Solla, E.J. Perreault, L.E. Miller, Prediction of upper limb muscle activity from motor cortical discharge during reaching. J. Neural Eng. **4**(4), 369–379 (2007). https://doi.org/10.1088/1741-2560/4/4/003

27. C. Ethier, E.R. Oby, M.J. Bauman, L.E. Miller, Restoration of grasp following paralysis through brain-controlled stimulation of muscles. Nature **485**(7398), 368–371 (2012). https://doi.org/10.1038/nature10987
28. C. Ethier, D. Acuna, S.A. Solla, L.E. Miller, Adaptive neuron-to-EMG decoder training for FES neuroprostheses. J. Neural Eng. **13**(4), 046009 (2016). https://doi.org/10.1088/1741-2560/13/4/046009
29. K. Singh, F.J. Richmond, G.E. Loeb, Recruitment properties of intramuscular and nerve-trunk stimulating electrodes. IEEE Trans. Rehabil. Eng. **8**(3), 276–285 (2000)
30. J.T. Mortimer, Motor prostheses, in *Handbook of physiology: The nervous system II*, (American Physiological Society, Bethesda, MD, 1981)
31. S.J. Bensmaia, L.E. Miller, Restoring sensorimotor function through intracortical interfaces: Progress and looming challenges. Nat. Rev. Neurosci. **15**(5), 313–325 (2014). https://doi.org/10.1038/nrn3724
32. M.C. Dadarlat, J.E. O'Doherty, P.N. Sabes, A learning-based approach to artificial sensory feedback leads to optimal integration. Nat. Neurosci. **18**(1), 138–144 (2015). https://doi.org/10.1038/nn.3883
33. S.N. Flesher, J.L. Collinger, S.T. Foldes, J.M. Weiss, J.E. Downey, E.C. Tyler-Kabara, S.J. Bensmaia, A.B. Schwartz, M.L. Boninger, R.A. Gaunt, Intracortical microstimulation of human somatosensory cortex. Sci. Transl. Med. **8**(361), 361ra141 (2016). https://doi.org/10.1126/scitranslmed.aaf8083
34. V.S. Polikov, P.A. Tresco, W.M. Reichert, Response of brain tissue to chronically implanted neural electrodes. J. Neurosci. Methods **148**(1), 1–18 (2005). https://doi.org/10.1016/j.jneumeth.2005.08.015
35. J.R. Eles, A.L. Vazquez, N.R. Snyder, C. Lagenaur, M.C. Murphy, T.D.Y. Kozai, X.T. Cui, Neuroadhesive L1 coating attenuates acute microglial attachment to neural electrodes as revealed by live two-photon microscopy. Biomaterials **113**, 279–292 (2017). https://doi.org/10.1016/j.biomaterials.2016.10.054
36. T.L. Hanson, C.A. Diaz-Botia, V. Kharazia, M.M. Maharbiz, P.N. Sabes, The 'sewing machine' for minimally invasive neural recording. *BioRxiv*, March. https://doi.org/10.1101/578542 (2019)
37. M.D. Golub, S.M. Chase, A.P. Batista, B.M. Yu, Brain-computer interfaces for dissecting cognitive processes underlying sensorimotor control. Curr. Opin. Neurobiol. **37**(April), 53–58 (2016). https://doi.org/10.1016/j.conb.2015.12.005
38. K. Ganguly, J.M. Carmena, Emergence of a stable cortical map for neuroprosthetic control. PLoS Biol. **7**(7), e1000153 (2009). https://doi.org/10.1371/journal.pbio.1000153
39. P.T. Sadtler, K.M. Quick, M.D. Golub, S.M. Chase, S.I. Ryu, E.C. Tyler-Kabara, B.M. Yu, A.P. Batista, Neural constraints on learning. Nature **512**(7515), 423–426 (2014). https://doi.org/10.1038/nature13665
40. E.R. Oby, M.D. Golub, J.A. Hennig, A.D. Degenhart, E.C. Tyler-Kabara, B.M. Yu, S.M. Chase, A.P. Batista, New neural activity patterns emerge with long-term learning. Proc. Natl. Acad. Sci. U. S. A. **116**(30), 15210–15215 (2019). https://doi.org/10.1073/pnas.1820296116
41. K.V. Shenoy, J.M. Carmena, Combining decoder design and neural adaptation in brain-machine interfaces. Neuron **84**(4), 665–680 (2014). https://doi.org/10.1016/j.neuron.2014.08.038
42. L.R. Hochberg, M.D. Serruya, G.M. Friehs, J.A. Mukand, M. Saleh, A.H. Caplan, A. Branner, D. Chen, R.D. Penn, J.P. Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature. 442(7099), 164–71 (2006).
43. L.R. Hochberg, D. Bacher, B. Jarosiewicz, N.Y. Masse, J.D. Simeral, J. Vogel, S. Haddadin, J. Liu, S.S. Cash, P. Van Der Smagt, J.P. Donoghue. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. Nature. 485(7398), 372–5 (2012).