

A Comparison between Nonlinear Mappings and Linear State Estimation to Model the Relation from Motor Cortical Neuronal Firing to Hand Movements

*Justin C. Sanchez, **Deniz Erdogmus,
**Jose C. Principe

Department of *Biomedical and **Electrical
Engineering
CNEL, University of Florida
Gainesville, FL 32611
[justin, deniz, principe]@cnel.ufl.edu

***Johan Wessberg, ***Miguel Nicolelis

***Department of Neurobiology
Duke University
Durham, NC 27710
[wessberg, nicoleli]@neuro.duke.edu

Abstract

Recently, several research groups demonstrated that linear models could estimate hand position using populations of action potentials collected in the pre-motor and motor cortical areas of a primate's brain. One of the practical applications of this discovery is to restore movement in patients suffering from paralysis. In this paper, we compare two different approaches to extend this work. One uses a recurrent neural network to create a nonlinear input-output model directly from spike trains to hand positions. The other utilizes both the spike trains and the hand positions as states and applies optimal linear state estimation (Kalman filter) to arrive at the desired model. Both approaches show very accurate position estimations (c.c. larger than 0.9) during reaching movements. Each approach has strengths and weaknesses that will be compared experimentally.

1 Introduction

Man-machine interaction has a long and diverse history. The crux of the interaction is based on transferring the intent of the individual to the machine. Throughout history, the operator's limbs (hands, arms, legs, or feet) have been the conveyers of control, under visual and/or proprioceptive feedback. The intent of our work is to substitute physical control by signals derived directly from the operator's brain to drive a variety of machines for purposes of locomotion, enhanced ability, or virtual reality (remote) interaction. One of the key motivations is to help patients suffering from neurological disorders in which motor control of the limbs has been lost.

Nicolelis and colleagues (Wessberg et al., 2000) demonstrated that firing patterns from ensembles of

cortical neurons could successfully predict (in real time through a linear model and time-delay neural network TDNN) the hand position of a primate. In the prediction procedure, large arrays of 100+ microelectrodes are implanted in the pre-motor and motor areas of a primate. Electrode outputs are processed by spike detection and sorting algorithms to determine firings of single neurons. Spike counts in 100 msec windows are then fed to either a 104x10 (104 channels and 10 delays) finite impulse response filter (FIR) trained with least squares (effectively a Wiener filter (Haykin, 1996)) or a 1040x15x3 TDNN (1040 for 104 channels with 10 delays each) trained with conjugate gradient to match the x, y, z coordinates of the primate's hand. They showed that they could successfully predict the location (3-D) of the primate hands in real-time with average correlation coefficients ranging from 0.6 to 0.7 over 20 minute averages.

Other groups have also demonstrated neural control of devices using many methods. Chapin and colleagues utilized a recurrent neural network to predict lever pressing from ensembles of rat cortical neurons (Chapin et al., 1999). Nonhuman primate spiral tracing prediction with the population vector algorithm has been proposed by Schwartz (Moran et al., 1999).

In this paper, we extend this work into two different directions with the hope of understanding the nature of the relationship we need to model. The first approach keeps the input-output modeling used in (Wessberg et al., 2000) but substitutes the linear filter and TDNN by a recurrent neural network (RNN). The potential advantages of the recurrent neural network are that it is a nonlinear universal approximator and requires much fewer parameters than the linear and TDNN models. However, training the RNN is a nontrivial task, and during testing the system parameters are fixed. Moreover,

it does not attempt to explicitly model the intrinsic noise in the data.

Due to these weaknesses, we also studied an alternative modeling approach based on an optimal recursive state estimation based on a Kalman filter. The Kalman filter is a state estimator that not only estimates the internal state variables for a linear dynamical system (Kalman, 1960), but also produces a generative model for the data and models probabilistically the noise. In this specific application, the system state variables include the hand position, velocity, and acceleration, and the spike trains, which are estimated in real time from the hand positions and firing patterns of cortical neurons. The recursion of the Kalman filter is well suited for learning the nature of biological motor systems, since the states are intrinsically related in time. Although the trained model, which translates the firing patterns of cortical neurons to control parameters, is time-invariant, the Kalman state estimator is robust due to its adjustable gain during testing. However, it assumes a linear relationship in the modeling and it has a large state vector. Hence, one of the goals of this paper is to find out how the different characteristics of the two modeling approaches are going to affect performance. This paper is organized as follows: first we present the two data models, then we show how they perform with real data, and finally we discuss the results and conclude.

2 MODELING

In this paper, we employ two modeling approaches. These are input-output modeling using recurrent neural networks and linear state-space modeling for use in Kalman filtering.

2.1 Recurrent Neural Network

In this modeling approach we assume that there exists an unknown system that maps spike trains into hand positions, and by observing both its inputs and the outputs we can adapt a nonlinear model that can approximate the desired relationship. Here the model is a recurrent multi-layer perceptron (MLP) proposed in (Puskorius et al., 1996). This network differs from an MLP since it contains feedback connections in its hidden layer. The architecture consists of an input layer with 104 channels, a hidden layer of nonlinear processing elements (PEs), (in this case tanh), and an output layer of linear PEs.

Fig. 1 depicts the topology of the recurrent network that is used in our studies. Each hidden layer PE is connected to every other hidden PE using a unit time delay. We can see in (1) that the state produced at the output of the first hidden layer is a nonlinear function of a weighted combination of the current input and the previous state. The feedback of the state allows for continuous representations on multiple timescales. The

output layer is a simple linear combination, shown in (2), of the hidden layer states.

$$y_1(n) = f(w_1 x(n) + w_f y_1(n-1)) \quad (1)$$

$$y_2(n) = w_2 y_1(n) + B \quad (2)$$

Unlike the linear model used in (Wessberg et al., 2000), there is no need to use memory at the input layer. This reduces the number of free parameters in the input layer dramatically (from 1,040 to 104). Memory is created by feeding the states of the hidden PEs among themselves. Each of the hidden PEs outputs can be thought of as a nonlinear adaptive basis of the input space utilized to project the large dimensionality data. These projections are then linearly combined to form the outputs of the RNN that will predict the desired hand movements. The MLP from which this topology is derived has been shown to be a universal mapper in R^n (Cybenko, 1989). The time delay neural network (TDNN) has been also shown to be a universal mapper in myopic functional spaces (Sandberg et al., 1997). Although no theoretical work to prove the universal approximation of the recurrent MLP is known, we expect it to display the same universality because it can be unfolded in a TDNN (Príncipe et al., 2000). Hence this network when properly dimensioned and trained has the power to find the mapping between spike trains and hand positions in 3D space.

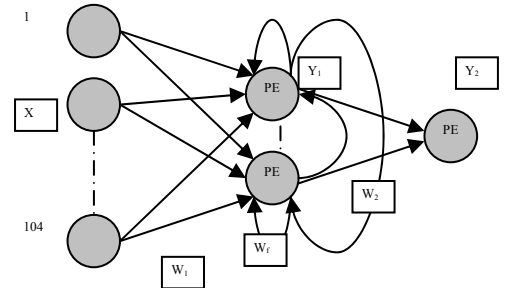


Figure 1. State Recurrent Fully Connected Neural Network

The most commonly used criterion to adapt the parameters of adaptive systems is the mean square error between the desired response and the system output (Príncipe et al., 2000). Although no analytic solution to solve the nonlinear equation of the RNN is known, adapting its parameters can be achieved using the gradient descent procedure. But since RNNs are recurrent systems, gradients display dependencies over time, and so the common backpropagation algorithm to train neural networks (Rumelhart et al., 1986) cannot be applied directly. Here we use the Backpropagation Through Time (BPTT) algorithm (Werbos, 1990) to train the RNN. In BPTT, the recurrent network is unfolded to create an equivalent feedforward network, with replicated weights, which span a time trajectory. The length of this trajectory is empirically determined. Input data spanning the trajectory is fed to the feedforward network with a random initial state and the PE outputs are stored. An error vector is created at the output and fed (reversed in time) through

the dual network to produce local errors. The weights are then updated. Finally, the process begins again for the next trajectory. We used the commercial simulator NeuroSolutions to train the RNN (Príncipe et al., 2000).

Sufficiently small stepsizes were chosen to promote generalization and ensure stability. Even with small stepsizes, feedback loops in the network can cause outputs to oscillate. Since previous states are used in future state computations, oscillations propagate and cause poor generalization. The network learning must be reinitialized in this situation. Recurrent networks trained with gradient descent methods have difficulty learning time dependencies in long trajectories. Gradients tend to decay exponentially through the trajectory due to the PE nonlinearities (Kolen et al., 2001). Training recurrent neural networks with BPTT also suffers from a high computational complexity. Activations, injected errors, and copies of weights have to be stored over a trajectory length. For one trajectory T steps long with an N node network, BPTT requires $O(N^2T)$ computations and $O(NT)$ storage (Príncipe et al., 2000).

2.2 Kalman Filtering

We assume the hand position, velocity, acceleration, and the neuronal spike counts are governed by a *linear* dynamical equation. In this model, the state vector is $x = [p_{xyz} \ v_{xyz} \ a_{xyz} \ f_{1...104}]^T$, where p , v , and a are the hand position, velocity, and acceleration vectors, respectively. The spike counts of 104 neurons are included in the state vector as f_1, \dots, f_{104} . First, we create a generative model for the spike train data. The linear dynamic equation for the state is $x_{k+1} = Ax_k + w_k$, where w_k is assumed to be a zero-mean noise term with covariance W . The output mapping (from state to spike trains) for this linear system is simply $y_k = Cx_k + v_k$, where v_k is the zero mean measurement noise with covariance V and y is a vector consisting of the neuron firing patterns binned in non-overlapping windows. In our specific formulation, the output-mapping matrix is $C = [0_{14 \times 9} \ I_{104 \times 104}]$ and the output noise is zero, i.e. $V=0$.

Suppose there are L training samples of x_k and y_k , and we want to determine the model parameters A and W using least squares. The optimization problem to be solved is given by

$$A = \arg \min_A \sum_{k=1}^{L-1} \|x_{k+1} - Ax_k\|^2 \quad (3)$$

The solution to this optimization problem is found to be $A = X_1 X_0^T (X_1 X_1^T)^{-1}$, where we define the matrices

$$X_0 = [x_1 \ \dots \ x_{L-1}], \quad X_1 = [x_2 \ \dots \ x_L] \quad (4)$$

The estimate of the covariance matrix W can then be obtained using

$$W = (X_1 - AX_0)(X_1 - AX_0)^T / (L-1) \quad (5)$$

Once the system parameters are determined using least squares on the training data, the model obtained (A , C , W) 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 and spike counts. Since the Kalman filter formulation requires a reference output from the model, we assign the spike counts, which are the only available signals, to the output.

The Kalman filter is an adaptive implementation of the Luenberger observer where the observer gains are optimized to minimize the state estimation error variance. In real-time operation, the Kalman gain matrix K , is updated using the following equations. Before starting this recursion, the *a priori* error covariance matrix estimate P^- has to be initialized as well as the state vector estimate \hat{x} .

$$P_k^- = AP_{k-1}A^T + W \quad (6)$$

$$K_k = P_k^- C^T (CP_k^- C^T)^{-1} \quad (7)$$

$$\hat{x}_k = A\hat{x}_{k-1} + K_k (Y_k - CAx_{k-1}) \quad (8)$$

$$P_k = (I - K_k C)P_k^- \quad (9)$$

3 Simulations

In this section, we will present experimental results obtained using both topologies described above on recorded neural signals.

3.1 Data

Synchronous, multichannel neuronal spike trains were collected at Duke University using owl monkeys (*Aotus trivirgatus*). Microwire electrodes were implanted in cortical regions with known motor associations (Nicolelis et al., 1997). The firing times of single neurons were recorded while the monkey performed a 3-D reaching task. The monkey hand position was also recorded (with a shared time clock) and digitized with a 200Hz sampling rate. The neuronal firings were binned (added) in non-overlapping windows of 100ms, which represents the local firing rate for a neuron. These spike counts were directly used as inputs to the RNN and as part of the state vector for the Kalman filter. In order to take the monkey's reaction time into account, the spike trains were delayed by 0.23 seconds with respect to the hand position. This delay was chosen based on loose neurophysiologic reasoning, and should be object to optimization in future studies.

A computational difficulty from an adaptive signal processing point of view is caused by the sparse nature of the neuronal data. As a result, most of the computation time is wasted on multiplications with zeros. In addition,

the data displays nonstationary behavior which prevents accurate long-term modeling with stationary architectures.

3.2 RNN Results

The spike counts of each of the 104 neurons were used to train a RNN (104x5x3) with 5 nonlinear hidden PEs and 3 linear output PEs to predict the x , y , and z coordinates of the monkey's hand. Each exemplar of data used for the BPTT training algorithm contained a trajectory length of thirty samples (3 secs). Weight updates occurred after the presentation of 10 exemplars (30 secs). A training set of 20,010 consecutive bins (2,001 secs) of data was utilized. In testing, the network parameters were fixed and 3,000 consecutive bins (300 secs) of novel neuronal data were fed in the network to predict new hand trajectories. The testing results are evaluated in terms of the correlation coefficients between the actual and estimated hand trajectories. The correlation coefficient gives a measure of how well the actual and estimated trajectories are linearly related. A correlation coefficient value of 1 indicates a perfect linear relationship between the two trajectories, while 0 indicates no correlation. This measure of success must be used with caution since high correlation coefficients do not account for biases in the trajectories. A second measure of performance developed is the signal to error ratio (SER) between the actual and estimated hand trajectories. The SER is defined as the square of the desired signal divided by the square of the estimation error, and it gives a measure of the accuracy of estimated position in terms of the error variance. High SERs are desired since they reflect segments with small output error variance.

The x , y , and z coordinates of hand trajectories during testing are presented in the top subplot of Fig. 2. Also shown in the middle subplot are the correlation coefficients between the actual hand position and the estimated coordinates, evaluated using a sliding window of 40 samples (4 seconds). The window length of 40 was selected because each movement spans about 4 seconds. Windowed correlation coefficients reached values above 0.9 while cumulative correlation coefficients (3000 samples, 300 seconds) were 0.52, 0.71, and 0.71 for the x -, y -, and z -directions. The SER shown in the bottom subplot of Fig. 2 (computed using the same sliding window of 40 samples and averaged over the three coordinate directions), reached a value of 31.98. The cumulative SER averaged over all coordinate directions and the entire test set was 1.58. The estimated hand trajectory superimposed on the actual trajectory is shown in Fig. 3.

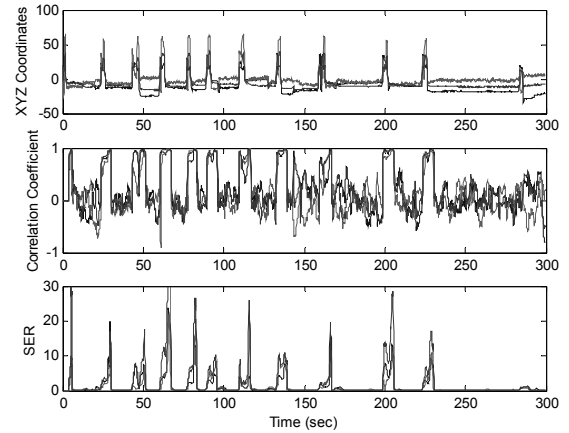


Figure 2. The actual hand coordinates of the primate (a), correlation coefficient between the actual and estimated hand coordinates (b), and SER (c) using RNN.

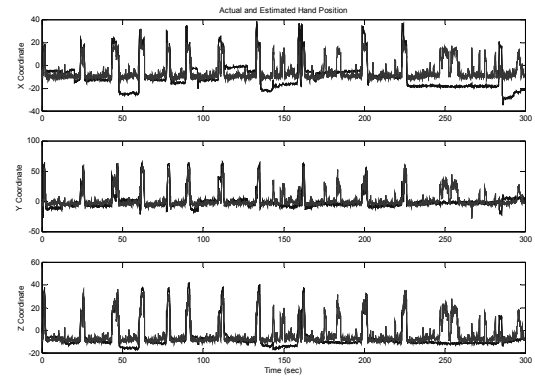


Figure 3. Actual and estimated hand coordinates for the testing period using RNN.

3.3 Kalman Filter Results

The training dataset for the Kalman filter contained 20,000 consecutive time bins (corresponding to 2,000 seconds) of neuronal recordings. During this period the matrices A and W were optimized according to the training data. The test set consisted of 3,000 consecutive time bins immediately following the training set where the Kalman filter was predicting the hand movements from the neural signals. The Kalman gain was continuously adapted during this period. Note that the training and testing data used for the Kalman filter and the RNN are exactly the same sets. This permits the direct comparison of experimental results.

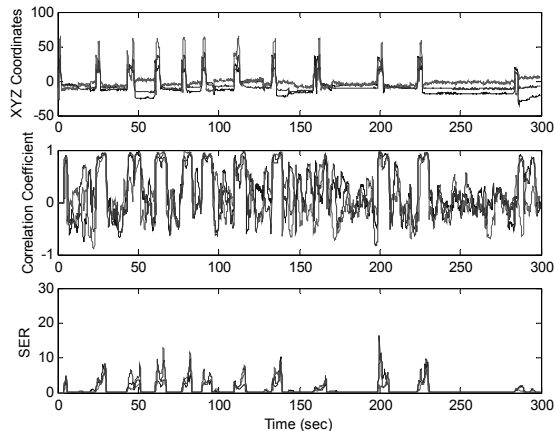


Figure 4. The actual hand coordinates of the primate (a), correlation coefficient between the actual and estimated hand coordinates (b), and SER using Kalman.

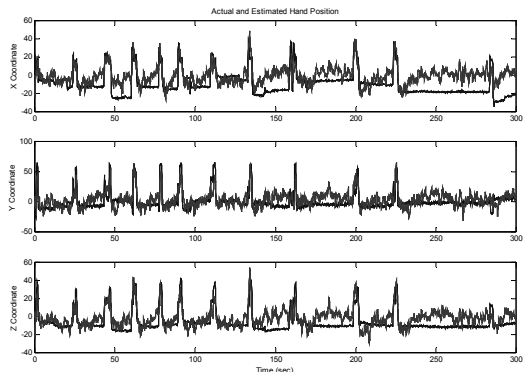


Figure 5. Actual and estimated hand coordinates for the testing period.

The x , y , and z coordinates of the continuous hand trajectories during testing are presented in the top subplot of Fig. 4. Also shown in the middle subplot are the correlation coefficients between the actual hand position and the estimated coordinates, evaluated using a sliding window of 40 samples (4 seconds). An interesting observation we make from these two plots is that the estimated hand trajectory shows high correlation with the actual one when the primate performs a movement (just as in the case of the RNN); in contrast, when its arm is in the rest state, the correlation diminishes. For example, around 225 seconds, the primate performs a movement and a high correlation is achieved, while at 250 seconds the hand is stationary and the correlation decreases. Windowed correlation coefficients reached values above 0.9 while cumulative correlation coefficients (3000 samples, 300 seconds) were 0.55, 0.63, and 0.66 for the x -, y -, and z -directions. Windowed SERs shown in the bottom subplot of Fig. 4 reached a value of 8.89. Note that this value is significantly smaller than the RNN. The cumulative SER averaged over all coordinate directions and the entire test set was 0.89. The Kalman filter

produced estimations with noise power that was fifteen percent larger than the desired signal. For convenience, we also provide the actual and estimated hand position over the testing period in Fig. 5.

4 Discussion and Conclusions

In this paper, we have compared two different frameworks to predict 3-D hand positions from neuronal spike populations. They differ in several relevant aspects:

- The recurrent network is able to find arbitrary (nonlinear) mappings from spike trains to hand movements, while the Kalman filter is restricted to linear mappings in state space.
- The RNN does not explicitly create a complete model for the data, unlike the Kalman filter; this means that the RNN is unable to handle noise in the spike train data.
- The Kalman filter is able to correct the gain on-line, in order to decrease the error between the measured output and the predicted output given by the optimal model. The RNN parameters are all fixed during testing.

Due to the novelty of this type of neuronal spike population data and the absence of neurophysiological models that could help decide on the most appropriate methodology, we decided to experimentally compare these two approaches with real data.

Our results show that during testing, the RNN provides a very good fit for the hand movement in the reaching part of the trajectories. Large SERs during reaching show that RNN estimates have a smaller error variance than the Kalman filter and may be better suited for target acquisition. The correlation between true and predicted hand trajectories is overall smaller than the linear and TDNN models proposed by Wessberg et al.; however, during the reaching movements, the two models proposed here show very accurate estimations (above 0.9). We conclude from the plots (Fig. 2 and Fig. 5) that the part of the trajectory that is more difficult to model for the RNN and the Kalman filter is when the monkey's hand is at rest close to the body. Tracking difficulties in the resting position can be derived either from a problem of the fitting methodology or/and a lack of information in the brain derived signals to model that particular condition. Further work needs to be conducted to explain these experimental findings.

Although not presented in this paper, we have performed estimation tasks for the trained (fixed-parameter) RNN for long periods of testing data (over 15 minutes). There is no noticeable progressive degradation of the fixed RNN model but we have to remember that the task is rather repetitive. Nevertheless, due to the occurrences of trajectories beyond the interpolation/extrapolation ability of the models (using the trajectories encountered during training), we noticed few instances of low correlation during movement. Therefore, larger training sets that span as many different reaching

positions as possible are necessary for successful generalization.

The optimal state estimation approach regards the neuronal firing patterns as the measured output of the system, whereas the state variables include these as well as the hand position, velocity, and acceleration. This is in principle a more powerful approach than linear or nonlinear filters trained to find a mapping from the neuronal firing patterns to the hand position. The proposed model exploits the fact that the future hand position is not only a function of the current cortical firing patterns, but also the current hand position, velocity, and acceleration. However, the experimental results in terms of trajectory fit measured with the correlation coefficient shows that the results are very similar to the RNN (both methods achieve correlation coefficients greater than 0.9 during reach-out movements). The Kalman predictor does a very good job in modeling the large excursions in movement, and has difficulty when the hand is at rest, just like the RNN. We did not see the expected advantage in noise reduction by modeling the data. It seems that for this problem the linear mapping in position, velocity and acceleration and previous spike counts is sufficient to model the unknown relationship with a high degree of precision (measured by the correlation coefficient). In a sense the linear state dynamics are as powerful as the nonlinear input output model created by the RNN. The expected advantage of adapting the Kalman gain to decrease the error during testing did not materialize. This may be due to a change of optimal parameters over time.

The training of the two systems should also be compared. Here, we think that the Kalman filter has an edge, since it uses a very straightforward and well-established training procedure (least squares). In our formulation of the Kalman filter long training data sets are required (20,000 bins) because the state equation has a large matrix with over 10,000 parameters. We tried other model formulations with smaller state vectors (just hand position, velocity and acceleration without the spike counts), but the trajectory estimates were noisier and tended to overshoot the target values. We conclude that to obtain good predictions, the spike counts should be present in the state vector. The RNN has many parameters that need to be tuned to the data, and the computational complexity is higher than that of the Kalman filter. In particular the trajectory length has to be tuned to the application because it should preferably include at least one full cycle of the hand movement. In our full experimentation to be reported elsewhere, the correlation coefficient degrades slowly with the size of the training set, but the learning rates have a critical role in the accuracy of the final model. Further improvements in the training of the RNN are in order. One issue that we did not address in this preliminary study is the selection of the delay between spike data and hand movements. Since this is an unknown parameter, and may even be time varying, it should be part of the optimization procedure. Time

delay estimation is however, a nontrivial problem if brute force procedures are to be avoided. For real-time, portable implementation (DSPs or FPGAs) both algorithms pose challenging problems. Further work to merge the two approaches (data modeling and nonlinear mappings) seems required to improve the accuracy of the predictions.

We would like to end this paper in a positive note. Although the task at hand seems impossible when we think of the complexity of the underlying motor neurophysiology, we (and other groups) have shown that rather simple modeling procedures are capable of approximating to a first order the intricate mappings required to predict hand position from neural spike train populations. We must however point out that these results are just a first step in the design of real-time BMIs. Many challenging problems need to be worked out at the fundamental science, instrumentation, system architecture and modeling levels to deliver an interface that can be used by paralyzed subjects. Some of the modeling issues are: how to effectively include feedback from the robotic arm to the subject; how to choose among many models for goal driven behavior; how to create real-time portable BMIs.

Acknowledgements: This work was partially supported by a seed grant from the College of Engineering, University of Florida and DARPA.

References

- Chapin, J. K., K. A. Moxon, R. S. Markowitz and M. A. Nicolelis (1999). "Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex." *Nature Neuroscience* **2**(7): 664-670.
- Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function." *Mathematics of Control, Signals, and Systems* **2**: 303-314.
- Haykin, S. S. (1996). *Adaptive filter theory*. Upper Saddle River, NJ, Prentice-Hall International.
- Kalman, R. E. (1960). "A New Approach to Linear Filtering and Prediction Problems." *Transactions of the ASME-Journal of Basic Engineering* **82**(Series D): 35-45.
- Kolen, J. F. and S. C. Kremer (2001). *A field guide to dynamical recurrent networks*. New York, IEEE Press.
- Moran, D. W. and A. B. Schwartz (1999). "Motor cortical activity during drawing movements: population representation during spiral tracing." *Journal of Neurophysiology* **82**(5): 2693-2704.
- Nicolelis, M. A., A. A. Ghazanfar, B. M. Faggin, S. Votaw and L. M. Oliveira (1997). "Reconstructing the engram: simultaneous, multisite, many single neuron recordings." *Neuron* **18**(4): 529-537.

- Príncipe, J. C., N. R. Euliano and W. C. Lefebvre (2000). *Neural and adaptive systems: fundamentals through simulations*. New York, Wiley.
- Puskorius, G. V., L. A. Feldkamp, L. I. Davis and Jr. (1996). "Dynamic neural network methods applied to on-vehicle idle speed control." *Proceedings of the IEEE* **84**(10): 1407-1420.
- Rumelhart, D., G. Hinton and R. Williams (1986). "Learning Internal Representations by Error Back-propagation." *Nature* **323**: 533-536.
- Sandberg, I. W. and L. Xu (1997). "Uniform approximation of multidimensional myopic maps." *IEEE Transactions on Circuits and Systems* **44**: 477-485.
- Werbos, P. J. (1990). "Backpropagation Through Time: What It Does and How to Do it." *Proceedings of the IEEE* **78**(10): 1550-1560.
- Wessberg, J., C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan and a. Nicolelis et (2000). "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates." *Nature* **408**(6810): 361-365.