CHAPTER 1

OPTIMAL SIGNAL PROCESSING FOR BRAIN–MACHINE INTERFACES

Justin C. Sanchez and Jose C. Principe

1.1 INTRODUCTION

Several landmark experimental paradigms have shown the feasibility of using neuroprosthetic devices to restore motor function and control in individuals who are "locked in" or have lost the ability to control movement of their limbs [1–19]. In these experiments, researchers seek to both rehabilitate and augment the performance of neural–motor systems using brain–machine interfaces (BMIs) that transfer the intent of the individual (as collected from the cortex) into control commands for prosthetic limbs and/or computers. Brain–machine interface research has been strongly motivated by the need to help the more than 2 million individuals in the United States suffering from a wide variety of neurological disorders that include spinal cord injury and diseases of the peripheral nervous system [20]. While the symptoms and causes of these disabilities are diverse, one characteristic is common in many of these neurological conditions: Normal functioning of the brain remains intact. If the brain is spared from injury and control signals can be extracted, the BMI problem becomes one of finding *optimal* signal processing techniques to efficiently and accurately convert these signals into operative control commands.

A variety of noninvasive and invasive techniques have been used to collect control signals from the cortex. Some of the earliest brain–computer interfaces (BCIs) utilized electrical potentials collected from the scalp through the use of electroencephalography (EEG) [21]. This approach to interfacing individuals with machines has the appeal of a low threshold of clinical use since no surgical procedures are required to install the sensors detecting the control signals. Additionally, EEG hardware technology is at a stage where it is relatively inexpensive, portable, and easy to use. In terms of ease of access to the neurophysiology, EEG is an attractive choice; however, in terms of signal processing, this approach has been limited to basic communication capabilities with a computer screen and suffers from a low bandwidth (a few bits per second). The fundamental difficulty of using EEG for BCI is due to the "spatiotemporal filtering" of neuronal activity resulting from the different conductivities of the scalp, skull, and dura, which limits the signal-to-noise ratio of the time series and blurs the localization of the neural population firings [22]. Long training sessions are often required for BCI users to learn to modulate their neuronal activity, and users can respond only after minutes of concentration. Another approach to noninvasively collecting control signals is through the use

Handbook of Neural Engineering. Edited by Metin Akay

of near-infrared (NIR) spectroscopy, which uses the scattering of light to detect blood oxygenation [23]. Like EEG, NIR spectroscopy is also a noninvasive technology with relatively course spatial resolution, however the temporal resolution is on the order of tens of milliseconds. To overcome the difficulties of recording control signals through the skull, researchers have moved to using a more invasive technique of measuring EEG on the surface of the brain with the electrocorticogram (ECoG). By placing dense arrays directly upon the motor cortex, this approach has the appeal of increasing the spatial resolution of EEG, and studies are now underway to assess the utility of the collected signals [24]. Recently, invasive techniques that utilize multiple arrays of microelectrodes that are chronically implanted into the cortical tissue have shown the most promise for restoring motor function to disabled individuals [10]. It has been shown that the firing rates of single cells collected from multiple cortices contain precise information about the motor intent of the individual. A variety of experimental paradigms have demonstrated that awake, behaving primates can learn to control external devices with high accuracy using optimal signal processing algorithms to interpret the modulation of neuronal activity as collected from the microelectrode arrays. A recent special issue of the *IEEE Transactions on Biomedical Engineering* (June 2004) provides a very good overview of the state of the art.

A conceptual drawing of a BMI is depicted in Figure 1.1, where neural activity from hundreds of cells is recorded (step 1), conditioned (step 2), and translated (step 3) directly into hand position (HP), hand velocity (HV), and hand gripping force (GF) of a prosthetic arm or cursor control for a computer. The focus of this chapter is centered on step 3 of the diagram where we will use optimal signal processing techniques to find the functional relationship between neuronal activity and behavior. From an optimal signal processing point of view, BMI modeling in step 3 is a challenging task because of several factors: the intrinsic partial access to the motor cortex information due to the spatial subsampling of the neural activity, the unknown aspects of neural coding, the huge dimensionality of the problem, the noisy nature of the signal pickup, and the need for real-time signal processing algorithms. The problem is further complicated by the need for good generalization in nonstationary environments, which is dependent upon model topologies, fitting criteria, and training algorithms. Finally, we must contend with reconstruction accuracy, which is linked to our choice of linear-versus-nonlinear and feedforward-versus-feedback models.

Since the basic biological and engineering challenges associated with optimal signal processing for BMI experiments requires a highly interdisciplinary knowledgebase involving neuroscience, electrical and computer engineering, and biomechanics, the BMI modeling problem will be addressed in several steps. First, an overview of the pioneering modeling approaches will give the reader depth into what has been accomplished in this area of research. Second, we will familiarize the reader with characteristics of the

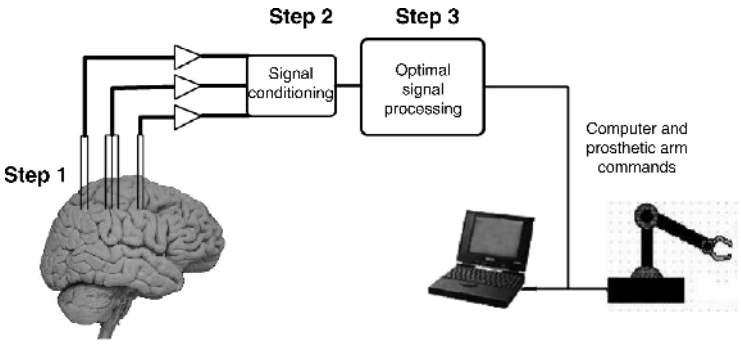

Figure 1.1 Conceptual drawing of BMI components.

neural recordings that the signal processing methods utilize. Third, we cover in detail the current approaches to modeling in BMIs. Finally, real implementations of optimal models for BMIs will be presented and their performance compared.

1.2 HISTORICAL OVERVIEW OF BMI APPROACHES/MODELS

The foundations of BMI research were probed in the early 1980s by E. Schmidt and E. Fetz, who were interested in finding out if it was possible to use neural recordings from the motor cortex of a primate to control an external device [1, 25]. In this pioneering work, Schmidt measured how well primates could be conditioned to modulate the firing patterns of *single* cortical cells using a series of eight target lamps each symbolizing a cellular *firing rate* that the primate was required to produce. The study did confirm that a primate was able to modulate neural firing to match the target rates and additionally estimated the information transfer rate in the neural recordings to be half that of using the intact motor system as the output. With this result, Schmidt proposed that engineered interfaces could be designed to use modulations of neural firing rates as control signals.

Shortly after Schmidt [1] published his results, Georgopoulos et al. [2] presented a theory for neural *population* coding of hand kinematics as well as a method for reconstructing hand trajectories called the population vector algorithm (PVA) [3]. Using center-out reaching tasks, Georgopoulos proposed that each cell in the *motor cortex* has a "preferred hand direction" for which it fires maximally and the distribution of cellular firing over a range of movement directions could be characterized by a simple cosine function [2]. In this theory, arm movements were shown to be constructed by a population "voting" process among the cells; each cell makes a vectoral contribution to the overall movement in its preferred direction with magnitude proportional to the cell's average firing rate [26].

Schmidt's proof of concept and Georgopoulos et al.'s BMI application to reaching tasks spawned a variety of studies implementing "out-of-the-box" signal processing modeling approaches. One of the most notable studies by Chapin et al. [6] showed that a recurrent neural network could be used to translate the neural activity of 21–46 neurons of rats trained to obtain water by pressing a lever with their paw. The usefulness of this BMI was demonstrated when the animals routinely stopped physically moving their limbs to obtain the water reward. Also in the neural network class, Lin et al. [27] used a self-organizing map (SOM) that clustered neurons with similar firing patters which then indicated movement directions for a spiral drawing task. Borrowing from control theory, Kalaska et al. [28] proposed the use for forward- and inverse-control architectures for reaching movements. Also during this period other researchers presented interpretations of population coding which included a probability-based population coding from Sanger [29] and muscle-based cellular tuning from Mussa-Ivaldi [30].

Almost 20 years after Schmidt's [1] and Georgopoulos et al.'s initial experiments, Wessberg and colleagues [10] presented the next major advancement in BMIs [2, 3] by demonstrating a real (nonportable) neuroprosthetic device in which the neuronal activity of a primate was used to control a robotic arm [10]. This research group hypothesized that the information needed for the BMI is distributed across several cortices and therefore neuronal activity was collected from 100 cells in *multiple* cortical areas (premotor, primary motor, and posterior parietal) while the primate performed a three-dimensional (3D) feeding (reaching) task. Linear and nonlinear signal processing techniques including a frequency-domain Wiener filter (WF) and a time delay neural network (TDNN) were used to estimate hand position. Trajectory estimates were then transferred via the Internet to a local robot and a robot located at another university.

In parallel with the work of Nicolelis [20], Serruya and colleagues [14] presented a contrasting view of BMIs by showing that a 2D computer cursor control task could be achieved using only a few neurons (7–30) located only in the primary motor cortex of a primate. The WF signal processing methodology was again implemented here; however this paradigm was *closed loop* since the primate received instant visual feedback from the cursor position output from the WF. The novelty of this experiment results from the primate's opportunity to incorporate the signal processing model into its motor processing.

The final BMI approach we briefly review is from Andersen's research group, which showed that the endpoint of hand reaching can be estimated using a Bayesian probabilistic method [31]. Neural recordings were taken from the parietal reach region (PRR) since they are believed to encode the planning and target of hand motor tasks. Using this hypothesis this research group devised a paradigm in which a primate was cued to move its hand to rectangular grid target locations presented on a computer screen. The neural-to-motor translation involves computing the likelihood of neural activity given a particular target. While this technique has been shown to accurately predict the endpoint of hand reaching, it differs for the aforementioned techniques by not accounting for the hand trajectory.

1.3 CHARACTERISTICS OF NEURAL RECORDINGS

One of the most important steps in implementing optimal signal processing technique for *any* application is data analysis. Here the reader should take note that optimality in the signal processing technique is predicated on the matching between the statistics of the data match and the a priori assumptions inherent in any signal processing technique [32]. In the case of BMIs, the statistical properties of the neural recordings and the analysis of neural ensemble data are not fully understood. Hence, this lack of information means that the neural–motor translation is not guaranteed to be the best possible, even if optimal signal processing is utilized (because the criterion for optimality may not match the data properties). Despite this reality, through the development of new neuronal data analysis techniques we can improve the match between neural recordings and BMI design [33, 34]. For this reason, it is important for the reader to be familiar with the characteristics of neural recordings that would be encountered.

The process of extracting signals from the motor, premotor, and parietal cortices of a behaving animal involves the implantation of subdural microwire electrode arrays into the brain tissue (usually layer V) [10]. At this point, the reader should be aware that current BMI studies involve the sampling of a minuscule fraction of motor cortex activity (tens to hundreds of cortical cells recorded from motor-related areas that are estimated to contain 100 million neurons) [35]. Each microwire measures the potentials (*action potentials*) resulting from ionic current exchanges across the membranes of neurons locally surrounding the electrode. Typical cellular potentials as shown in Figure 1.2*a* have magnitudes ranging from hundreds of microvolts to tens of millivolts and time durations of tens to a couple of milliseconds [34]. Since action potentials are so short in duration, it is common to treat them as point processes where the continuous voltage waveform is converted into a series of *time stamps* indicating the instance in time when the spike occurred. Using the time stamps, a series of pulses (*spikes*—zeros or ones) can be used to visualize the activity of each neuron; this time series shown in Figure 1.2*b* is referred to as a *spike train*. The spike trains of neural ensembles are sparse, nonstationary, and discontinuous. While the statistical properties of neural recordings can vary depending on the sample area, animal, and behavior paradigm, in general spike trains are assumed to have a Poisson distribution [33]. To reduce the sparsity in neuronal recordings, a method

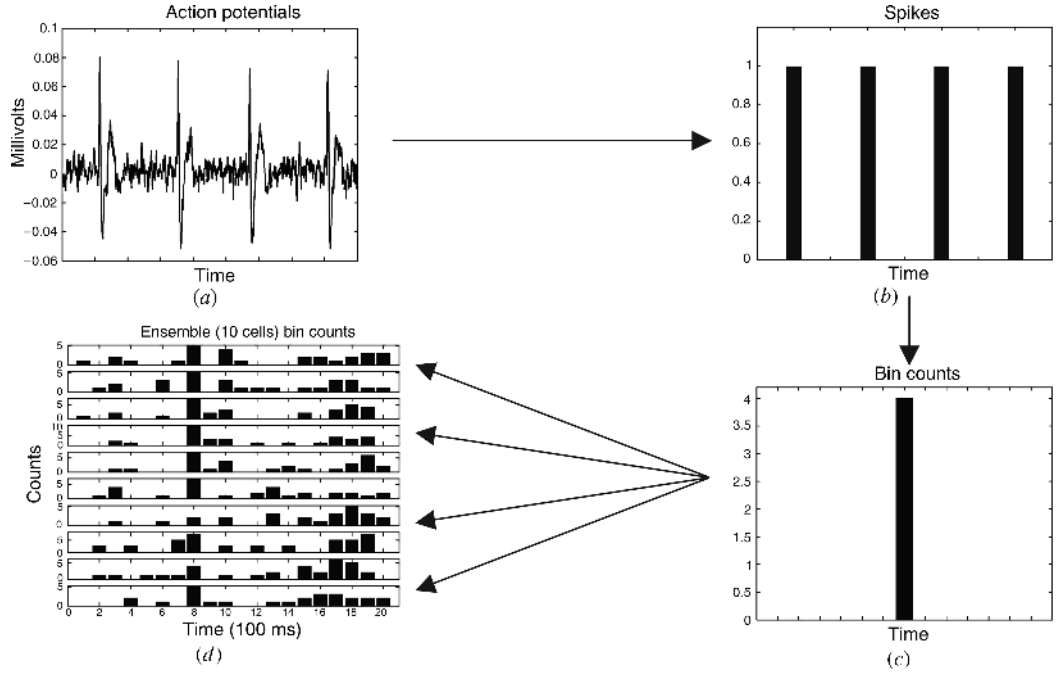

Figure 1.2 Spike-binning process: (*a*) cellular potentials; (*b*) a spike train; (*c*) bin count for single cell; (*d*) ensemble of bin counts.

of binning is used to count the number of spikes in 100-ms nonoverlapping windows as shown in Figure 1.2*c*. This method greatly reduces the number of zeros in the digitized time series and also provides a time-to-amplitude conversion of the firing events. Even with the binning procedure, the data remain extremely sparse. In order for the reader to assess the degree of sparsity and nonstationarity in BMI data, we present in Table 1.1 observations from a 25-min BMI experiment recorded at the Nicolelis's laboratory at Duke University. From the table we can see that the percentage of zeros can be as high as 80% indicating that the data are extremely sparse. Next we compute the firing rate for each cell in nonoverlapping 1-min windows and compute the average across all cells. The ensemble of cells used in this analysis primarily contains low firing rates given by the small ensemble average. Additionally we can see the time variability of the 1-min ensemble average given by the associated standard deviation.

In Figure 1.3, the average firing rate of the ensemble (computed in nonoverlapping 60-s windows) is tracked for a 38-min session. From minute to minute, the mean value in the firing rate can change drastically depending on the movement being performed. Ideally we would like our optimal signal processing techniques to capture the changes observed in Figure 1.3. However, the reader should be aware that any of the out-of-the-box signal processing techniques such as WFs and artificial neural networks assume stationary statistics

TABLE 1.1 Neuronal Activity for 25-min Recording Session

	Percentage of zeros	Average firing rate (spikes/cell/min)
3D reaching task (104 cells)	86	0.25 ± 0.03
2D cursor control task (185 cells)	60	0.69 ± 0.02

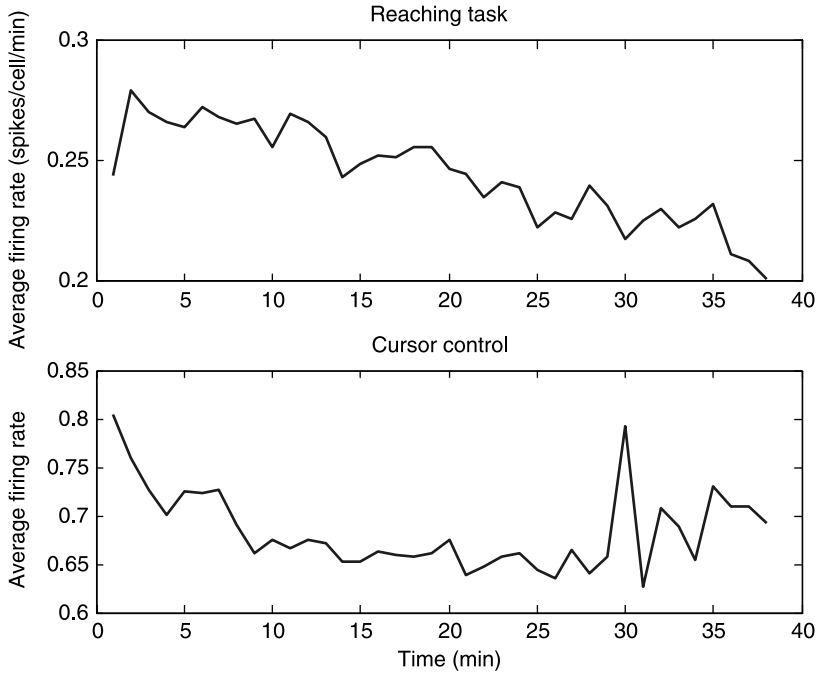

Figure 1.3 Time-varying statistics of neuronal recordings for two behaviors.

over time, which means that the derived neural-to-motor mapping will not be optimal unless the window size is very well controlled. More importantly, any performance evaluations and model interpretations drawn by the experimenter can be biased by the mismatch between data and model type.

1.4 MODELING PROBLEM

The models implemented in BMIs must learn to interpret neuronal activity and accurately translate it into commands for a robot that mimic the intended motor behavior. By analyzing recordings of neural activity collected simultaneously with behavior, the aim is to find a *functional* relationship between neural activity and the kinematic variables of position, velocity, acceleration, and force. An important question here is how to choose the class of functions and model topologies that best match the data while being sufficiently powerful to create a mapping from neuronal activity to a variety of behaviors. As a guide, prior knowledge about the nervous system can be used to help develop this relationship. Since the experimental paradigm involves modeling two related multidimensional time variables (neural firing and behavior), we are directed to a general class of input–output (I/O) models. Within this class there are several candidate models available, and based on the amount of neurophysiological information that is utilized about the system, an appropriate modeling approach can be chosen. Three types of I/O models based on the amount of prior knowledge exist in the literature [36]:

- "White Box" The model is perfectly known (except for the parameters) and based on physical insight and observations.
- "Gray Box" Some physical insight is available but the model is not totally specified and other parameters need to be determined from the data.

- "Black Box" Little or no physical insight is available or used to choose the model, so the chosen model is picked on other grounds (e.g., robustness, easy implementation).

The choice of white, gray, or black box is dependent upon our ability to access and measure signals at various levels of the motor system as well as the computational cost of implementing the model in our current computing hardware.

The first modeling approach, white box, would require the highest level of physiological detail. Starting with behavior and tracing back, the system comprises muscles, peripheral nerves, the spinal cord, and ultimately the brain. This is a daunting task of system modeling due to the complexity, interconnectivity, and dimensionality of the involved neural structures. Model implementation would require the parameterization of a complete motor system [37] that includes the cortex, cerebellum, basal ganglia, thalamus, corticospinal tracts, and motor units. Since all of the details of each component/subcomponent of the described motor system remain unknown and are the subject of study for many neurophysiological research groups around the world, it is not presently feasible to implement white-box BMIs. Even if it was possible to parameterize the system to some high level of detail, the task of implementing the system in our state-of-the-art computers and digital signal processors (DSPs) would be an extremely demanding task.

The gray-box model requires a reduced level of physical insight. In the gray-box approach, one could take a particularly important feature of the motor nervous system, incorporate this knowledge into the model, and then use data to determine the rest of the unknown parameters. Two examples of gray-box models can be found in the BMI literature. One of the most common examples is Georgopoulos's PVA [3]. Using observations that cortical neuronal firing rates were dependent on the direction of arm movement, a model was formulated to incorporate the weighted sum of the neuronal firing rates. The weights of the model are then determined from the neural and behavioral recordings. A second example is given by Todorov, who extended the PVA by observed multiple correlations of M1 firing with movement position, velocity, acceleration, force exerted on an object, visual target position, movement preparation, and joint configuration [7, 8]. With these observations, Todorov [42] proposed a minimal, linear model that relates the delayed firings in M1 to the sum of many mechanistic variables (position, velocity, acceleration, and force of the hand). Todorov's model is intrinsically a *generative model* [16, 43]. Using knowledge about the relationship between arm kinematics and neural activity, the *states* (preferably the feature space of Todorov) of linear or nonlinear *dynamical* systems can be assigned. This methodology is supported by a well-known training procedure developed by Kalman for the linear case [44] and has been recently extended to the nonlinear case under the graphical models or Bayesian network frameworks. Since the formulation of generative models is recursive in nature, it is believed that the model is well suited for learning about motor systems because the states are all intrinsically related in time.

The last I/O model presented is the black-box model for BMIs. In this case, it is assumed that no physical insight is available for the model. The foundations of this type of time-series modeling were laid by Norbert Wiener for applications of gun control during World War II [45]. While military gun control applications may not seem to have a natural connection to BMIs, Wiener provided the tools for building models that correlate *unspecified* time-series inputs (in our case neuronal firing rates) and outputs (hand/arm movements). While this WF is topographically similar to the PVA, it is interesting to note that it was developed more than 30 years before Georgopoulos was developing his linear model relating neuronal activity to arm movement direction.

The three I/O modeling abstractions have gained large support from the scientific community and are also a well-established methodology in control theory for system

identification [32]. Here we will concentrate on the last two types, which have been applied by engineers for many years to a wide variety of applications and have proven that the methods produce viable phenomenological descriptions when properly applied [46, 47]. One of the advantages of the techniques is that they quickly find, with relatively simple algorithms, optimal mappings (in the sense of minimum error power) between different time series using a nonparametric approach (i.e., without requiring a specific model for the time-series generation). These advantages have to be counterweighted by the abstract (nonstructural) level of the modeling and the many difficulties of the method, such as determining what a reasonable fit, a model order, and a topology are to appropriately represent the relationships among the input and desired response time series.

1.4.1 Gray Box

1.4.1.1 Population Vector Algorithm The first model discussed is the PVA, which assumes that a cell's firing rate is a function of the velocity vector associated with the movement performed by the individual. The PVA model is given by

$$s_n(\mathbf{V}) = b_0^n + b_x^n v_x + b_y^n v_y + b_z^n v_z = \mathbf{B} \cdot \mathbf{V} = |\mathbf{B}||\mathbf{V}| \cos\theta \tag{1.1}$$

where the firing rate s for neuron n is a weighted ($b_{x,y,z}^n$) sum of the vectoral components ($v_{x,y,z}$) of the unit velocity vector $\mathbf{V}$ of the hand plus the mean firing rate b_0^n. The relationship in (1.1) is the inner product between the velocity vector of the movement and the weight vector for each neuron. The inner product (i.e., spiking rate) of this relationship becomes maximum when the weight vector $\mathbf{B}$ is collinear with the velocity vector $\mathbf{V}$. At this point, the weight vector $\mathbf{B}$ can be thought of as the cell's preferred direction for firing since it indicates the direction for which the neuron's activity will be maximum. The weights b^n can be determined by multiple regression techniques [3]. Each neuron makes a vectoral contribution w in the direction of $\mathbf{P}_i$ with magnitude given in (1.2). The resulting population vector or movement is given by (1.3), where the reconstructed movement at time t is simply the sum of each neuron's preferred direction weighted by the firing rate:

$$w_n(\mathbf{V}, t) = s_n(\mathbf{V}) - b_0^n \tag{1.2}$$

$$\mathbf{P}(\mathbf{V}, t) = \sum_{n=1}^{N} w_n(\mathbf{V}, t) \frac{\mathbf{B}_n}{\|\mathbf{B}_n\|} \tag{1.3}$$

It should be noted that the PVA approach includes several assumptions whose appropriateness in the context of neural physiology and motor control will be considered here. First, each cell is considered independently in its contribution to the kinematic trajectory. The formulation does not consider feedback of the neuronal firing patterns—a feature found in real interconnected neural architectures. Second, neuronal firing counts are linearly combined to reproduce the trajectory. At this point, it remains unknown how the neural activation of nonlinear functions will be necessary for complex movement trajectories.

1.4.1.2 Todorov's Mechanistic Model An extension to the PVA has been proposed by Todorov [42], who considered multiple correlations of M1 firing with movement velocity and acceleration, position, force exerted on an object, visual target position, movement preparation, and joint configuration [2, 4, 10, 12–15, 38–41]. Todorov advanced the alternative hypothesis that the neural correlates with kinematic variables are epiphenomena of muscle activation stimulated by neural activation. Using studies showing that M1 contains multiple, overlapping representations of arm muscles and forms dense

corticospinal projections to the spinal cord and is involved with the triggering of motor programs and modulation of spinal reflexes [35], Todorov [42] proposed a minimal, linear model that relates the delayed firings in M1 to the *sum* of mechanistic variables (position, velocity, acceleration, and force of the hand). Todorov's model takes the form

$$\mathbf{U}\mathbf{s}(t-\Delta) = \mathbf{F}^{-1}\mathbf{G}\mathbf{F}(t) + m\mathbf{H}\mathbf{A}(t) + b\mathbf{H}\mathbf{V}(t) + k\mathbf{H}\mathbf{P}(t) \tag{1.4}$$

where the neural population vector $\mathbf{U}$ is scaled by the neural activity $\mathbf{s}(t)$ and is related to the scaled kinematic properties of gripping force $\mathbf{GF}(t)$, hand acceleration $\mathbf{HA}(t)$, velocity $\mathbf{HV}(t)$, and position $\mathbf{HP}(t)$.[1] From the BMI experimental setup, spatial samplings (in the hundred of neurons) of the input $\mathbf{s}(t)$ and the hand position, velocity, and acceleartion are collected synchronously; therefore the problem is one of finding the appropriate constants using a system identification framework [32]. Todorov's model in (1.4) assumes a first-order force production model and a local linear approximation to multijoint kinematics that may be too restrictive for BMIs. The mechanistic model for neural control of motor activity given in (1.4) involves a dynamical system where the output variables, position, velocity, acceleration, and force, of the motor system are driven by an high-dimensional input signal that is comprised of delayed ensemble neural activity [42]. In this interpretation of (1.4), the neural activity can be viewed as the *cause* of the changes in the mechanical variables and the system will be performing the decoding. In an alternative interpretation of Eq. (1.4), one can regard the neural activity as a distributed *representation* of the mechanical activity, and the system will be performing generative modeling. Next, a more general state space model implementation, the Kalman filter, will be presented. This filter corresponds to the *representation* interpretation of Todorov's model for neural control. Todorov's model clearly builds upon the PVA and can explain multiple neuronal and kinematic correlations; however, it is still a linear model of a potentially nonlinear system.

1.4.1.3 Kalman Filter A variety of Bayesian encoding/decoding approaches have been implemented in BMI applications [12, 16, 48]. In this framework, model designs have been developed based upon various assumptions that include the Kalman filter (linear, Gaussian model), extended Kalman filter (EKF, nonlinear, Gaussian model), and particle filter (PF, linear/nonlinear, Poisson model). Our discussion will begin with the most basic of the Bayesian approaches: the Kalman filter. This approach assumes a linear relationship between hand motion states and neural firing rates as well as Gaussian noise in the observed firing activity. The Kalman formulation attempts to estimate the state $\mathbf{x}(t)$ of a linear dynamical system as shown in Figure 1.4. For BMI applications, we define the states as the hand position, velocity, and acceleration, which are governed by a linear dynamical equation, as shown in

$$\mathbf{x}(t) = [\,\mathbf{H}\mathbf{P}(t) \quad \mathbf{H}\mathbf{V}(t) \quad \mathbf{H}\mathbf{A}(t)]^{\mathrm{T}} \tag{1.5}$$

where $\mathbf{HP}$, $\mathbf{HV}$, and $\mathbf{HA}$ are the hand position, velocity, and acceleration vectors,[2] respectively. The Kalman formulation consists of a generative model for the data specified by the linear dynamic equation for the state in

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{u}(t) \tag{1.6}$$

[1]The mechanistic model reduces to the PVA if the force, acceleration, and position terms are removed and neurons are independently considered.

[2]The state vector is of dimension $9 + N$; each kinematic variable contains an x, y, and z component plus the dimensionality of the neural ensemble.

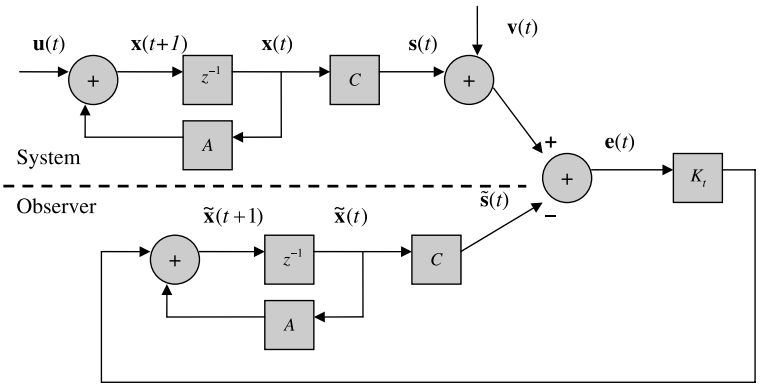

Figure 1.4 Kalman filter block diagram.

where $\mathbf{u}(t)$ is assumed to be a zero-mean Gaussian noise term with covariance $\mathbf{U}$. The output mapping (from state to spike trains) for this BMI linear system is simply

$$\mathbf{s}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{v}(t) \tag{1.7}$$

where $\mathbf{v}(t)$ is the zero-mean Gaussian measurement noise with covariance $\mathbf{V}$ and $\mathbf{s}$ is a vector consisting of the neuron firing patterns binned in nonoverlapping windows. In this specific formulation, the output-mapping matrix $\mathbf{C}$ has dimensions $N \times 9$. Alternatively, we could have also included the spike counts of N neurons in the state vector as $\mathbf{f}_1, \ldots, \mathbf{f}_N$. This specific formulation would exploit the fact that the future hand position is a function of not only the current hand position, velocity, and acceleration but also the current cortical firing patterns. However, this advantage comes at the cost of large training set requirements, since this extended model would contain many more parameters to be optimized. To train the topology given in Figure 1.4, L training samples of $\mathbf{x}(t)$ and $\mathbf{s}(t)$ are utilized, and the model parameters $\mathbf{A}$ and $\mathbf{U}$ as given in (1.6) are determined using least squares. The optimization problem to be solved is given by

$$\mathbf{A} = \arg\min_{\mathbf{A}} \sum_{t=1}^{L-1} \|\mathbf{x}(t+1) - \mathbf{A}\mathbf{x}(t)\|^2 \tag{1.8}$$

The solution to this optimization problem is found to be

$$\mathbf{A} = \mathbf{X}_1\mathbf{X}_0^{\mathrm{T}}(\mathbf{X}_1\mathbf{X}_1^{\mathrm{T}})^{-1} \tag{1.9}$$

where the matrices are defined as $\mathbf{X}_0 = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_{L-1}]$, $\mathbf{X}_1 = [\mathbf{x}_2 \quad \cdots \quad \mathbf{x}_L]$. The estimate of the covariance matrix $\mathbf{U}$ can then be obtained using

$$\mathbf{U} = \frac{(\mathbf{X}_1 - \mathbf{A}\mathbf{X}_0)(\mathbf{X}_1 - \mathbf{A}\mathbf{X}_0)^{\mathrm{T}}}{L-1} \tag{1.10}$$

Once the system parameters are determined using least squares on the training data, the model obtained ($\mathbf{A}$, $\mathbf{C}$, $\mathbf{U}$) can be used in the Kalman filter to generate estimates of hand positions from neuronal firing measurements. Essentially, the model proposed here assumes a linear dynamical relationship between current and future trajectory states. Since the Kalman filter formulation requires a reference output from the model, the spike counts are assigned to the output, as they are the only available signals.

The Kalman filter is an adaptive state estimator (observer) where the observer gains are optimized to minimize the state estimation error variance. In real-time

operation, the Kalman gain matrix K (1.12) is updated using the projection of the error covariance in (1.11) and the error covariance update in (1.14). During model testing, the Kalman gain correction is a powerful method for decreasing estimation error. The state in (1.13) is updated by adjusting the current state value by the error multiplied with the Kalman gain:

$$\mathbf{P}^{-}(t+1) = \mathbf{A}\mathbf{P}(t)\mathbf{A}^{\mathrm{T}} + \mathbf{U} \tag{1.11}$$

$$\mathbf{K}(t+1) = \mathbf{P}^{-}(t+1)\mathbf{C}^{\mathrm{T}}(\mathbf{C}\mathbf{P}^{-}(t+1)\mathbf{C}^{\mathrm{T}})^{-1} \tag{1.12}$$

$$\tilde{\mathbf{x}}(t+1) = \mathbf{A}\tilde{\mathbf{x}}(t) + \mathbf{K}(t+1)(\mathbf{S}(t+1) - \mathbf{C}\mathbf{A}\tilde{\mathbf{x}}(t)) \tag{1.13}$$

$$\mathbf{P}(t+1) = (\mathbf{I} - \mathbf{K}(t+1)\mathbf{C})\mathbf{P}^{-}(t+1) \tag{1.14}$$

While the Kalman filter equations provide a closed-form decoding procedure for linear Gaussian models, we have to consider the fact that the relationship between neuronal activity and behavior may be nonlinear. Moreover, measured neuronal firing often follows Poisson distributions. The consequences for such a mismatch between the model and the real system will be expressed as additional errors in the final position estimates. To cope with this problem, we need to go beyond the linear Gaussian model assumption. In principle, for an arbitrary nonlinear dynamical system with arbitrary known noise distributions, the internal states (**HP**, **HV**, and **HA**) can be estimated from the measured outputs (neuronal activity). Algorithms designed to this end include the EKF, the unscented Kalman filter (UKF), the PF, and their variants. All of these algorithms basically try the complicated recursive Bayesian state estimation problem using various simplifications and statistical techniques. In the literature, BMI researchers have already implemented the PF approach [16].

In this most general framework, the state and output equations can include nonlinear functions $f_1(\cdot)$ and $f_2(\cdot)$ as given in

$$\mathbf{x}(t+1) = f_1(\mathbf{x}(t)) + \mathbf{u}(t) \qquad \mathbf{s}(t) = f_2(\mathbf{x}(t)) + \mathbf{v}(t) \tag{1.15}$$

Experimental observations have shown that the measured spike trains typically follow a Poisson distribution, which is given in

$$p(\mathbf{s}(t) \mid \mathbf{x}(t)) = \prod_{n=1}^{N} \frac{e^{-\lambda_i(\mathbf{x}(t))}[\lambda_i(\mathbf{x}(t))]^{y_i(t)}}{[s_i(t)]!} \tag{1.16}$$

The tuning function of the ith neuron is denoted by λ_i and $s_i(t)$ is the firing count of the ith neuron at time instant t. In order to decode neuronal activity into hand kinematics for the nonlinear observation equations and Poission spiking models, the recursive Bayesian estimator called the PF can be used. In this approach, we seek to recursively update the posterior probability of the state vector given the neuronal measurements as shown in (1.17), where $\mathbf{S}_t$ is the entire history of neuronal firing up to time t:

$$p(\mathbf{x}(t)|\mathbf{S}_t) = \mu p(\mathbf{s}(t)|x(t))p(\mathbf{x}(t)|\mathbf{S}_{t-1}) \tag{1.17}$$

After some manipulations, the equivalent expression in (1.18) can be obtained [49]:

$$p(\mathbf{x}(t)|\mathbf{S}_t) = \mu p(\mathbf{s}(t)|\mathbf{x}(t)) \int p(\mathbf{x}(t)|\mathbf{x}(t-1))p(\mathbf{x}(t-1)|\mathbf{S}_{t-1})\, d\mathbf{x}(t-1) \tag{1.18}$$

Notice that (1.18) is essentially the recursion of the conditional state distribution that we seek from $p(\mathbf{x}(t-1)|\mathbf{S}_{t-1})$ to $p(\mathbf{x}(t)|\mathbf{S}_t)$. Analytical evaluation of the integral on the right-hand side is, in general, impossible. Therefore, the following simplifying assumption is made: The conditional state distribution can be approximated by a delta train, which

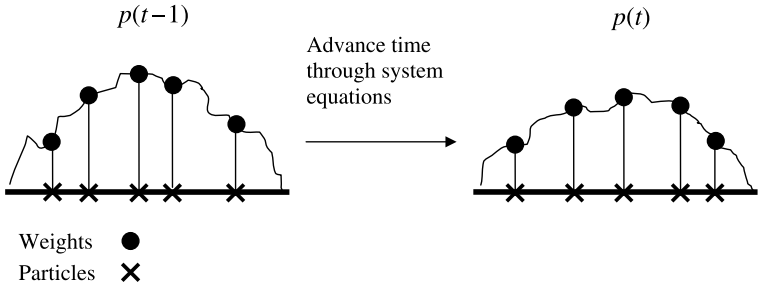

Figure 1.5 Estimation of continuous distributions by sampling (particles).

samples the continuous distribution at appropriately selected values called particles. Each particle also has a weight associated with it that represents the probability of that particle, as shown in Figure 1.5. The elegance of the PF approach lies in the simplification of the integral in (1.18) to a summation. These simplifications lead to the following update equations for the particles, their weights, and the state estimate:

$$x_i(t+1) = f_1(x_i(t)) \tag{1.19}$$

$$\tilde{w}_i(t+1) = P(s(t)|x_i(t))w_i(t) \tag{1.20}$$

$$w_i(t+1) = \frac{\tilde{w}_i(t+1)}{\sum_i \tilde{w}_i(t+1)} \tag{1.21}$$

$$\tilde{\mathbf{x}}(t+1) = \sum_{i=1}^{N} w_i(t+1)x_i(t+1) \tag{1.22}$$

We have shown that, using measured position, velocity, and acceleration as states and neuronal firing counts as model outputs within this recursive, probabilistic framework, this approach may seem to be the best state-of-the-art method available to understand the encoding and decoding between neural activity and hand kinematics. Unfortuntely, for BMIs this particular formulation is faced with problems of parameter estimation. The generative model is required to find the mapping from the low-dimensional kinematic parameter state space to the high-dimensional output space of neuronal firing patterns (100+ dimensions). Estimating model parameters from the collapsed space to the high-dimensional neural state can be difficult and yield multiple solutions. Moreover, in the BMI literature the use of the PF has only produced marginal improvements over the standard Kalman formulation, which may not justify the extra computational complexity [16]. For this modeling approach, our use of physiological knowledge and choice of modeling framework actually complicates the mapping process. As an alternative, one could disregard any knowledge about the system being modeled and use a strictly data-driven methodology to build the model.

1.4.2 Black Box

1.4.2.1 Wiener Filters The first black-box model we will discuss assumes that there exists a linear mapping between the desired hand kinematics and neuronal firing counts. In this model, the delayed versions of the firing counts, $\mathbf{s}(t-l)$, are the bases that construct the output signal. Figure 1.6 shows the topology of the multiple input–multiple output (MIMO) WF where the output y_j is a weighted linear combination of the l most recent

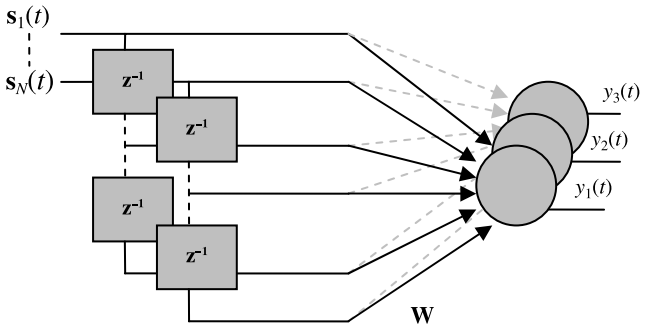

Figure 1.6 FIR filter topology. Each neuronal input $\mathbf{s}_N$ contains a tap delay line with l taps.

values[3] of neuronal inputs $\mathbf{s}$ given in (1.23) [32]. Here y_j can be defined to be any of the single coordinate directions of the kinematic variables **HP**, **HV**, **HA**, or **GF**. The model parameters are updated using the optimal linear least-squares (LS) solution that matches the Wiener solution. The Wiener solution is given by (1.24), where $\mathbf{R}$ and $\mathbf{P}_j$ are the autocorrelation and cross-correlation functions, respectively, and $\mathbf{d}_j$ is hand trajectory, velocity, or gripping force:[4]

$$y_j(t) = \mathbf{W}_j \mathbf{s}(t) \tag{1.23}$$

$$\mathbf{W}_j = \mathbf{R}^{-1}\mathbf{P}_j = E(\mathbf{s}^{\mathrm{T}}\mathbf{s})^{-1}E(\mathbf{s}^{\mathrm{T}}\mathbf{d}_j) \tag{1.24}$$

The autocorrelation matrix $\mathbf{R}$ and the cross-correlation matrix $\mathbf{P}$ can be estimated directly from the data using either the autocorrelation or the covariance method [32]. Experimentally we verified that the size of the data block should contain at least 10 min of recordings for better performance. In this MIMO problem, the autocorrelation matrix is not Toeplitz, even when the autocorrelation method is employed. One of the real dangers of computing the WF solution to BMIs is that $\mathbf{R}$ may not be full rank [50]. Instead of using the Moore–Penrose inverse, we utilize a regularized solution substituting $\mathbf{R}^{-1}$ by $(\mathbf{R} + \lambda\mathbf{I})^{-1}$, where λ is the regularization constant estimated from a cross-validation set. Effectively this solution corresponds to ridge regression [51]. The computational complexity of the WF is high for the number of input channels used in our experiments. For 100 neural channels using 10 tap delays and three outputs, the total number of weights is 3000. This means that one must invert a 1000×1000 matrix every N samples, where N is the size of the training data block.

As is well known in adaptive filter theory [32], search procedures can be used to find the optimal solution using gradient descent or Newton-type search algorithms. The most widely used algorithm in this setting is the least mean-square (LMS) algorithm, which utilizes stochastic gradient descent [32]. For real data we recommend the normalized LMS algorithm instead,

$$\mathbf{W}_j(t+1) = \mathbf{W}_j(t) + \frac{\eta}{\|s(t)\|}\mathbf{e}_j(t)\mathbf{s}(t) \tag{1.25}$$

where η is the step size or learning rate, $e(t) = d_x(t) - y_x(t)$ is the error, and $\|\mathbf{s}(t)\|$ is the power of the input signal contained in the taps of the filter. Using the normalized LMS

[3]In our studies we have observed neuronal activity correlated with behavior for up to 10 lags.

[4]Each neuronal input and desired trajectory for the WF was preprocessed to have a mean value of zero.

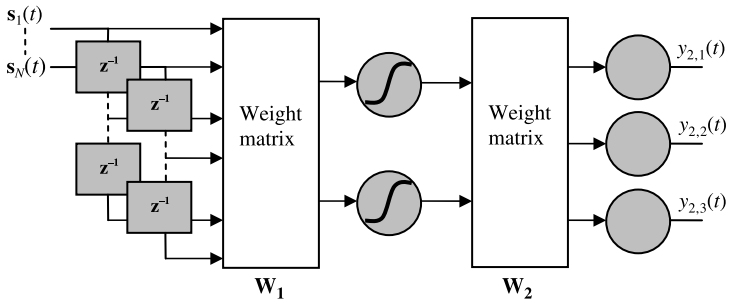

Figure 1.7 Time delay neural network topology.

algorithm (NLMS), the model parameters are updated incrementally at every new sample and so the computation is greatly reduced. The step size must be experimentally determined from the data. One may think that the issues of nonstationarity are largely resolved since the filter is always being updated, tracking the changing statistics. However, during testing the desired response is not available so the weights of the filter have to be frozen after training. Therefore, NLMS is still subject to the same problems as the Wiener solution, although it may provide slightly better results when properly trained (when the data are not stationary, the LMS and the Wiener solution do not necessarily coincide [32]).

Linear filters trained with mean-square error (MSE) provide the best linear estimate of the mapping between neural firing patterns and hand position. Even though the solution is guaranteed to converge to the global optimum, the model assumes that the relationship between neural activity and hand position is linear, which may not be the case. Furthermore, for large input spaces, including memory in the input introduces many extra degrees of freedom to the model, hindering generalization capabilities.

1.4.2.2 Time Delay Neural Network Spatiotemporal nonlinear mappings of neuronal firing patterns to hand position can be constructed using a TDNN [52]. The TDNN architecture consists of a tap delay line memory structure at the input in which past neuronal firing patterns in time can be stored, followed by a one-hidden-layer perceptron with a linear output as shown in Figure 1.7. The output of the first hidden layer of the network can be described with the relation $\mathbf{y}_1(t) = f(\mathbf{W}_1\mathbf{s}(t))$, where $f(\cdot)$ is the hyperbolic tangent nonlinearity [$\tanh(\beta x)$].[5] The input vector $\mathbf{s}$ includes l most recent spike counts from N input neurons. In this model the nonlinear weighted and delayed versions of the firing counts $\mathbf{s}(t-l)$ construct the output of the *hidden layer*. The number of delays in the topology should be set so that there is significant coupling between the input and desired signal. The number of hidden processing elements (PEs) is determined through experimentation. The output layer of the network produces the hand trajectory $\mathbf{y}_2(t)$ using a linear combination of the hidden states and is given by $\mathbf{y}_2(t) = \mathbf{W}_2\mathbf{y}_1(t)$. The weights ($\mathbf{W}_1$, $\mathbf{W}_2$) of this network can be trained using static backpropagation[6] with the MSE as the learning criterion. This is the great advantage of this artificial neural network.

This topology is more powerful than the linear finite impulse response (FIR) filter because it is effectively a nonlinear combination of FIR filters (as many as the number of hidden PEs). Each of the hidden PE outputs can be thought of as a basis function of the output space (nonlinearly adapted from the input) utilized to project the

[5]The logistic function is another common nonlinearity used in neural networks.

[6]Backpropagation is a simple application of the chain rule, which propagates the gradients through the topology.

high-dimensional data. While the nonlinear nature of the TDNN may seem as an attractive choice for BMIs, putting memory at the input of this topology presents difficulties in training and model generalization. Adding memory to the high-dimensional neural input introduces many free parameters to train. For example, if a neural ensemble contains 100 neurons with 10 delays of memory and the TDNN topology contains five hidden PEs, 5000 free parameters are introduced in the input layer alone. Large data sets and slow learning rates are required to avoid overfitting. Untrained weights can also add variance to the testing performance, thus decreasing accuracy.

1.4.2.3 Nonlinear Mixture of Competitive Local Linear Models

The next model topology that we will discuss is in general similar to the TDNN; however, the training procedure undertaken here is significantly different. This modeling method uses the divide-and-conquer approach. Our reasoning is that a complex nonlinear modeling task can be elucidated by dividing it into simpler linear modeling tasks and combining them properly [53]. Previously, this approach was successfully applied to nonstationary signal segmentation, assuming that a nonstationary signal is a combination of piecewise stationary signals [54]. Hypothesizing that the neural activity will demonstrate varying characteristics for different localities in the space of the hand trajectories, we expect the multiple-model approach, in which each linear model specializes in a local region, to provide a better overall I/O mapping. However, here the problem is different since the goal is not to segment a signal but to segment the joint input/desired signal space. The overall system archtitecture is depicted in Figure 1.8.

The local linear models can be conceived as a committee of experts each specializing in one segment of the hand trajectory space. The multilayer perceptron (MLP) is introduced to the system in order to nonlinearly combine the predictions generated by all the linear models. Experiments demonstrated that this nonlinear mixture of competitive local linear models is potentially more powerful than a gated mixture of linear experts, where only the winning expert's opinion or their weighted sum is considered. In addition,

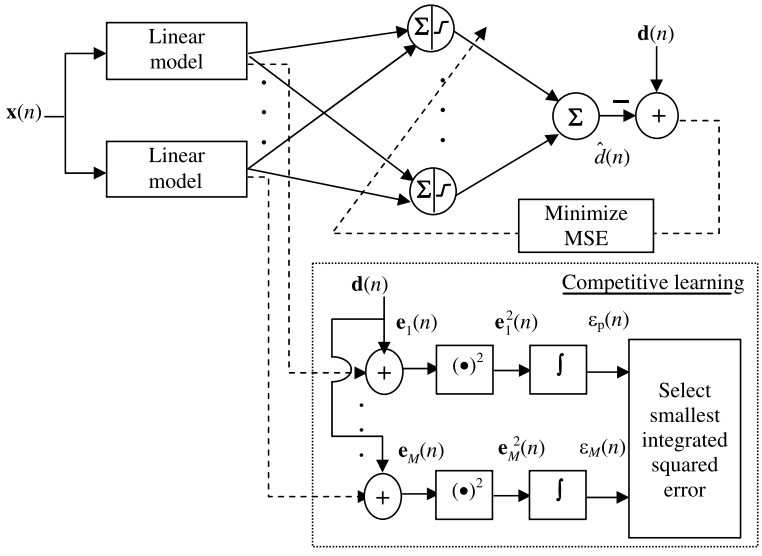

Figure 1.8 This topology consists of selecting a winner using integrated squared errors from each linear model. Outputs from M trained linear models are then fed to a MLP.

for BMI applications, it has the advantage that it does not require a selection scheme in testing, which can only be accurately done using the desired output. For example, in a prosthetic arm application, the desired hand position is not available in practice.

The topology allows a two-stage training procedure that can be performed independently: first competitive learning for the local linear models and then error back-propagation learning for the MLP. In off-line training, this can be done sequentially, where first the local linear models are optimized and then their outputs are used as training inputs for the following MLP. It is important to note that in this scheme both the local linear models and the MLP are trained to approximate the same desired response, which is the hand trajectory of the primate.

The training of the multiple linear models is accomplished by competitively (hard or soft) updating their weights in accordance with previous approaches using the normalized least-mean-square (NLMS) algorithm [32]. The winning model is determined by comparing the (leaky) integrated squared errors of all competing models and selecting the model that exhibits the least integrated error for the corresponding input [54]. In competitive training, the leaky integrated squared error for the ith model is given by

$$\varepsilon_i(t) = (1 - \mu)\varepsilon_i(t-1) + \mu e_i^2(t) \qquad i = 1, \ldots, M \tag{1.26}$$

where M is the number of models and μ is the time constant of the leaky integrator. If hard competition is employed, then only the weight vector of the winning model is updated. Specifically, the hard-competition update rule for the weight vector of the winning model is

$$\mathbf{w}_{\text{winner}}(t+1) = \mathbf{w}_{\text{winner}}(t) + \frac{\eta e_{\text{winner}}(t)\mathbf{s}(t)}{\gamma + \|\mathbf{s}(t)\|^2} \tag{1.27}$$

where $\mathbf{w}_{\text{winner}}$ is the weight vector, $\mathbf{s}(t)$ is the current input, $e_{\text{winner}}(t)$ is the instantaneous error of the winning model, η is the learning rate, and γ is the small positive constant. If soft competition is used, a Gaussian weighting function centered at the winning model is applied to all competing models. Every model is then updated proportional to the weight assigned to that model by this Gaussian weighting function:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \frac{\eta(t)\Lambda_{i,j}(t)e_t(t)\mathbf{x}(t)}{\gamma + \|\mathbf{x}(t)\|^2} \qquad i = 1, \ldots, M \tag{1.28}$$

where $\mathbf{w}_i$ is the weight vector of the ith model, the jth model is the winner, and $\Lambda_{i,j}(t)$ is the weighting function

$$\Lambda_{i,j}(t) = \exp\left(-\frac{d_{i,j}^2}{2\sigma^2(t)}\right) \tag{1.29}$$

where $d_{i,j}$ is the Euclidean distance between index i and j which is equal to $|j - i|$, $\eta(t)$ is the annealed learning rate, and $\sigma^2(t)$ is the kernel width, which decreases exponentially as t increases. The learning rate also decreases exponentially with time.

Soft competition preserves the topology of the input space by updating the models neighboring the winner; thus it is expected to result in smoother transitions between models specializing in topologically neighboring regions (of the state space). However, in the experimental results, it was shown that the hard-competition rule comparisons on data sets utilized in BMI experiments did not show any significant difference in generalization performance (possibly due to the nature of the data set used in these experiments).

The competitive training of the first layer of linear models to match the hand trajectory using the neural activity creates a set of basis signals from which the following

single hidden-layer MLP can generate accurate hand position predictions. However, this performance is at the price of an increase in the number of free model parameters which results from each of the local linear models (100 neurons × 10 tap delays × 3 coordinates × 10 models = 30,000 parameters in the input). Additional experimentation is necessary to evaluate the long-term performance (generalization) of such a model in BMI applications.

1.4.2.4 Recurrent Multilayer Perceptron The final black-box BMI model discussed is potentially the most powerful because it not only contains a nonlinearity but also includes dynamics through the use of feedback. The recurrent multilayer perceptron (RMLP) architecture (Fig. 1.9) consists of an input layer with N neuronal input channels, a fully connected hidden layer of PEs (in this case tanh), and an output layer of linear PEs. Each hidden layer PE is connected to every other hidden PE using a unit time delay. In the input layer equation (1.30), the state produced at the output of the first hidden layer is a nonlinear function of a weighted combination (including a bias) of the current input and the previous state. The feedback of the state allows for continuous representations on multiple time scales and effectively implements a short-term memory mechanism. Here, $f(\cdot)$ is a sigmoid nonlinearity (in this case *tanh*), and the weight matrices $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{W}_f$ as well as the bias vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ are again trained using synchronized neural activity and hand position data. Each hidden PE output can be thought of as a nonlinear adaptive basis of the output space utilized to project the high-dimensional data. These projections are then linearly combined to form the outputs of the RMLP that will predict the desired hand movements. One of the disadvantages of the RMLP when compared with the Kalman filter is that there is no known closed-form solution to estimate the matrices $\mathbf{W}_f$, $\mathbf{W}_1$, and $\mathbf{W}_2$ in the model; therefore, gradient descent learning is used. The RMLP can be trained with backpropagation through time (BPTT) or real-time recurrent learning (RTRL) [55]:

$$\mathbf{y}_1(t) = f(\mathbf{W}_1\mathbf{s}(t) + \mathbf{W}_f\mathbf{y}_1(t-1) + \mathbf{b}_1) \tag{1.30}$$

$$\mathbf{y}_2(t) = \mathbf{W}_2\mathbf{y}_1(t) + \mathbf{b}_2 \tag{1.31}$$

If the RMLP approach for the BMI is contrasted with Todorov's model, one can see that the RMLP accepts the neuronal activity (also as binned spike counts) as input and generates a prediction of the hand position using first-order internal dynamics. Although the model output $\mathbf{y}_2$ can consist of only the hand position, the RMLP must learn to build an efficient internal dynamical representation of the other mechanical variables (velocity,

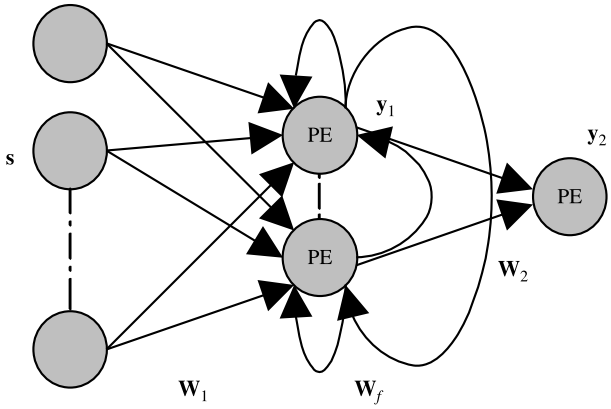

Figure 1.9 Fully connected, state recurrent neural network.

acceleration, and force) through the use of feedback. In fact, in this model, the hidden state vector ($\mathbf{y}_1$) can be regarded as the RMLP representation of these mechanical variables driven by the neural activity in the input ($\mathbf{s}$). Hence, the dynamical nature of Todorov's model is implemented through the nonlinear feedback in the RMLP. The output layer is responsible for extracting the position information from the representation in $\mathbf{y}_1$ using a linear combination. An interesting analogy exists between the output layer weight matrix $\mathbf{W}_2$ in the RMLP and the matrix $\mathbf{U}$ in Todorov's model. This analogy stems from the fact that each column of $\mathbf{U}$ represents a direction in the space spanning the mixture of mechanical variables to which the corresponding individual neuron is cosine tuned, which is a natural consequence of the inner product. Similarly, each column of $\mathbf{W}_2$ represents a direction in the space of hand position to which a nonlinear mixture of neuronal activity is tuned. In general, the combination of Todorov's theory with the use of nonlinearity and dynamics gives the RMLP a powerful approximating capability.

1.4.3 Generalization/Regularization/Weight Decay/ Cross Validation

The primary goal in BMI experiments is to produce the best estimates of HP, HV, and GF from neuronal activity that has not been used to train the model. This testing performance describes the generalization ability of the models. To achieve good generalization for a given problem, the two first considerations to be addressed are the choice of model topology and training algorithm. These choices are especially important in the design of BMIs because performance is dependent upon how well the model deals with the large dimensionality of the input as well as how the model generalizes in nonstationary environments. The generalization of the model can be explained in terms of the bias-variance dilemma of machine learning [56], which is related to the number of free parameters of a model. The MIMO structure of BMIs built for the data presented here can have as few as several hundred to as many as several thousand free parameters. On one extreme if the model does not contain enough parameters, there are too few degrees of freedom to fit the function to be estimated, which results in bias errors. On the other extreme, models with too many degrees of freedom tend to overfit the function to be estimated. In terms of BMIs, models tend to err on the latter because of the large dimensionality of the input. The BMI model overfitting is especially a problem in topologies where memory is implemented at the input layer. With each new delay element, the number of free parameters will scale with the number of input neurons as in the FIR filter and TDNN.

To handle the bias-variance dilemma, one could use the traditional Akaike or BIC criteria; however, the MIMO structure of BMIs excludes these approaches [57]. As a second option, during model training regularization techniques could be implemented that attempt to reduce the value of unimportant weights to zero and effectively prune the size of the model topology [58].

In BMI experiments we are not only faced with regularization issues but also we must consider ill-conditioned model solutions that result from the use of finite data sets. For example, computation of the optimal solution for the linear WF involves inverting a poorly conditioned input correlation matrix that results from sparse neural firing data that are highly variable. One method of dealing with this problem is to use the pseudoinverse. However, since we are interested in both conditioning and regularization, we chose to use ridge regression (RR) [51] where an identity matrix is multiplied by a white-noise variance and is added to the correlation matrix. The criterion function of RR is given by

$$J(\mathbf{w}) = E[\|\mathbf{e}\|^2] + \delta\|\mathbf{w}\|^2 \tag{1.32}$$

where $\mathbf{w}$ are the weights, $\mathbf{e}$ is the model error, and the additional term $\delta\|\mathbf{w}\|^2$ smooths the cost function. The choice of the amount of regularization (δ) plays an important role in the generalization performance and for larger deltas performance can suffer because SNR is sacrificed for smaller condition numbers. It has been proposed by Larsen et al. that δ can be optimized by minimizing the generalization error with respect to δ [47]. For other model topologies such as the TDNN, RMLP, and the LMS update for the FIR, weight decay (WD) regularization is an on-line method of RR to minimize the criterion function in (1.32) using the stochastic gradient, updating the weights by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\nabla(J) - \delta\mathbf{w}(n) \tag{1.33}$$

Both RR and WD can be viewed as the implementations of a Bayesian approach to complexity control in supervised learning using a zero-mean Gaussian prior [59].

A second method that can be used to maximize the generalization of a BMI model is called cross validation. Developments in learning theory have shown that during model training there is a point of maximum generalization after which model performance on unseen data will begin to deteriorate [60]. After this point the model is said to be *overtrained*. To circumvent this problem, a cross-validation set can be used to indicate an early stopping point in the training procedure. To implement this method, the training data are divided into a training set and a cross-validation set. Periodically during model training, the cross-validation set is used to test the performance of the model. When the error in the validation set begins to increase, the training should be stopped.

1.5 EXAMPLES

Four examples of how optimal signal processing can be used on real behavioral and neural recordings for the development of BMIs will now be given. The examples are focused on comparing the performance of linear, generative, nonlinear, feedforward, and dynamical models for the hand-reaching motor task. The four models include the FIR Wiener filter, TDNN (the nonlinear extension to the WF), Kalman filter, and RMLP. Since each of these models employs very different principles and has different mapping power, it is expected that they will perform differently and the extent to which they differ will be quantitatively compared. With these topologies, it will be shown how BMI performance is affected by the number of free model parameters, computational complexity, and nonlinear dynamics. In the following examples, the firing times of single neurons were recorded by researchers at Duke University while a primate performed a 3D reaching task that involved a right-handed reach to food and subsequent placing of the food in the mouth, as shown in Figure 1.10 [10]. Neuronal firings, binned (added) in nonoverlapping windows of 100 ms, were directly used as inputs to the models. The primate's hand position (HP), used as the desired signal, was also recorded (with a time-shared clock) and digitized with a 200-Hz sampling rate. Each model was trained using 20,010 consecutive time bins (2001 s) of data.

One of the most difficult aspects of modeling for BMIs is the dimensionality of the neuronal input (in this case 104 cells). Because of this large dimensionality, even the simplest models contain topologies with thousands of free parameters. Moreover, the BMI model is often trying to approximate relatively simple trajectories resembling sine waves which practically can be approximated with only two free parameters. Immediately, we are faced with avoiding overfitting the data. Large dimensionality also has an impact on the computational complexity of the model, which can require thousands more multiplications, divisions, and function evaluations. This is especially a problem if we wish to

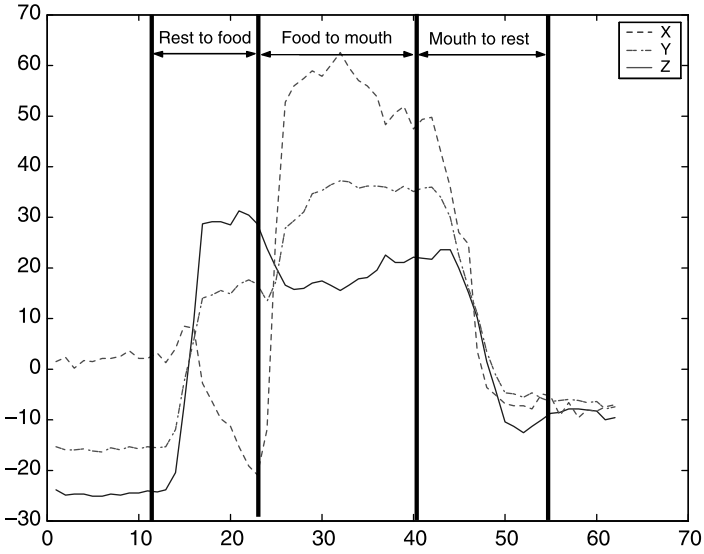

Figure 1.10 Reaching movement trajectory.

implement the model in low-power portable DSPs. Here we will assess each of the four BMI models in terms of their number of free parameters and computational complexity.

Model overfitting is often described in terms of prediction risk (PR), which is the expected performance of a topology when predicting new trajectories not encountered during training [61]. Several estimates of the PR for linear models have been proposed in the literature [62–65]. A simple way to develop a formulation for the prediction risk is to assume the quadratic form in (1.34), where e is the training error for a model with Φ parameters and N training samples. In this quadratic formulation, we can consider an optimal number of parameters, Φ^{Opt}, that minimizes the PR. We wish to estimate how the PR will vary with Φ, which can be given by a simple Taylor series expansion of (1.34) around Φ^{Opt} as performed in [65]. Manipulation of the Taylor expansion will yield the general form in (1.35). Other formulations for the PR include the generalized cross validation (GCV) and Akaike's final prediction error (FPE) given in (1.36) and (1.37). The important characteristic of (1.35)–(1.37) is that they all involve the interplay of the number of model parameters to the number of training samples. In general, the PR increases as the number of model parameters increases:

$$\text{PR} = E[e^2(\Phi_N)] \tag{1.34}$$

$$\text{PR} \approx e^2\left(1 + \frac{\Phi}{N}\right) \tag{1.35}$$

$$\text{GCV} = \frac{e^2}{(1 - \Phi/N)^2} \tag{1.36}$$

$$\text{FPE} = e^2\left(\frac{1 + \Phi/N}{1 - \Phi/N}\right) \tag{1.37}$$

The formulations for the PR presented here have been extended to nonlinear models [66]. While the estimation of the PR for linear models is rather straightforward, in the case nonlinear models the formulation is complicated by two factors. First, the nonlinear

formulation involves computing the effective number of parameters (a number that differs from the true number of parameter in the model), which is nontrivial to estimate since it depends on the amount of model bias, model nonlinearity, and amount of regularization used in training [66]. Second, the formulation involves computing the noise covariance matrix of the desired signal, another parameter that is nontrivial to compute, especially in the context of BMI hand trajectories.

For the reaching task data set, all of the models utilize 104 neuronal inputs, as shown in Table 1.2. The first encounter with an explosion in the number of free parameters occurs for both the WF and TDNN since they contain a 10-tap delay line at the input. Immediately the number of inputs is multiplied by 10. The TDNN topology has the greatest number of free parameters, 5215, of the feedforward topologies because the neuronal tap delay memory structure is also multiplied by the 5 hidden processing elements following the input. The WF, which does not contain any hidden processing elements, contains 3120 free parameters. In the case of the Kalman filter, which is the largest topology, the number of parameters explodes due to the size of the **A** and **C** matrices since they both contain the square of the dimensionality of the 104 neuronal inputs. Finally, the RMLP topology is the most frugal since it moves its memory structure to the hidden layer through the use of feedback, yielding a total of 560 free parameters.

To quantify how the number of free parameters affects model training time, a Pentium 4 class computer with 512 MB DDR RAM, the software package NeuroSolutions for the neural networks [67], and MATLAB for computing the Kalman and Wiener solution were used to train the models. The training times of all four topologies are given in Table 1.2. For the WF, the computation of the inverse of a 1040×1040 autocorrelation matrix took 47 s in MATLAB, which is optimized for matrix computations. For the neural networks, the complete set of data is presented to the learning algorithm in several iterations called epochs. In NeuroSolutions, whose programming is based on C, 20,010 samples were presented 130 and 1000 times in 22 min, 15 s and 6 min, 35 s for the TDNN and RMLP, respectively [67]. The TDNN was trained with backpropagation and the RMLP was trained with BPTT [55] with a trajectory of 30 samples and learning rates of 0.01, 0.01, and 0.001 for the input, feedback, and output layers, respectively. Momemtum learning was also implemented with a rate of 0.7. One hundred Monte

TABLE 1.2 Model Parameters

	WF	TDNN	Kalman filter	RMLP
Training time	47 s	22 min, 15 s	2 min, 43 s	6 min, 35 s
Number of epochs	1	130	1	1000
Cross validation	N/A	1000 pts.	N/A	1000 pts.
Number of inputs	104	104	104	104
Number of tap delays	10	10	N/A	N/A
Number of hidden PEs	N/A	5	113 (states)	5
Number of outputs	3	3	9	3
Number of adapted weights	3120	5215	12073	560
Regularization	0.1 (RR)	1×10^{-5} (WD)	N/A	1×10^{-5} (WD)
Learning rates	N/A	1×10^{-4} (input) 1×10^{-5} (output)	N/A	1×10^{-2} (input) 1×10^{-2} (feedback) 1×10^{-3} (output)

Carlo simulations with different initial conditions were conducted of neuronal data to improve the chances of obtaining the global optimum. Of all the Monte Carlo simulations, the network with the smallest error achieved a MSE of 0.0203 ± 0.0009. A small training standard deviation indicates the network repeatedly achieved the same level of performance. Neural network training was stopped using the method of cross validation (batch size of 1000 pts.), to maximize the generalization of the network [60]. The Kalman proved to be the slowest to train since the update of the Kalman gain requires several matrix multiplies and divisions. In these simulations, the number of epochs chosen was based upon performance in a 1000-sample cross-validation set, which will be discussed in the next section. To maximize generalization during training, ridge regression, weight decay, and slow learning rates were also implemented.

The number of free parameters is also related to the computational complexity of each model given in Table 1.3. The number of multiplies, adds, and function evaluations describe how demanding the topology is for producing an output. The computational complexity especially becomes critical when implementing the model in a low-power portable DSP, which is the intended outcome for BMI applications. In Table 1.3 define N_0, t, d, and N_1 to be the number of inputs, tap delays, outputs, and hidden PEs, respectively. In this case, only the number of multiplications and function evaluations are presented since the number of additions is essentially identical to the number of multiplications. Again it can be seen that demanding models contain memory in the neural input layer. With the addition of each neuronal input the computational complexity of the WF increases by 10 and the TDNN by 50. The Kalman filter is the most computationally complex $[O((N_0 + 9)^3)]$ since both the state transition and output matrix contain dimensionality of the neuronal input. For the neural networks, the number of function evaluations is not as demanding since they contain only five for both the TDNN and RMLP. Comparing the neural network training times also exemplifies the computational complexity of each topology; the TDNN (the most computationally complex) requires the most training time and allows only a hundred presentations of the training data. As a rule of thumb, to overcome these difficulties, BMI architectures should avoid the use of memory structures at the input.

In testing, all model parameters were fixed and 3000 consecutive bins (300 s) of novel neuronal data were fed into the models to predict new hand trajectories. Figure 1.11 shows the output of the three topologies in the test set with 3D hand position for one reaching movement. While only a single movement is presented for simplicity, it can be shown that during the short period of observation (5 min) there is no noticeable degradation of the model fitting across time. From the plots it can be seen that qualitatively all three topologies do a fair job at capturing the reach to the food and the initial reach to the mouth. However, both the WF and TDNN cannot maintain the peak values of HP at the mouth position. Additionally the WF and TDNN have smooth transitions between the food and mouth while the RMLP sharply changes its position in this region. The traditional way to quantitatively report results in BMI is through the correlation coefficient (CC) between

TABLE 1.3 Model Computational Complexity

	Multiplications	Function evaluations
WF	$N_0 \times t \times d$	N/A
TDNN	$N_0 \times t \times N_1 \times d$	N_1
Kalman filter	$O((N_0 + 9)^3)$	N/A
RMLP	$N_0 \times N_1 + N_1 \times d + N_1 \times N_1$	N_1

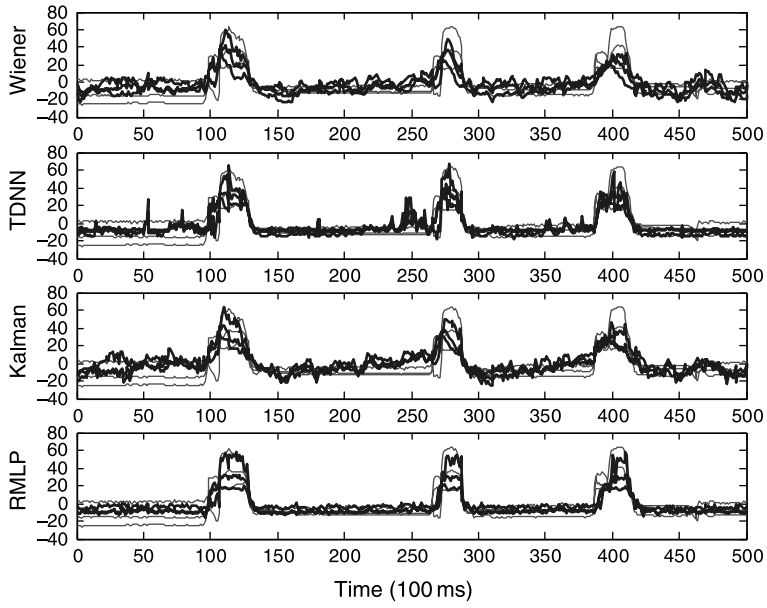

Figure 1.11 Testing performance for three reaching movements.

the actual and estimated hand position trajectories as shown in Table 1.4 which was computed for the entire trajectory. The WF and TDNN perform similarly on average in their CC values while the RMLP has significantly greater performance. In general, the overall experimental performance can also depend on the movement trajectory studied, animal, daily recording session, and variability in the neurons probed.

The reaching task which consists of a reach to food and subsequent reach to the mouth is embedded in periods where the animal's hand is at rest as shown by the flat trajectories to the left and right of the movement. Since we are interested in how the models perform in each mode of the movement we present CC for movement and rest periods. The performance metrics are also computed using a sliding window of 40 samples (4 s) so that an estimate of the standard deviation could be quantified. The window length of 40 was selected because each movement spans about 4 s.

The reaching task testing metrics are presented in Table 1.5. It can be seen in the table that the CC can give a misleading perspective of performance since all the models produce approximately the same values. Nevertheless, the Kolmogorov–Smirnov (K–S) for a p value of 0.05 is used to compare the correlation coefficients with the simplest model, the WF. The TDNN, Kalman, and RMLP all produced CC values that were significantly different than the FIR filter and the CC values itself can be used to gauge if

TABLE 1.4 Model Performance

	Correlation coefficient		
	X	*Y*	*Z*
WF	0.52	0.60	0.64
TDNN	0.46	0.56	0.58
Kalman filter	0.56	0.64	0.65
RMLP	0.67	0.76	0.83

TABLE 1.5 Comparison of Reaching vs. Resting Movements

	WF	TDNN	Kalman filter	RMLP
Correlation coefficient (movement)	0.83 ± 0.09	0.08 ± 0.17	0.83 ± 0.11	0.84 ± 0.15
CC K–S test (movement)	0	1	1	1
Correlation coefficient (rest)	0.10 ± 0.29	0.04 ± 0.25	0.03 ± 0.26	0.06 ± 0.25
CC K–S test (rest)	0	1	1	1

the difference is significantly better. All four models have poor resting CC values which can be attributed to the output variability in the trajectory (i.e. there is not a strong linear relationship between the output and desired trajectories).

1.6 PERFORMANCE DISCUSSION

With the performance results reported in these experiments we can now discuss practical considerations when building BMIs. By far the easiest model to implement is the WF. With its quick computation time and straightforward linear algebra mathematics it is clearly an attractive choice for BMIs. We can also explain its function in terms of simple weighted sums of delayed versions of the ensemble neuronal firing (i.e., it is correlating neuronal activity with HP). However, from the trajectories in Figure 1.11, subplot 1, the output is noisy and does not accurately capture the details of the movement. These errors may be attributed, first, to the solution obtained from inverting a poorly conditioned autocorrelation matrix and, second, to the number of free parameters in the model topology. While we may think that by adding nonlinearity to the WF topology as in the TDNN we can obtain a more powerful tool, we found that the large increase in the number of free parameters overshadowed the increase in performance. We have also found that training the TDNN for this problem is slow and tedious and subject to getting trapped in local minima. The next model studied, the Kalman filter, was the most computationally complex to train and contained the largest number of free parameters (see comparison in Tables 1.2 and 1.3), which resulted in noisy trajectories similar to the linear model. Training this model involved the difficult mapping from a lower dimensional kinematic state space to the neuronal output space as well as initial estimates of the noise covariances, which are unknown to the operator. In contrast, many of these training and performance issues can be overcome in the RMLP. With the choice of moving the memory structure to the hidden layer, we immediately gain a reduction in the number of free parameters. This change is not without a cost since the BPTT training algorithm is more difficult to implement than, for example, the WF. Nevertheless, using a combination of dynamics and nonlinearity in the hidden layer also allowed the model to accurately capture the quick transitions in the movement as well as maintain the peak hand positions at the mouth. Capturing these positions resulted in larger values in the correlation coefficient. While the RMLP was able to outperform the other two topologies, it is not free from error; the output is still extremely noisy for applications of real BMIs (imagine trying to grasp a glass of water). Additionally the negative sloping trajectories for the reach to the food were not accurately captured. The search for the right modeling tools and techniques to overcome the errors presented here is the subject of future research for optimal signal processing for BMIs.

REFERENCES

1. SCHMIDT, E. M. (1980). "Single neuron recording from motor cortex as a possible source of signals for control of external devices." *Ann. Biomed. Eng.* 8:339–349.
2. GEORGOPOULOS, A., J. KALASKA, ET AL. (1982). "On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex." *J. Neurosci.* 2:1527–1537.
3. GEORGOPOULOS, A. P., A. B. SCHWARTZ, ET AL. (1986). "Neuronal population coding of movement direction." *Science* 233(4771):1416–1419.
4. GEORGOPOULOS, A. P., J. T. LURITO, ET AL. (1989). "Mental rotation of the neuronal population vector." *Science* 243(4888):234–236.
5. ANDERSEN, R. A., L. H. SNYDER, ET AL. (1997). "Multimodal representation of space in the posterior parietal cortex and its use in planning movements." *Annu. Rev. Neurosci.* 20:303–330.
6. CHAPIN, J. K., K. A. MOXON, ET AL. (1999). "Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex." *Nature Neurosci.* 2(7):664–670.
7. MORAN, D. W. AND A. B. SCHWARTZ (1999). "Motor cortical activity during drawing movements: Population representation during spiral tracing." *J. Neurophysiol.* 82(5):2693–2704.
8. MORAN, D. W. AND A. B. SCHWARTZ (1999). "Motor cortical representation of speed and direction during reaching." *J. Neurophysiol.* 82(5):2676–2692.
9. SHENOY, K. V., D. MEEKER, ET AL. (2003). "Neural prosthetic control signals from plan activity." *NeuroReport* 14:591–597.
10. WESSBERG, C. R., J. D. STAMBAUGH, ET AL. (2000). "Real-time prediction of hand trajectory by ensembler or cortical neurons in primates." *Nature* 408:361–365.
11. SCHWARTZ, A. B., D. M. TAYLOR, ET AL. (2001). "Extraction algorithms for cortical control of arm prosthetics." *Curr. Opin. Neurobiol.* 11(6):701–708.
12. SANCHEZ, J. C., D. ERDOGMUS, ET AL. (2002). A comparison between nonlinear mappings and linear state estimation to model the relation from motor cortical neuronal firing to hand movements. SAB Workshop on Motor Control in Humans and Robots: on the Interplay of Real Brains and Artificial Devices, University of Edinburgh, Scotland.
13. SANCHEZ, J. C., S. P. KIM, ET AL. (2002). Input-output mapping performance of linear and nonlinear models for estimating hand trajectories from cortical neuronal firing patterns. International Work on Neural Networks for Signal Processing, Martigny, Switzerland, IEEE.
14. SERRUYA, M. D., N. G. HATSOPOULOS, ET AL. (2002). "Brain-machine interface: Instant neural control of a movement signal." *Nature* 416:141–142.
15. TAYLOR, D. M., S. I. H. TILLERY, AND A. B. SCHWARTZ (2002). "Direct cortical control of 3D neuroprosthetic devices." *Science* 296:1829–1832.
16. GAO, Y., M. J. BLACK, ET AL. (2003). A quantitative comparison of linear and non-linear models of motor cortical activity for the encoding and decoding of arm motions. The 1st International IEEE EMBS Conference on Neural Engineering, Capri, Italy, IEEE.
17. KIM, S. P., J. C. SANCHEZ, ET AL. (2003). Modeling the relation from motor cortical neuronal firing to hand movements using competitive linear filters and a MLP. International Joint Conference on Neural Networks, Portland, OR, IEEE.
18. KIM, S. P., J. C. SANCHEZ, ET AL. (2003). "Divide-and-conquer approach for brain machine interfaces: Nonlinear mixture of competitive linear models." *Neural Networks* 16(5/6):865–871.
19. SANCHEZ, J. C., D. ERDOGMUS, ET AL. (2003). Learning the contributions of the motor, premotor AND posterior parietal cortices for hand trajectory reconstruction in a brain machine interface. IEEE EMBS Neural Engineering Conference, Capri, Italy.
20. NICOLELIS, M. A. L. (2003). "Brain-machine interfaces to restore motor function and probe neural circuits." *Nature Rev. Neurosci.* 4:417–422.
21. WOLPAW, J. R., N. BIRBAUMER, ET AL. (2002). "Brain-computer interfaces for communication and control." *Clin. Neurophys.* 113:767–791.
22. NUNEZ, P. L. (1981). *Electric Fields of the Brain: The Neurophysics of EEG*. New York, Oxford University Press.
23. JASDZEWSKI, G., G. STRANGMAN, ET AL. (2003). "Differences in the hemodynamic response to event-related motor and visual paradigms as measured by near-infrared spectroscopy." *NeuroImage* 20:479–488.
24. ROHDE, M. M., S. L. BEMENT, ET AL. (2002). "Quality estimation of subdurally recorded event-related potentials based on signal-to-noise ratio." *IEEE Trans. Biomed. Eng.* 49(1):31–40.
25. FETZ, E. E. AND D. V. FINOCCHIO (1975). "Correlations between activity of motor cortex cells and arm muscles during operantly conditioned response patterns." *Exp. Brain Res.* 23(3):217–240.
26. GEORGOPOULOS, A. P., R. E. KETTNER, ET AL. (1988). "Primate motor cortex and free arm movements to visual targets in three-dimensional space. II. Coding of the direction of movement by a neuronal population." *J. Neurosci. Official J. Soc. Neurosci.* 8(8): 2928–2937.
27. LIN, S., J. SI, ET AL. (1997). "Self-organization of firing activities in monkey's motor cortex: trajectory computation from spike signals." *Neural Computation* 9:607–621.
28. KALASKA, J. F., S. H. SCOTT, ET AL. (1997). Cortical control of reaching Movements. *Curr. Opin. Neurobiol.* 7:849–859.
29. SANGER, T. D. (1996). "Probability density estimation for the interpretation of neural population codes." *J. Neurophysiol.* 76(4):2790–2793.
30. MUSSA-IVALDI, F. A. (1988). "Do neurons in the motor cortex encode movement directions? An alternative hypothesis." *Neurosci. Lett.* 91:106–111.

31. SHENOY, K. V., D. MEEKER, ET AL. (2003). Neural prosthetic control signals from plan activity. *NeuroReport* 14:591–597.
32. HAYKIN, S. (1996). *Adaptive Filter Theory*. Upper Saddle River, NJ, Prentice-Hall International.
33. RIEKE, F. (1996). *Spikes: Exploring the Neural Code*. Cambridge, MA, MIT Press.
34. NICOLELIS, M. A. L. (1999). *Methods for Neural Ensemble Recordings*. Boca Raton, FL, CRC Press.
35. LEONARD, C. T. (1998). *The Neuroscience of Human Movement*. St. Louis: Mosby, 1998.
36. LJUNG, L. (2001). Black-box models from input-output measurements. IEEE Instrumentation and Measurement Technology Conference, Budapest, Hungary, IEEE.
37. KANDEL, E. R., J. H. SCHWARTZ, ET AL., Eds. (2000). *Principles of Neural Science*. New York, McGraw-Hill.
38. THATCH, W. T. (1978). "Correlation of neural discharge with pattern and force of muscular activity, joint position, and direction of intended next movement in motor cortex and cerebellum." *J. Neurophys.* 41:654–676.
39. FLAMENT, D. AND J. HORE (1988). "Relations of motor cortex neural discharge to kinematics of passive and active elbow movements in the monkey." *J. Neurophysiol.* 60(4):1268–1284.
40. KALASKA, J. F., D. A. D. COHEN, ET AL. (1989). "A comparison of movement direction-related versus load direction-related activity in primate motor cortex, using a two-dimensional reaching task." *J. Neurosci.* 9(6):2080–2102.
41. SCOTT, S. H. AND J. F. KALASKA (1995). "Changes in motor cortex activity during reaching movements with similar hand paths but different arm postures." *J. Neurophysiol.* 73(6):2563–2567.
42. TODOROV, E. (2000). "Direct cortical control of muscle activation in voluntary arm movements: a model." *Nature Neurosci.* 3:391–398.
43. WU, W., M. J. BLACK, ET AL. (2002). "Inferring hand motion from multi-cell recordings in motor cortex using a Kalman filter." In *SAB Workshop on Motor Control in Humans and Robots: On the Interplay of Real Brains and Artificial Devices*. University of Edinburgh, Scotland, 66–73.
44. KALMAN, R. E. (1960). "A new approach to linear filtering and prediction problems." *Trans. ASME J. Basic Eng.* 82(Series D):35–45.
45. WIENER, N. (1949). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications.* Cambridge, MA, MIT Press.
46. HAYKIN, S. (1994). *Neural Networks: A Comprehensive Foundation*. New York and Toronto, Macmillan and Maxwell Macmillan Canada.
47. ORR, G. AND K.-R. MÜLLER (1998). *Neural Networks: Tricks of the Trade*. Berlin and New York, Springer.
48. BROCKWELL, A. E., A. L. ROJAS, ET AL. (2003). "Recursive Bayesian decoding of motor cortical signals by particle filtering." *J. Neurophysiol.* 91:1899–1907.
49. MASKELL, S. AND N. GORDON (2001). "A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking." *Target Tracking: Algorithms and Applications* 2:1–15.
50. SANCHEZ, J. C., J. M. CARMENA, ET AL. (2003). "Ascertaining the importance of neurons to develop better brain machine interfaces." *IEEE Trans. Biomed. Eng.* 61(6):943–953.
51. HOERL, A. E. AND R. W. KENNARD (1970). "Ridge regression: Biased estimation for nonorthogonal problems." *Technometrics* 12(3):55–67.
52. SANCHEZ, J. C. (2004). From cortical neural spike trains to behavior: Modeling and Analysis. Department of Biomedical Engineering, University of Florida, Gainesville.
53. FARMER, J. D. AND J. J. SIDOROWICH (1987). "Predicting chaotic time series." *Phys. Rev. Lett.* 50:845–848.
54. FANCOURT, C. AND J. C. PRINCIPE (1996). *Temporal Self-Organization Through Competitive Prediction*. International Conference on Acoustics, Speech, and Signal Processing, Atlanta.
55. PRÍNCIPE, J. C., N. R. EULIANO, ET AL. (2000). *Neural and Adaptive Systems: Fundamentals Through Simulations*. New York, Wiley.
56. GEMAN, S., E. BIENENSTOCK, ET AL. (1992). "Neural networks and the bias/variance dilemma." *Neural Computation* 4:1–58.
57. AKAIKE, H. (1974). "A new look at the statistical model identification." *IEEE Trans. Auto. Control* 19:716–723.
58. WAHBA, G. (1990). *Spline Models for Observational Data*. Montpelier, Capital City Press.
59. NEAL, R. (1996). *Bayesian Learning for Neural Networks*. Cambridge, Cambridge University Press.
60. VAPNIK, V. (1999). *The Nature of Statistical Learning Theory*. New York, Springer Verlag.
61. MOODY, J. (1994). Prediction risk and architecture selection for neural networks. In V. CHERKASSKY, J. H. FRIEDMAN, AND H. WECHSLER, Eds., *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*. New York, Springer-Verlag.
62. AKAIKE, H. (1970). "Statistical predictor identification." *Ann. Inst. Statist. Math.* 22:203–217.
63. CRAVEN, P. AND G. WAHBA (1979). "Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation." *Numer. Math.* 31:377–403.
64. GOLUB, G., H. HEATH, ET AL. (1979). "Generalized cross validation as a method for choosing a good ridge parameter." *Technometrics* 21:215–224.
65. SODERSTROM, T. AND P. STOICA (1989). *System Identification*. New York, Prentice-Hall.
66. MOODY, J. (1992). The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. Advances in Neural Information Processing Systems, San Meteo, CA.
67. LEFEBVRE, W. C., J. C. PRINCIPE, ET AL. (1994). *NeuroSolutions.* Gainesville, NeuroDimension.