Adaptive Learning Algorithms for Nernst Potential and I-V Curves in Nerve Cell Membrane Ion Channels modelled as Hidden Markov Models

Vikram Krishnamurthy, *Senior Member, IEEE*, Shin-Ho Chung

***Abstract*— We present discrete stochastic optimization algorithms that adaptively learn the Nernst potential in membrane ion channels. The proposed algorithms dynamically control both the ion channel experiment and the resulting Hidden Markov Model (HMM) signal processor and can adapt to time-varying behaviour of ion channels. One of the most important properties of the proposed algorithms are their its self-learning capability – they spends most of the computational effort at the global optimizer (Nernst potential). Numerical examples illustrate the performance of the algorithms on computer generated synthetic data.**

***Index Terms*— Nernst potential, ion channel currents, Discrete Stochastic Approximation, Hidden Markov Models**

I. Introduction

An ion channel is a hole or pore in a nerve cell membrane. In physical structure, an ion channel is a large protein molecule whose different configurations correspond to the ion channel being in a *closed* state or *open* state. The measurement of ionic currents flowing through single ion channels in cell membranes has been made possible by the giga-seal *patch-clamp* technique [1], [2]. This was a major breakthrough for which the authors of [1] won the 1991 Nobel prize in Medicine. Because all electrical activities in the nervous system, including communications between cells and the influence of hormones and drugs on cell function, are regulated by membrane ion channels, understanding their mechanisms at a molecular level is a fundamental problem in biology. Moreover, elucidation of how single ion channels work will ultimately help neurobiologists find the causes of, and possibly cures for, a number of neurological and muscular disorders.

Ion channel currents are typically of the order of pico-amps (i.e. , 10^{-12} amps). In patch clamp experiments these minute ion channel currents are obfuscated by large amounts of thermal noise. Chung *et al.* [3], [4] first introduced the powerful paradigm of Hidden Markov Models (HMMs) to characterize patch-clamp recordings of small ion channel currents contaminated by random and deterministic noise. By using sophisticated HMM signal processing methods, [3], [4] demonstrated that the underlying parameters of the HMM could be obtained to a remarkable precision despite the extremely poor signal to noise ratio. These HMM parameter estimates yield important information into the dynamics of ion channels.

Prior to the works [3], [4], HMMs were mainly used in electrical engineering in the disciplines of artificial speech recognition and target tracking in defense systems. Since the publication of [3], [4], several papers have appeared in the neuro-biological community that generalize the HMM signal models in [3], [4] in various ways to model measurements of ion channels, see [5] and the references therein. With these HMM techniques, it has now possible for neurobiologists to analyze not only large ion channel currents but also small conductance fluctuations occurring in noise.

In this paper, we address the deeper and more fundamental problem of how to adaptively learn and control the behaviour (open state current level) of a single ion channel in a nerve cell membrane. By using recent state-of-the art methods from the electrical engineering disciplines of discrete-event-systems and stochastic control, we develop algorithms to adaptively control the applied voltage to a patch clamp experiment in order to dynamically learn the so called "*Nernst*" potential and current-voltage characteristics of the ion channel. This research transcends the work in [3], [4], [5] that dealt exclusively with HMM signal processing to estimate the channel currents. It addresses the deeper underlying issue of how to dynamically learn and adaptively control the behaviour of the ion channel and the associated HMM signal processing algorithm.

I-V Curve and Nernst Potential: A typical trace of the ion channel current measurement from a patch clamp experiment (after suitable anti-aliasing filtering and sampling) shows that the channel current is piecewise constant discrete time signal that randomly jumps between two values – zero amperes which denotes the *closed state* of the channel, and $I(v)$ amperes which denotes the *open* state. $I(v)$ is called the *open-state* current level. Sometimes the current recorded from single ion channel dwells on one or more intermediate levels, known as conductance substates. For simplicity, here we deal with a two-state process, but the algorithm we propose can readily be extended to channel recordings that contain conductance substates. The open state current level $I(v)$ depends on the voltage v that is applied by the experimenter to the ion channel. Let $\{i_n(v)\}$ denote the discrete-time ion channel current sequence with $n = 0, 1, \ldots,$ denoting discrete time.

In characterizing different types of ion channels, neuro-

Manuscript received ; revised . This work was supported by NSERC and the Australian Research Council

V. Krishnamurthy is with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, V6T 1Z4, Canada and with the Department of Electrical and Electronic Engineering, University of Melbourne (email: vikramk@ece.ubc.ca)

S.H. Chung is with the Biophysics Group, Research School of Physical Sciences and Engineering, Australian National University, ACT 0200, Australia. (email: shin-ho.chung@anu.edu.au)

biologists routinely construct current-voltage (I-V) curves. The curve represents the variation of the open state current level $I(v)$ of the ion channel as a function of the applied voltage value v. Such I-V curves yield a unique signature of a particular ion channel, revealing its operating characteristics. Different ion channel types show different shapes of I-V curves. For example, the relationship may be linear or Ohmic in some ion channels, whereas it may be superlinear or sublinear for other ion channels. The magnitude of the current flowing in one direction at a given potential difference may be equal in some ion channels, whereas the two arms of the I-V curve are asymmetrical in other ion channels. The I-V curve is always monotonically increasing – i.e. , $I(v)$ is a monotonically increasing function of v.

The zero point of the I-V curve, i.e. , the voltage v^* at which the open state current level $I(v^*)$ is zero, is known as the *Nernst potential*. The Nernst potential gives information about the relative concentrations at the two faces of the ionic channel. The value of the open state current level $I(v)$ is described by the Nernst-Planck equation that combines Ohm's and Fick's laws.

Fig.1 shows an example of an I-V curve of a membrane ion channel. The shape of the I-V curve illustrated in the figure incorporates several features observed in many experimentally observed I-V curves: linear and nonlinear segments, a saturation of current with increasing driving force (voltage), and an asymmetry between outward and inward currents (i.e., the I-V curve for $I(v) > 0$ and $I(v) < 0$, respectively). The Nernst potential (voltage at which vertical broken line intersects the v axis) is $v^* = -64$ mV.

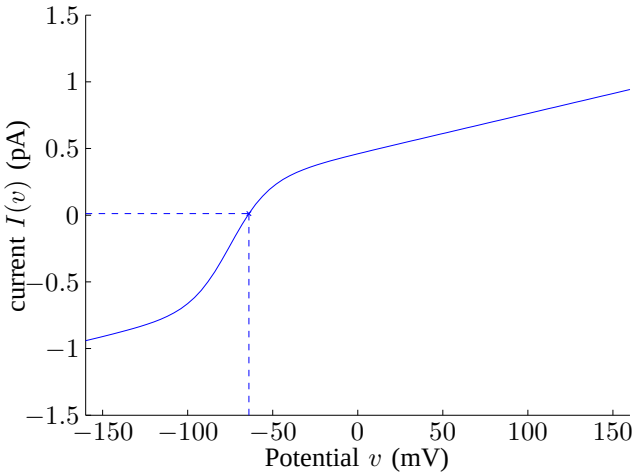

Fig. 1. Typical I-V curve of Membrane Ion Channel. This graph of $I(v)$ versus v shows how the open state current level $I(v)$ varies with applied voltage v. The Nernst potential is the voltage v at which $I(v) = 0$ (dashed lines).

Before describing our adaptive learning and control methodology, we briefly outline two existing methods for estimating the Nernst potential and I-V curves:

(i) *Filtered Trace Brute force Approach*: This method is widely used by neurobiologists to estimate the I-V curve and determine the Nernst potential. First, several possible candidate voltages v are chosen. For each of these voltages v, a patch clamp experiment is run for a long time – typically several minutes. The *measured* ion channel current sequence $\{y_n(v)\}$ is then heavily low-pass filtered – and an estimate of the current level $I(v)$ is determined visually. The Nernst potential is typically deduced by measuring the current levels near its vicinity and then linearly extrapolating the data points. This approach is highly unreliable when the signal to noise ratio is low – which is almost always the case in typical patch clamp experiments. It is also very expensive since the experiments are not controlled to extract maximum information about the Nernst potential.

(ii) *HMM Brute Force Approach*: Given that ionic channels can be modeled extremely well by Hidden Markov Models (HMMs) [4], [5], an obvious improvement to the above approach is to replace the visual estimation step by a HMM parameter estimation algorithm operating over long data sequence $\{y_n(v)\}$ for each voltage value v. The HMM parameter estimation algorithm yields a maximum likelihood estimate (MLE) of the open state current level $I(v)$. In this way by running a separate experiment along with a HMM estimator for each possible voltage value v, the I-V curve can be accurately estimated. Such a brute force approach for estimating the I-V curve is experimentally and computationally inefficient, since running a single expensive patch clamp experiment for several minutes and obtaining an estimate of $(I(v), v)$ at an arbitrary value of v yields no information about the Nernst potential v^*. Furthermore this approach is inherently off-line and is not suitable for tracking I-V curves and Nernst potentials v^* that change slowly with time.

Our approach: The aim of this paper is to propose algorithms that efficiently learns the I-V curve from the noisy observed channel current sequence $\{y_n(v)\}$ by dynamically controlling the applied voltage v. A schematic of our proposed methodology is summarized in Fig.2. The proposed algorithm dynamically controls (schedules) the choice of voltage v at which the ion channel operates in order to efficiently estimate the Nernst potential (zero current point) and deduce how the current increases or decreases as the applied voltage deviates from the Nernst potential. Thus at a given time instant, given the current estimate from a HMM estimator operating at a particular voltage, the aim is to devise a scheduling algorithm that dynamically decides which voltage value to pick at the next time instant to apply to the ion channel. The most important aspect of the resulting combined experiment scheduling/HMM estimation algorithm is its self learning capability – it is provably convergent to the Nernst potential estimate and is provably efficient – that is the algorithm spends more time running the ion channel at the Nernst potential than any other voltage.

The adaptive learning algorithms (see Fig.2) we propose are based on recent discrete stochastic approximation algorithms that have recently been developed in the operations research literature [6], [7]. The basic idea is to generate a homogeneous Markov chain taking values which spends more time at the zero point v^* than at any other voltage v. We propose a novel modification to track a time varying Nernst potential

for adaptive learning.

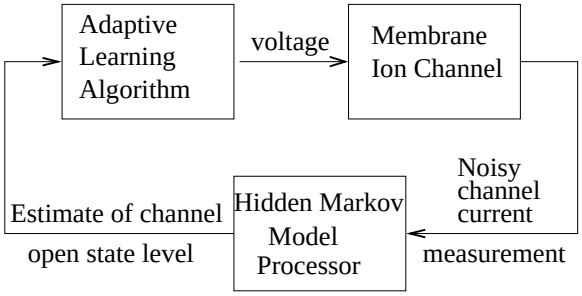

Fig. 2. Dynamic Control of Ion Channel for Learning Nernst Potential

In terms of Fig.2 which depicts our approach, the filtered trace brute force approach (i) outlined above, omits the HMM processor and learning algorithm and yields very poor estimates of the I-V curve. The brute force HMM approach (ii) outlined above, omits the adaptive learning step and hence is highly inefficient.

The rest of this paper is organized as follows. Sec.II describes the patch clamp experiment setup and formally presents a stochastic signal model (HMM) for a channel ion current. In Sec.III the learning algorithm is described. In Sec.IV an adaptive learning algorithm is presented for tracking time varying Nernst potentials. Also the performance of the algorithm is analysed analytically. Finally, Sec.V illustrates the performance of these algorithms in computer simulations. These simulations show that using the learning algorithms result in a remarkable improvement in overall efficiency.

II. Ion Channel Model and Current Voltage (I-V) Curve

In this section we give a precise formulation of the ion channel current signal model and the experimental setup. This allows us to mathematically formulate the Nernst potential and I-V curve learning problem in Sec.III.

A. Measurements of Ionic Currents Flowing Across Single Ion Channels

The measurement of ionic currents flowing through single ion channels in cell membranes has been made possible by the giga-seal patch-clamp technique [1], [2]. A tight seal between the rim of the electrode tip and the cell membrane drastically reduces the leakage current and extraneous background noise, enabling the resolution of the discrete changes in conductance, which occur when single ion channels open or close.

neurobiologists widely use either one of the following two patch clamp experimental setups – the techniques in this paper apply to both these setups.

(i) Cell-Attached Patch: To record currents from single ion channels, the tip an electrode, with the diameter of about 1 μm, is pushed against the surface of a cell, and then a tight seal is formed between the rim of the electrode tip and the cell membrane. A patch of the membrane surrounded by the electrode tip usually contains one or more single ion channels. The current flowing from the inside of the cell to the tip of the electrode through a single ion channel is monitored. This is known as "cell-attached" configuration of patch clamp techniques for measuring ion channel currents through a single ion channel. Fig.3 shows the schematic setup of the cell in electrolyte and the electrode pushed against the surface of the cell.

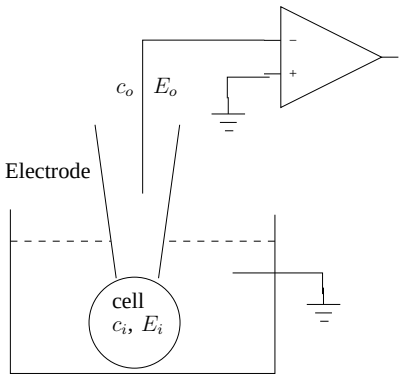

Fig. 3. Cell-Attached Patch Experimental Setup

In a living cell, there is a potential difference between its interior and the outside environment, known as the membrane potential. Typically, the cell interior is about 60 mV more negative with respect to outside. Also, the ionic concentrations (mainly Na^+, Cl^- and K^+) inside of a cell is very different from outside of the cell. In the cell-attached configuration, the ionic strength in the electrode is usually made same as that in the outside of the cell. Let E_i and E_o, respectively, denote the resting membrane potential and the potential applied to the electrode. If E_o is identical to the membrane potential, there will be no potential gradient across the membrane patch confined by the tip of the electrode. Let c_i denote the intracellular ionic concentration and c_o the ionic concentration in the electrode. Here the intra-cellular concentration c_i inside the cell is unknown as is the resting membrane potential E_i. c_o and E_o are set by the experimenter and are known.

Let $v = E_o - E_i$ denote the potential gradient. Both the potential gradient v and concentration gradient $c_o - c_i$ drive ions across an ion channel resulting in an ion channel current $\{i_n(v)\}$. As described in Sec.I, this ion channel current is a piece-wise constant signal that jumps between the values of zero and $I(v)$, where $I(v)$ denotes the current when the ion channel is in the *open state*.

The potential E_o (and hence potential difference v) is adjusted experimentally until the current $I(v)$ goes to zero. This voltage v^* at which the current $I(v^*)$ vanishes is called the *Nernst potential* and satisfies the so called Nernst equation

$$v^* = -\frac{kT}{e} \ln \frac{c_o}{c_i} = -59 \log_{10} \frac{c_o}{c_i} (\text{mV}), \qquad (1)$$

where $e = 1.6 \times 10^{-19}$ C denotes the charge of an electron, k denotes Boltzmann's constant and T denotes the absolute temperature. The Nernst equation (1) gives the potential difference v required to maintain electro-chemical equilibrium

when the concentrations are different on the two faces of the membrane.

Once the Nernst potential v^* is determined, by computing estimates of the current $I(v)$ at several values of v around v^*, the experimenter can straightforwardly determine if the I-V response of the ion channel is Ohmic (linear) or not. Often in the cell-attached experimental setup, the membrane potential can slowly fluctuate with time which results in the Nernst potential slowly evolving with time. The proposed algorithm can track how the membrane potential changes spontaneously or in response to certain experimental maneuver.

(ii) Excised Patch: A patch of the nerve cell membrane confined by the tip of the electrode can be detached from the cell, as shown in Fig. 4. In this excised configuration of patch-clamp technique, both c_i and c_o, as well as E_i and E_o, are known. Here again, it is important to determine the reversal potential E_{rev} accurately. For example, the ionic solutions in the electrode and the bath may contain a mixture of ionic species, such as Na^+, Cl^- and K^+. The ion channel contained in the membrane patch may be permeable to, for example, Na^+ and K^+, as the case with the acetylcholine receptor or Na^+ and Cl^- as the case with certain mutant glycine receptors [8]. It is important to deduce the permeability ratio of the ion channel, namely, the ratio between the number of Na^+ ions and K^+ ions that move across the ion channel per unit time. This ratio can be deduced by accurately determining the zero current using the Goldman-Hodgkin-Katz voltage equation [9] of the form:

$$E_{rev} = \frac{kT}{e} \ln \frac{P_{\mathrm{K}}[\mathrm{K}]_o + P_{\mathrm{Na}}[\mathrm{Na}]_o + P_{\mathrm{Cl}}[\mathrm{Cl}]_o}{P_{\mathrm{K}}[\mathrm{K}]_i + P_{\mathrm{Na}}[\mathrm{Na}]_i + P_{\mathrm{Cl}}[\mathrm{Cl}]_i}. \quad (2)$$

Here P_{K}, P_{Na} and P_{Cl} refer to the permeability of K^+, Na^+ and Cl^-, respectively. As in the cell-attached configuration, the reversal potential may drift slowly in time, owing to changes in the junction potential – implying that the Nernst potential slowly evolves with time.

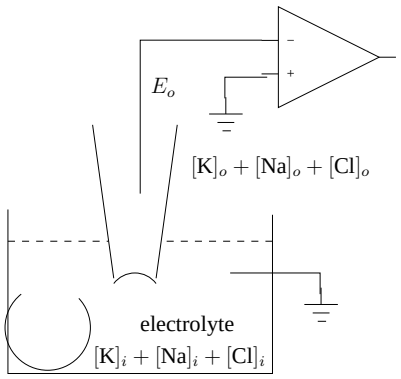

Fig. 4. Excised Patch Experimental Setup

B. HMM for Ion Channel Current and Parameter Estimation of Burst Current State $I(v)$

Since our aim in this paper is to adaptively learn and control the behaviour of an ion channel current modelled as a HMM, in this subsection we formally define the HMM for the ion channel current and briefly describe MLE algorithms for HMMs. Such probabilistic models for ion channels based on HMMs are now widely used [4], [5].

Markov Model for Ion Channel Current: Suppose a patch clamp experiment is conducted with a voltage v applied across the ion channel. Then, as described in [4], [5], the ion channel current $\{i_n(v)\}$, can be modelled as a three state homogeneous first order Markov chain. The state space of this Markov chain is $\{0_{\mathrm{g}}, 0_{\mathrm{b}}, I(v)\}$ corresponding to the physical states of *gap mode*, *burst-mode-closed* and *burst-mode-open*. For convenience, we will refer to the burst mode closed and burst-mode-open states as the *open* and *closed* states, respectively. In the gap mode and the closed state the ion channel current is zero. In the open state, the ion channel current has a value of $I(v)$.

The (3×3) transition probability matrix $A(v)$ of the Markov chain $\{i_n(v)\}$, which governs the probabilistic behaviour of the channel current, is given by

$$A(v) = \begin{array}{c|ccc} & 0_{\mathrm{g}} & 0_{\mathrm{b}} & I(v) \\ \hline 0_{\mathrm{g}} & a_{11}(v) & a_{12}(v) & 0 \\ 0_{\mathrm{b}} & a_{21}(v) & a_{22}(v) & a_{23}(v) \\ I(v) & 0 & a_{32}(v) & a_{33}(v) \end{array} \quad (3)$$

The elements of $A(v)$ are the transition probabilities $a_{ij}(v) = P(i_{n+1}(v) = j | i_n(v) = i)$ where $i, j \in \{0_{\mathrm{g}}, 0_{\mathrm{b}}, I(v)\}$. The zero probabilities in the above matrix $A(v)$ reflect the fact that a ion channel current cannot directly jump from the gap mode to the open state, similarly an ion channel current cannot jump from the open state to the gap mode. Note that in general, the applied voltage v affects both the transition probabilities and state levels of the ion channel current $\{i_n(v)\}$.

HMM Observations: Let $\{y_n(v)\}$ denote the measured noisy ion channel current at the electrode when conducting a patch clamp experiment:

$$y_n(v) = i_n(v) + w_n(v), \quad n = 1, 2, \ldots \quad (4)$$

Here $\{w_n(v)\}$ is thermal noise and is modelled as zero mean white Gaussian noise with variance $\sigma^2(v)$. Typical sample paths of this HMM observation sequence $\{y_n(v)\}$ are shown in Figs.2, 3 and 4. Thus the observation process $\{y_n(v)\}$ is a Hidden Markov model sequence parameterized by the model

$$\lambda(v) = \{A(v), I(v), \sigma^2(v)\} \quad (5)$$

where v denotes the applied voltage. We remark here that the results in this paper trivially extend to observations models where the noise process $w_n(v)$ includes a time-varying deterministic component together with white noise – only the HMM parameter estimation algorithm needs to be modified as in [10].

HMM Parameter Estimation of Current Level $I(v)$: Given the HMM mode for the ion channel current above, estimating $I(v)$ for a fixed voltage v, involves processing the noisy observation $\{y_n(v)\}$ through a Hidden Markov Model maximum likelihood parameter estimator. The most popular way of computing the maximum likelihood estimate (MLE) $I(v)$ is via the Expectation Maximization (EM) algorithm (Baum Welch equations). The EM algorithm is an iterative algorithm

for computing the MLE. It is now fairly standard in the signal processing and neuro-biology literature – see [11] for a recent exposition – or [4] which is aimed at neurobiologists. For convenience the EM algorithm for estimating $I(v)$ is summarized in the appendix.

Let $\hat{I}_\Delta(v)$ denote MLE of $I(v)$ based on the Δ-point measured channel current sequence $(y_1(v), \ldots, y_\Delta(v))$. For sufficiently large batch size Δ of observations, due to the asymptotic normality of the MLE for a HMM [12],

$$\sqrt{\Delta}\left(\hat{I}_\Delta(v) - I(v)\right) \sim N(0, \Sigma(v)) \tag{6}$$

where $\Sigma^{-1}(v)$ is the Fisher information matrix. Thus asymptotically $\hat{I}_\Delta(v)$ is an unbiased estimator of $I(v)$, i.e. , $\mathbf{E}\{\hat{I}_\Delta(v)\} = I(v)$ where $\mathbf{E}\{\cdot\}$ denotes the mathematical expectation operator.

III. Discrete Stochastic Optimization based HMM Algorithm

In this section we formulate the ion channel learning/control problem as a discrete stochastic optimization problem. A novel discrete stochastic approximation algorithm is presented for efficiently solving this problem.

A. Formulation as Discrete Stochastic Optimization Problem

As described in the previous section, determining the Nernst potential v^* requires conducting experiments at different values of voltage v. In patch clamp experiments, the applied voltage v is usually chosen from a finite set. Let

$$v \in V = \{\theta(1), \ldots, \theta(M)\}$$

denote the finite set of possible voltage values that the experimenter can pick. For example, in typical experiments, if one needs to determine the Nernst potential to a resolution of 4 mV, then $M = 80$ and $\theta(i)$ are uniformly spaced in 4 mV steps from $\theta(1) = -160$ mV and $\theta(M) = 160$ mV.

Note that the Nernst potential v^* (zero crossing point) does not necessarily belong to the discrete set V – instead we will find the point in V that is closest to v^* (with resolution $\theta(2) - \theta(1)$). For the rest of this paper with slight abuse of notation we will denote the element in V closest to the Nernst potential as v^*. Thus determining $v^* \in V$ can be formulated as a discrete optimization problem:

$$v^* = \arg\min_{v \in V} |I(v)|^2$$

Our choice of using a quadratic objective function (instead of for example, $\arg\min_{v\in V} |I(v)|$) is because it allows us to conveniently reformulate the optimization problem to be linear in the expected value – see (10) below.

As explained in the HMM formulation above, due to the presence of large amounts of thermal noise, $I(v)$ cannot be exactly evaluated and only unbiased estimates $\hat{I}(v)$ are available. Thus computing the Nernst potential is equivalent to the following discrete stochastic optimization problem:

$$\text{Compute } v^* = \arg\min_{v \in V} \left[\mathbf{E}\{\hat{I}(v)\}\right]^2 \tag{7}$$

where $\hat{I}(v)$ is the MLE of the parameter $I(v)$ of the HMM. Since for a HMM, no closed form expression is available for $\Sigma^{-1}(v)$ in (6), the above expectation cannot be evaluated analytically. This motivates the need to develop a simulation based (stochastic approximation) algorithm.

HMM Brute Force Approach: As explained in Sec.I, the brute force approach [13, Chapter 5.3] for solving (7) involves an exhaustive enumeration as follows: For each $v \in V$, run an independent experiment to gather the sample path $\{y_1(v), y_2(v), \ldots, y_\Delta(v)\}$ for a very large batch size Δ. Compute the MLE $\hat{I}(v)$ via a HMM parameter estimator. Finally pick $\hat{v}^* = \arg\min_{v\in V} |\hat{I}(v)|^2$. Since for any fixed $v \in V$, the MLE $\hat{I}(v)$ is strongly consistent [14], $\hat{I}(v) \to I(v)$ w.p.1, as the batch size $\Delta \to \infty$. This and the finiteness of V imply that as $\Delta \to \infty$,

$$\arg\min_{v\in V} \big(\hat{I}(v)\big)^2 \to \arg\min_{v\in V} \big(I(v)\big)^2 \text{ w.p.1.} \tag{8}$$

Thus in principle, the above brute force simulation method can solve the discrete stochastic optimization problem (7) for large Δ and the estimate is *consistent*, i.e. , (8) holds. However, the method is highly inefficient since $\hat{I}(v)$ needs to be evaluated for each $v \in V$. The evaluations of $\hat{I}(v)$ for $v \neq v^*$ are wasted because they contribute nothing to the estimation of the ion channel current $i(v^*)$ at the Nernst potential v^*. Also the brute force approach does not exploit the fact that the I-V curve is monotonically increasing. Finally, the brute force method is inherently off-line, it cannot be used to adaptively track a slowly time varying Nernst potential.

For $M = 80$, the brute force approach to compute the Nernst potential requires conducting a total of 80 experiments, one at each value of $v \in V$. For a typical sampling rate of 100 kHz and 10 minutes of data per experiment, 6×10^7 observations are obtained per experiment which need to be processed by a HMM MLE estimator.

B. Discrete Stochastic Approximation Algorithm

The idea of discrete stochastic approximation [7] is to design a plan of experiments which provides more observations in areas where the Nernst potential is expected and less in other areas. More precisely what is needed is a dynamic resource allocation (control) algorithm that dynamically controls (schedules) the choice of voltage at which the HMM estimator operates in order to efficiently obtain the zero point and deduce how the current increases or decreases as the applied voltage deviates from the Nernst potential. We propose a discrete stochastic approximation algorithm that is both *consistent* (i.e. , (8) holds) and *attracted* to the Nernst potential. That is, the algorithm should spend more time gathering observations $\{y_n(v)\}$ at the Nernst potential $v = v^*$ and less time for other values of $v \in V$. Thus in discrete stochastic approximation the aim is to devise an *efficient* [13, Chapter 5.3] adaptive search (sampling plan) which allows to find the minimizer v^* with as few samples as possible by not making unnecessary observations at non-promising values of v.

There are several different classes of methods that can be used to solve the discrete stochastic optimization problem (7);

see [7], [15] for a recent survey. When the feasible set V is small (usually 2 to 20 elements), ranking and selection methods, and multiple comparison methods can be used to locate the optimal solution. However for large V the computational complexity of these methods becomes prohibitive.

Problem (7) can also be viewed as a multi-armed bandit problem — which is a special kind of an infinite horizon Markov decision process with an "indexable" optimal policy. However, as mentioned in [7] multi-armed bandit solutions and learning automata procedures often tend to be conservative because they are designed to spend as much time as possible at the optimum solution. Moreover, in the tracking case – where the Nernst potential slowly evolves with time, a bandit formulation would require explicit knowledge of the dynamics of the change of the Nernst potential which is virtually unknown.

In recent years a number of discrete stochastic approximation algorithms have been proposed. Several of these algorithms [6], [7], [16] including simulated annealing type procedures and stochastic ruler [16] fall into the category of random search. In this paper we construct algorithms based on the random search procedures in [6], [7]. The basic idea is to generate a homogeneous Markov chain taking values in V which spends more time at the global optimum than at any other element of V. We will show that these algorithms can be modified for tracking time-varying Nernst potentials. Finally, it is worthwhile mentioning that there are other classes of simulation-based discrete stochastic optimization algorithms such as nested partition methods [15] which combines partitioning, random sampling and backtracking to create a Markov chain that converges to the global optimum – we will examine such methods in future work.

Notation: Let $n = 1, 2, \ldots$ denote discrete time. The proposed algorithm is recursive and requires conducting experiments on batches of data. Since experiments will be conducted over batches of data, it is convenient to introduce the following notation. Group the discrete time into batches of length Δ – typically $\Delta = 10,000$ in experiments. We use the index $N = 1, 2, \ldots$ to denote batch number. Thus batch N comprises of the Δ discrete time instants $n \in \{N\Delta, N\Delta + 1, \ldots, (N + 1)\Delta - 1\}$.
Let $D_N = (D_N(1), \ldots, D_N(M))'$ denote the vector of duration times the algorithm spends at the M possible potential values in V.

Finally for notational convenience define the M dimensional unit vectors, e_m, $m = 1, \ldots, M$ as

$$e_m = \begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}' \tag{9}$$

with 1 in the m-th position and zeros elsewhere.

The discrete stochastic approximation algorithm of [6] is not directly applicable to the cost function (7) – since it applies to optimization problems of the form $\min_{v \in V} \mathbf{E}\{C(v)\}$. However, (7) can easily be converted to this form as follows: Let $\hat{I}_1(v)$, $\hat{I}_2(v)$ be two statistically independent unbiased HMM estimates of $I(v)$. Then defining $\hat{C}(v) = \hat{I}_1(v)\hat{I}_2(v)$, it straightforwardly follows that

$$\mathbf{E}\{\hat{C}(v)\} = \left[\mathbf{E}\{\hat{I}(v)\}\right]^2 = |I(v)|^2 \tag{10}$$

The discrete stochastic approximation algorithm we propose is as follows:

Algorithm 1: [Algorithm for Learning Nernst Potential]

- `Step 0:` (Initialization.) At batch-time $N = 0$, select starting point $X_0 \in \{1, \ldots, M\}$ randomly. Set $D_0 = e_{X_0}$, Set initial solution estimate $\widehat{v}_0^* = \theta(X_0)$.
- `Step 1:` (Sampling.) At batch-time N, sample $\tilde{X}_N \in \{X_N - 1, X_N + 1\}$ with uniform distribution.
- `Step 2:` (Evaluation and Acceptance.) Apply voltage $\tilde{v} = \theta(\tilde{X}_N)$ to patch clamp experiment. Obtain two Δ length batches of HMM observations. Let $\hat{I}_N^{(1)}(\tilde{v})$ and $\hat{I}_N^{(2)}(\tilde{v})$ denote the HMM-MLE estimates for these two batches which are computed using the EM algorithm of Appendix A. Set $\hat{C}_N(\tilde{v})) = \hat{I}_N^{(1)}(\tilde{v})\hat{I}_N^{(2)}(\tilde{v})$.
  Then apply voltage $v = \theta(X_N)$. Compute the HMM-MLE estimates for these two batches, denoted as $\hat{I}_N^{(1)}(v)$ and $\hat{I}_N^{(2)}(v)$. Set $\hat{C}_N(v)) = \hat{I}_N^{(1)}(v)\hat{I}_N^{(2)}(v)$.
  If $\hat{C}_N(\tilde{v}) < \hat{C}_N(v)$, set $X_{N+1} = \tilde{X}_N$, else, set $X_{N+1} = X_N$.
- `Step 3:` (Update occupation probabilities of X_N.)

$$D_{N+1} = D_N + e_{X_{N+1}}$$

- `Step 4:` (Update estimate of Nernst potential.) $\widehat{v}_N^* = \theta(m^*)$ where $m^* = \arg\max_{m \in \{1,\ldots,M\}} D_{N+1}(m)$, set $N \to N + 1$, go to Step 1.

The proof of convergence of the algorithm is given in Theorem 1 below. The main idea behind the above algorithm is that the sequence $\{X_N\}$ (or equivalently $\{\theta(X_N)\}$) generated by Steps 1 and 2 is a homogeneous Markov chain with state space $\{1, \ldots, M\}$ (respectively, V) that is designed to spend more time at the global maximizer v^* than any other state. In the above algorithm, $\hat{v}_N^*$ denotes the estimate of the Nernst potential at batch N.

In Step 3, the vector D_{N+1} is a counter for the occupation times of X_N in the M possible states, i.e. , $D_N(m)$, $m \in \{1, \ldots, M\}$ measures the number of times the Markov chain $\{X_N\}$ has visited the state m until batch N.

The maximization in Step 4 means that the estimate $\hat{v}_N^*$ of the Nernst potential v^* is merely the particular state in V at which the Markov chain $\theta(X_N)$ has spent most time. We will show below that $\hat{v}_N^* \to v^*$ w.p.1, meaning that the algorithm is both attracted to the maximum (i.e. , spends more time in v^* compared to any other state in V) and is consistent.

Interpretation of Step 3 as Decreasing Step Size Adaptive Filtering Algorithm: Define the occupation probability estimate vector as $\hat{\pi}_N = D_N/N$. Then the update in Step 3 can be re-expressed as

$$\hat{\pi}_{N+1} = \hat{\pi}_N + \mu_{N+1}\left(e_{X_{N+1}} - \hat{\pi}_N\right), \quad \hat{\pi}_0 = e_{X_0} \qquad (11)$$

This is merely an adaptive filtering algorithm for updating $\hat{\pi}_N$ with decreasing step size $\mu_N = 1/N$.

Hence Algorithm 1 can be viewed as a decreasing step size algorithm which involves a least mean squares (LMS) algorithm (with decreasing step size) in tandem with a random search step and evaluation (Steps 1 and 2) for generating X_m. Fig. 5 shows a schematic diagram of the algorithm with this LMS interpretation for Step 3.

Implementation Details:

1. Since the sequence $\{\hat{v}_N^*\}$ does not feed back into the recursive part of Algorithm 1, it does not have to be computed at each time instant. It is clearly not necessary to store the sequences $\{X_N\}$, $\{\hat{v}_N\}$, $\{\hat{C}_N\}$ $\{D_N\}$ for all $N = 1, 2, \ldots$. They can be overwritten at each time. The main memory overhead required for the algorithm is storing the local variables at batch N, which requires $O(M)$ memory.

2. In Step 2, at batch-time N, instead of running the experiment to record 2 batches of data and computing $\hat{C}_N(v)$, in numerical experiments we used $\hat{C}_{N-1}(v)$ (i.e. , the estimate from the previous batch). The acceptance step is then: If $\hat{C}_N(\tilde{v}) < \hat{C}_{N-1}(v)$, set $X_{N+1} = \tilde{X}_N$, else, set $X_{N+1} = X_N$. In our numerical experiments this did not seem to affect convergence of the algorithm. With this simplification, Step 2 only requires applying the voltage $\tilde{v}$ to obtain two Δ-length batches of HMM observations.

3. *Complexity*: The computational cost of Steps 1, 3 and 4 are negligible compared to Step 2, hence we only consider the complexity of Step 2. For a batch size of $\Delta = 10000$ HMM data points, running the EM algorithm for 500 iterations on a 2 GFlop Pentium 4 takes approximately 0.002 secs. (This is because each iteration of the EM algorithm for a S-state Markov chain and Δ length data batch involves $S^2\Delta$ multiplications). In comparison, since the experimental data is obtained at a sampling frequency of 100 kHz, obtaining a batch of 10000 HMM measurements takes approximately, 0.1 secs. So the computational time of Algorithm 1 is only 2% of the data acquisition time – meaning that the algorithm is suitable for real time control of the patch clamp experiment. In particular, for Step 2 after collecting the first batch of Δ points, a HMM processor can be used to directly process these measurements (in 0.002 secs) while the second batch of Δ points is being collected (which takes 0.1 secs).

4. *Test for Ohmic I-V Curve*: Because Algorithm 1 is attracted to the Nernst potential v^*, i.e. , it spends more time at v^* than any other voltage in V, the algorithm also spends more time at voltages $v \in V$ close to v^* than those far away from v^*. This results in improved accuracy of the estimates $\hat{I}(v)$ for v close to v^*. Having determined these estimates $\hat{I}(v)$ for v close to v^*, simple hypothesis tests can be used to determine if $\hat{I}(v)$ vs v is linear in the neighborhood of v^*.

C. Convergence and Attraction of Algorithm 1

Throughout this section we assume:

(N) The batch size Δ is sufficiently large (e.g $\Delta = 10,000$) so that due to (6), the MLE estimates $\hat{I}_N^{(1)}(v)$, $\hat{I}_N^{(2)}(v)$ are $N(I(v), \Sigma(v))$ Gaussian random variables.

As mentioned above, processing a batch size $\Delta = 10,000$ takes negligible time compared to the data acquisition time. Actually in the theorem below, we only require that $\hat{I}_N^{(1)}(v)$, $\hat{I}_N^{(2)}(v)$ have symmetric probability density functions with mean $I(v)$. In [6], the following stochastic ordering assumption was used for convergence of the Algorithm 1.

(O) For any $m \in \{1, \ldots, M-1\}$,

$$I^2(\theta(m+1)) > I^2(\theta(m)) \implies P\left(\hat{C}(\theta(m+1)) > \hat{C}(\theta(m))\right) > 0.5$$

$$I^2(\theta(m+1)) < I^2(\theta(m)) \implies P\left(\hat{C}(\theta(m+1)) > \hat{C}(\theta(m))\right) < 0.5$$

Theorem 1: Under the condition (O) above, the sequence $\{\theta(X_N)\}$ generated by Algorithm 1 is a homogeneous, aperiodic, irreducible Markov chain with state space V. Furthermore, Algorithm 1 is attracted to the Nernst potential v^*, i.e. , for sufficiently large N, the sequence $\{\theta(X_N)\}$ spends more time at v^* than an other state. (Equivalently, if $\theta(m^*) = v^*$, then $D_N(m^*) > D_N(j)$ for $j \in \{1, \ldots, M\} - \{m^*\}$.)

Proof: Given that the objective function (10) is exactly of the form of the cost function in [6], we only need to verify condition (O). This is done below. It then follows from [6, Theorem 2.1] that Algorithm 1 converges to a local minimum. However, $I(v)$ is a monotonically increasing function of v.

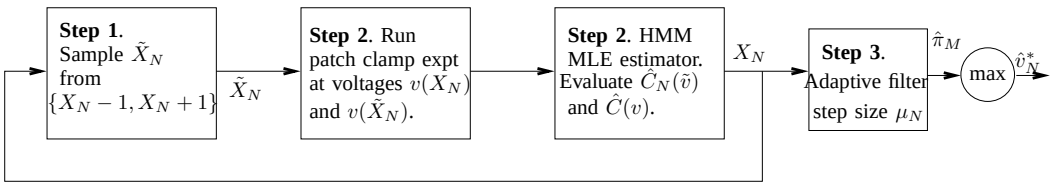

Fig. 5. Schematic of Algorithm 1.

Hence $I^2(v)$ has only a single minimum which is the global minimum. Thus under condition (O), Algorithm 1 converges to the Nernst potential.

We now verify condition (O) for the HMM estimator based I-V model. Recall from (10) that $\hat{C}(v) = \hat{I}_1(v)\hat{I}_2(v)$ where $\hat{I}_1(v), \hat{I}_2(v) \sim N(I(v), \Sigma(v))$ due to normality assumption (N). Expanding out $\hat{C}(v) = \hat{I}_1(v)\hat{I}_2(v)$ yields $\hat{C}(v) = I^2(v) + W(v)$ where $W(v)$ is a zero mean random variable with symmetric probability density function. It is proved in Theorem 3.1 and Lemma 3.1 of [17] that if $W(v)$ is a zero mean random variable with symmetric probability density function then condition (O) holds. ■

Remark: The proof of the above theorem holds as long as the HMM MLE estimates $\hat{I}_1(v)$ and $\hat{I}_2(v)$ are unbiased with symmetric density functions (not necessarily Gaussian) about the mean $I(v)$.

IV. Adaptive Learning Algorithms for Tracking time-varying Current-Voltage (I-V) Curves

Here we consider the case where due to slow fluctuations in the membrane potential, the Nernst potential slowly evolves with time. We denote the time varying Nernst potential by the sequence $\{v_N^*\}$, where as usual $N = 1, 2, \ldots$ denotes batch-time and $v_N^* \in V$. Such non-stationary environments are at the very heart for applications of adaptive stochastic approximation algorithms in statistical signal processing to track time-varying parameters.

A. Constant Step-size Discrete Stochastic Approximation Algorithm

We propose the following constant step-size discrete stochastic approximation algorithm for tracking the time-varying parameter.

Algorithm 2: [Adaptive Algorithm for Tracking Time-Varying Nernst Potential]

- `Step 0, 1 and 2` Identical to Algorithm 1.
- `Step 3` (Constant Step Size Adaptive Filter to update Occupation Probabilities). Replace (11) in Step 3 of Algorithm 1 with the following fixed step-size algorithm least mean square (LMS) algorithm

$$\hat{\pi}_{N+1} = \hat{\pi}_N + \mu \left(e_{X_{N+1}} - \hat{\pi}_N \right), \quad \hat{\pi}_0 = e_{X_0} \qquad (12)$$

  where $0 < \mu < 1$ is the constant step size.
- `Step 4` Identical to Algorithm 1.

Remark: As long as the step size satisfies $0 < \mu < 1$, $\hat{\pi}_N$ is guaranteed to be a probability vector for any N. To see this, note that with $\mathbf{1}_M{}'$ denoting the M dimensional column vector of ones, $\mathbf{1}_M{}'(e_{X_{N+1}} - \hat{\pi}_N) = 0$ implying that $\mathbf{1}_M{}'\hat{\pi}_{N+1} = \mathbf{1}_M{}'\hat{\pi}_N = 1$. Also rewriting (12) as $(1-\mu)\hat{\pi}_N + \mu e_{X_{N+1}}$ implies that all elements of $\hat{\pi}_{N+1}$ are non-negative for $0 < \mu < 1$.

The constant step size μ introduces an exponential forgetting of the past occupation probabilities and permits us to track slowly time-varying Nernst potentials v_N^*. Its schematic is similar to Fig.5 but with μ now a fixed step size in the LMS algorithm. Thus Algorithm 2 can be viewed as a discrete sampling and evaluation step in tandem with a fixed step-size adaptive filtering (e.g. , LMS) algorithm.

B. Performance Analysis of Algorithm 2

The aim of this section is to *analytically* characterize the performance of Algorithm 2 when tracking a randomly time varying Nernst potential. In contrast, Sec.V examines the performance of Algorithm 2 via computer simulations.

In presenting Algorithm 2 above we have made no assumption on how the Nernst potential varies with time. In general, little can can be said analytically about the performance of a tracking algorithm without making some assumption about how the underlying parameter (Nernst potential) varies with time. In adaptive signal processing, a typical method for analyzing the tracking performance of an adaptive algorithm is to postulate a random *hypermodel* for the time variation of the parameter and then characterize the performance of the algorithm for this hypermodel.

Since the time varying Nernst potential $v_N^* \in V$ belongs to a finite state space, for the purpose of our analysis, we chose to describe the evolution of the Nernst potential as a slow Markov chain on V. *Note that this hypermodel assumption is only used for our subsequent analysis, it does not enter the implementation of Algorithm 2 in any way*. The Markov chain hypermodel is one of the most general models available for a finite-state model.

Formally, we make the following assumptions about the time evolution of the Nernst potential.

(M1) *Hypermodel*: Assume that there is a small parameter $\epsilon > 0$ and that the time varying Nernst potential $\{v_N^*\}$ is a discrete-time homogeneous Markov chain, with state space V. The transition probability matrix of this Markov chain is

$$P^\epsilon = I + \epsilon Q, \qquad (13)$$

Here I is an $\mathbb{R}^{M \times M}$ identity matrix, and $Q = (q_{ij}) \in \mathbb{R}^{M \times M}$ is a generator of a continuous-time Markov chain (i.e. , Q satisfies $q_{ij} \geq 0$ for $i \neq j$ and $\sum_{m=1}^{M} q_{im} = 0$ for each $i = 1, \ldots, M$).

Remark: The quantity ϵ being small is to ensure that the Nernst potential v_N^* evolves slowly with time, i.e. , it jumps very infrequently. The idea behind this hyper-model is that the chain will spend most of its time at a constant value. However, due to the presence of the generator Q, from time to time, the chain jumps into some other location. As a result, the Nernst potential becomes one that is slowly but randomly time-varying. Note that μ is the step size used in Algorithm 2 for estimating $\hat{\pi}_N$. Typically for an adaptive algorithm to successfully track a time varying optimum, the rate of change in the true optimum (i.e., ϵ) should be comparable in magnitude or smaller than the tracking speed of the tracking algorithm (i.e., μ).

Theorem 1 says that for fixed Nernst potential $v_N^* = v$, the sequence $\{X_N\}$ generated by Algorithm 2 is a conditional Markov chain (conditioned on v_N^*). Thus the behaviour of the sequence $\{\hat{\pi}_N\}$ generated by Algorithm 2 exactly fits the following assumption.

(S) Let $\{X_N\}$ be an M-state conditional Markov chain (conditioned on the true Nernst potential v_N^* at batch-time N). The state space of $\{X_N\}$ is $\{1,\ldots,M\}$. For each $v \in M$, $A(v) = (a_{ij}(v)) \in S \times S$, the transition probability matrix of X_N, is defined by

$$a_{ij}(v) = P(X_{N+1} = j | X_N = i, v_N^* = v), \quad v \in V \tag{14}$$

where $i, j \in \{1, ..., M\}$. Each of the m_0 matrices $A(v)$, for $v \in M$ is irreducible and aperiodic.

The assumptions on irreducibility and aperiodicity of $A(v)$ imply that for each $v \in V$, there exists a unique stationary distribution $\pi(v) \in \mathbb{R}^{M\times 1}$ satisfying (recall M is the number of elements in V)

$$\pi'(v) = \pi'(v)A(v), \quad \text{and} \quad \pi'(v)\mathbf{1}_M = 1, \tag{15}$$

where $\mathbf{1}_M \in \mathbb{R}^{M\times 1}$ with all entries being equal to 1. Given that (12) is an LMS algorithm for estimating and tracking $\pi(v_N^*)$, our aim is to analyse the performance of (12) in tracking the time-varying distribution $\pi(v_n^*)$ that depends on the underlying time varying Nernst potential v_n^*.

The following result gives a mean-squared bound on the tracking error of the occupation probability estimate $\hat{\pi}_N$ generated by the adaptive Algorithm 2. Define $\widetilde{\pi}_N = \widehat{\pi}_N - \pi(v_N^*)$. Then (12) can be rewritten as

$$\widetilde{\pi}_{N+1} = \widetilde{\pi}_N - \mu\widetilde{\pi}_N + \mu(e_{X_{N+1}} - \pi(v_N^*)) + \pi(v_N^*) - \pi(v_{N+1}^*). \tag{16}$$

The theorem below says that for small ϵ (rate of change of Nernst potential) and μ (step size), the error in $\hat{\pi}_N$ of Step 3, Algorithm 2 compared to the true occupation probability $\pi(v_N^*)$ is small – the error is $O(\mu)$. So as $\mu \to 0$, this error goes to zero. The proof is given in [18] and involves some fairly sophisticated concepts in martingales and stochastic Lyapunov functions.

Theorem 2: Under the conditions (M1) and (S), if if $\epsilon = O(\mu)$, then for sufficiently large N,

$$\mathbf{E}|\widetilde{\pi}_N|^2 = O(\mu). \tag{17}$$

Due to the discrete valued nature of the underlying Nernst potential $v_N^* \in V$, it makes sense to give bounds on the probability of error of the Nernst potential estimate $\hat{v}_N^*$ generated by Step 4 of Algorithm 2. Define the error event E and probability of error $P(E)$ as

$$E = \{\widehat{v}_N^* \neq v_N^*\}, \quad P(E) = P(\widehat{v}_N^* \neq v_N^*). \tag{18}$$

Thus E depicts the event that the Nernst potential estimate at batch-time N is incorrect. Clearly E depends on the batch-time N; however, we suppress the N here for notational simplicity. Based on the mean square error of Theorem 2 above, the following result holds:

Theorem 3: Under conditions (M) and (S), if $\mu = \epsilon$, then for sufficiently large N, the error probability of the Nernst potential estimate $\widehat{v}_N^*$ generated by Algorithm 2 satisfies

$$P(E) \leq K\mu^{1-2\gamma}, \tag{19}$$

where K and $0 < \gamma < 1/2$ are arbitrary positive constants independent of μ and ϵ.

The above result serves as a useful consistency check for Algorithm 2: As $\mu \to 0$ and $\epsilon \to 0$, the probability of error $P(E)$ of Algorithm 2 in estimating the Nernst potential goes to zero. So for small μ and ϵ, the error probability is small.

Proof: The estimate of the maximum generated by the discrete stochastic approximation algorithm at batch-time N is $\widehat{\pi}_N^* = \arg\max_j \widehat{\pi}_N^j$ (where $\widehat{\pi}^j$ denotes the jth component of the M-dimensional vector $\widehat{\pi}_N$). Thus the error event E in (18) is equivalent to $E = \left\{I(\arg\max_i \pi^i(v_N^*) \neq \arg\max_j \widehat{\pi}_N^j)\right\}$, where $I(\cdot)$ denotes the indicator function. Then clearly the complement event $\bar{E} = \left\{I(\arg\max_i \pi^i(v_N^*) = \arg\max_j \widehat{\pi}_N^j)\right\}$ satisfies

$$\begin{aligned}\bar{E} &\supseteq \left\{I(|\max_i \pi^i(v_N^*) - \max_j \widehat{\pi}_N^j| \leq \min_{i,j} |\pi^i(v_N^*) - \widehat{\pi}_n^j|)\right\} \\ &\supseteq \left\{I(|\max_i \pi^i(v_N^*) - \max_j \widehat{\pi}_N^j| \leq L)\right\}\end{aligned}$$

where

$$L \leq \min_{i,j} |\pi^i(v_N^*) - \widehat{\pi}_n^j| \tag{20}$$

is a positive constant. Then the probability of no error is

$$\begin{aligned}P(\bar{E}) &= P(\arg\max_i \pi^i(v_N^*) \\ &= \arg\max_j \widehat{\pi}_N^j) > P(|\max_i \pi^i(v_N) - \max_j \widehat{\pi}_N^j| \leq L)\end{aligned} \tag{21}$$

for any sufficiently small positive number L. Then using the above equation and Theorem 2

$$\begin{aligned}P(E) &\leq P(|\max_i \pi^i(v_N^*) - \max_j \widehat{\pi}_N^j]| > L) \\ &\leq P(\max_i |\pi^i(v_N^*) - \widehat{\pi}_N^i| > L)\end{aligned} \tag{22}$$

Applying Chebyshev's inequality to (17) yields for any i,

$$P(|\pi^i(v_N^*) - \widehat{\pi}_N^i| > L) \leq \frac{1}{L^2} K\mu$$

for some constant K. Thus (22) yields

$$P(\max_i |\pi^i(v_N^*) - \widehat{\pi}_N^i| > L) \leq \frac{1}{L^2} K\mu \tag{23}$$

It only remains to pick a sufficiently small L. Choose $L = \mu^\gamma$ where $0 < \gamma < \frac{1}{2}$ is arbitrary. It is clear that for sufficiently small μ, L satisfies (20). Then (23) yields $P(E) \leq K\mu^{1-2\gamma}$. ■

V. Numerical Results

Using computer generated synthetic data, we illustrate the performance of Algorithms 1 and 2. All the examples below including simulation of the Markov chain and HMM sample paths were conducted using Matlab™ and its random number generators.

Simulation Model for Ion Channel: We simulated sample paths of the ion channel current $\{i_n(v)\}$ as a Markov chain with transition probability matrix A (see (3)) and open state current level $I(v)$. Here $I(v)$ was generated using I-V curve of Fig.1 and

$$A = \begin{bmatrix} 0.97 & 0.03 & 0 \\ 0.3 & 0.6 & 0.1 \\ 0 & 0.1 & 0.9 \end{bmatrix} \tag{24}$$

The choice of A in (24) implies that the steady state probability vector of the Markov chain computed using (15) is $\pi = [0.834,\ 0.083,\ 0.083]'$. This is consistent with patch clamp experimental data which shows that typically the ion channel current spends about 80% of its time in the gap mode, and about 10% of its time in each of the closed and open states. The I-V curve of Fig.1 was used since it is a particularly difficult case to handle – it is non Ohmic (nonlinear) and has asymmetric behaviour for positive and negative currents.

The observed channel current at the electrode was simulated by adding white Gaussian noise with standard deviation $\sigma(v) = 0.3$ to the simulated ion channel current sequence $\{i_n(v)\}$, resulting in the HMM sequence $\{y_n(v)\}$ (see (4)).

Example 1. Learning and Control Performance of Algorithm 1: Given the above simulation model for the ion channel current, we used Algorithm 1 to determine the Nernst potential v^*. Experiments were run over batch sizes $\Delta = 10,000$.

At Step 0, we selected the starting point at $X_0 = 1$, i.e. , initial applied voltage $v = -160$ mV.

In Step 2, the EM algorithm was run for 500 iterations on each Δ-length batch of HMM observations. As described at the end of Sec.III-B, this takes only about 0.002 secs on a 2 GFlop Pentium 4. The resulting MLEs for the 4 batches, namely $\hat{I}_N^{(1)}(\tilde{v})$, $\hat{I}_N^{(2)}(\tilde{v})$, $\hat{I}_N^{(1)}(v)$ and $\hat{I}_N^{(2)}(v)$ were used to determine $\hat{C}_N(\tilde{v})$ and $\hat{C}_N(v)$.

Fig.6 shows the Nernst potential estimates $\widehat{v}_N^*$ generated by Algorithm 1 for batch-times $N = 0, 1 \ldots, 10000$. As can be seen, the estimate $\widehat{v}_N^*$ rapidly converges to the Nernst potential $v^* = -64$ mV. Also shown in Fig.6 is the true open state current level $I(\widehat{v}_N^*)$ based in the estimated Nernst potential $\widehat{v}_N^*$. As can be seen from the figure, Algorithm 1 dynamically controls the applied voltage to the Nernst potential v^*, so that the open state current level $I(\widehat{v}_N^*)$ goes to zero.

To illustrate the attraction (learning) property of Algorithm 1, i.e. , it spends more time gathering information near the Nernst potential than other voltages, Fig.7 shows the occupation probabilities computed by Eq.(11) of Step 3. As shown in Fig.7, Algorithm 1 spends approximately 14% of its time at the Nernst potential. In comparison, a brute force HMM approach would spend equal resources at all voltages $v \in V$, i.e, its would spend $1/320$ of its time at the Nernst potential v^*. Thus Algorithm 1 is approximately 45 times more efficient than the brute force HMM approach. This implies that to get equally accurate estimates of the Nernst potential, the brute force HMM approach requires the patch clamp experiment to be run 45 times longer than the controlled patch clamp experiment that uses Algorithm 1.

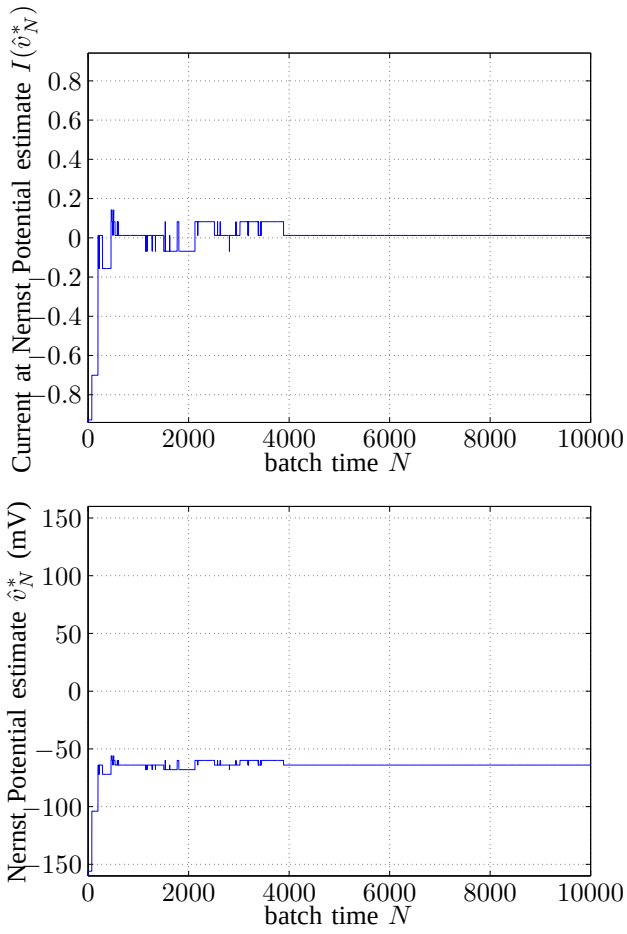

Fig. 6. Nernst potential learning using Algorithm 1.

Example 2. Learning and Adaptive Control Performance of Algorithm 2: Here we consider the case where the I-V curve and hence the Nernst potential jump changes at infrequent intervals. We illustrate how Algorithm 2 can dynamically learn and adaptively control the applied voltage to efficiently compute the Nernst potential.

Our ion channel current model is as follows: For batch-time $0 \leq N < 10,000$, the channel current I-V response and HMM model was the same as above, i.e. , Fig.1. For batch-time $10,000 \leq N \leq 20,000$ the I-V response is shown in Fig.8.

We used Algorithm 2 with step size $\mu = 5 \times 10^{-4}$. Fig.9 shows the Nernst potential estimates, $\hat{v}_N^*$ and the corresponding open state current level estimate $I(\hat{v}_N^*)$ generated by Algorithm 2. As can be seen from Fig.9, although the I-V curve and Nernst jump changes at time $10,000$, the algorithm quickly learns the new ion channel and adaptively controls the channel current to zero. As is commonly observed with adaptive filtering algorithms, we noticed a trade off between

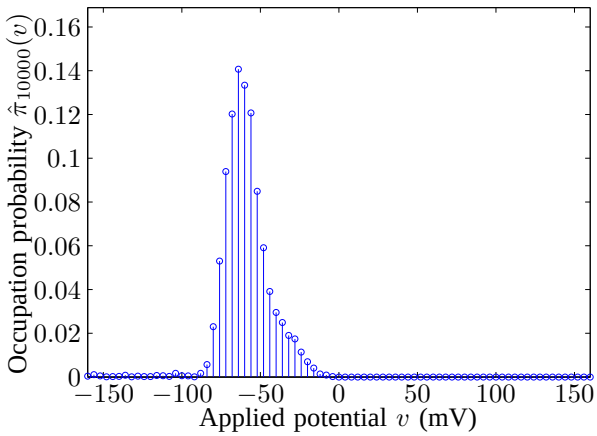

Fig. 7. Occupation probabilities of Discrete Stochastic Approximation Algorithm 1. These occupation probabilities illustrate the attraction property of Algorithm 1 – the algorithm spends more time near the Nernst potential $v^* = -64$ mV than other values of $v \in V$.

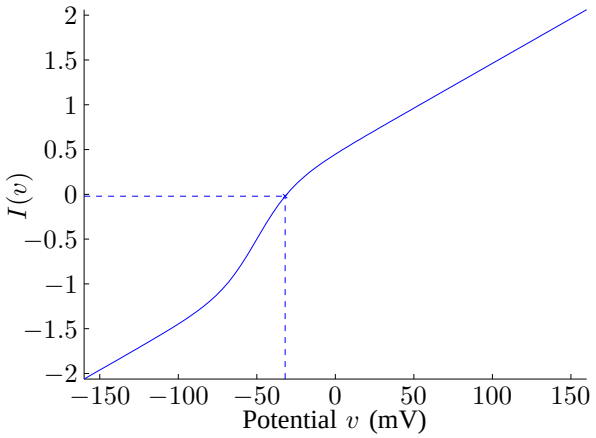

Fig. 8. I-V curve of Membrane Ion Channel from batch-time $10,000 \leq N \leq 20000$ for Example 2 of Sec.V. For batch-time $N < 10,000$ the I-V curve is given in Fig.1.

adaptation speed and steady state convergence depending on the choice of step size μ. For larger μ, the algorithm quickly adapted to the change at time 10,000 but it jumped around the steady state value. For smaller μ the algorithm adapted slowly to the change at time $10,000$ but did not drift around the steady state value.

To demonstrate the attraction behaviour of Algorithm 2, Fig.10 shows snapshots of the occupation probability vector $\hat{\pi}_N$ (generated by Step 3 of Algorithm 2) at a batch-times $N = 10,000$, 12,000 and 20,000. Until batch-time $N = 10,000$, since the Nernst potential is a constant, there is a single visible peak in $\hat{\pi}_{10000}$ at the Nernst potential. At batch-time $N = 12,000$, the algorithm has not yet fully adapted to the change in the I-V curve and Nernst potential. There are two peaks in $\hat{\pi}_{12000}$, one around the old Nernst potential and the other about the new one. By batch-time $20,000$ there is a single pronounced peak in $\hat{\pi}_{20000}$ at the new Nernst potential. Thus Algorithm 2 can dynamically adapt to a time varying behaviour of the ion channel and spends substantially more time at the Nernst potential than any other value of v.

VI. Conclusions and Extensions

In this paper we presented two state-of-the art discrete stochastic approximation algorithms for learning and controlling the behaviour of nerve cell membrane ion channels. A notable feature of the algorithms is their self learning capability – they spend more time gathering information about the Nernst potential than other less interesting voltages. The algorithms use a novel random sampling scheme combined with a stochastic adaptive filter and are provably convergent and attracted to the Nernst potential. In numerical examples on synthetic experimental data, we have shown that the algorithms demonstrate remarkable improvements (e.g. , 40 fold improvement) in efficiency compared to brute force experimental approaches. In future work we will examine the use of other discrete stochastic approximation algorithms such as nested partition methods [15] which combines partitioning, random sampling and backtracking.

Appendix I
Appendix–EM algorithm for Maximum Likelihood Estimation of HMM Parameters

Step 2 of Algorithms 1 and 2 require computing the maximum likelihood estimate (MLE) $\hat{I}_N(v)$ of the HMM parameter $I(v)$ given a Δ-length batch of observations. Recall that $q(v)$ denotes the open state level and $v \in V$ denotes the applied voltage. Here we summarize the EM algorithm for computing the MLE $\hat{\lambda}_N(v) = \{\hat{A}_N(v), \hat{I}_N(v), \hat{\sigma^2}_N(v)\}$ of the Nth batch for the HMM $\lambda(v) = \{A(v), I(v), \sigma^2(v)\}$ defined in (5).

Given the Δ length HMM data sequence $y_1, \ldots, y_\Delta$ of the Nth batch, the EM algorithm is an iterative algorithm for computing the MLE, we refer the reader to [19], [4] for details. The EM algorithm proceeds as follows (for notational convenience we have omitted the voltage v and subscript N in the notation below, i.e. , $\hat{\lambda}_N(v)$ is written as $\hat{\lambda}$):

Algorithm 3: [EM algorithm for MLE computation of a HMM (`Niter` iterations)]

- `Step 0:` (Initialization.) At iteration $J = 0$, initialize parameters $\lambda^{(0)} = \{A^{(0)}, I^{(0)}, \sigma^{2\,(0)}\}$.
- `Step 1:` (Expectation Step.) At iteration J, set $\lambda = \lambda^{(J)}$, and compute the following statistics.

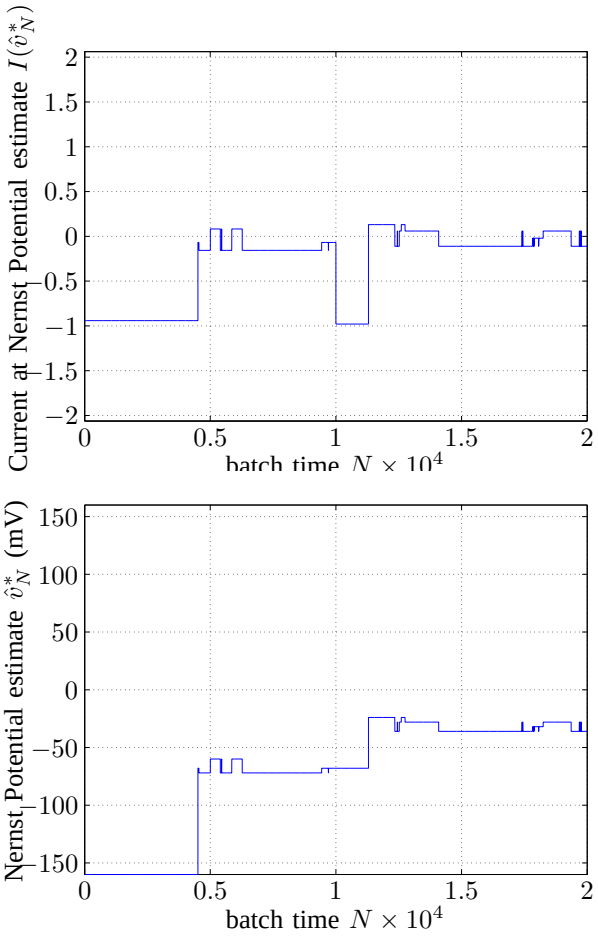

Fig. 9. Performance of Adaptive Discrete Stochastic Approximation Algorithm 2 for tracking time varying Nernst potential of Example 2 in Sec.V

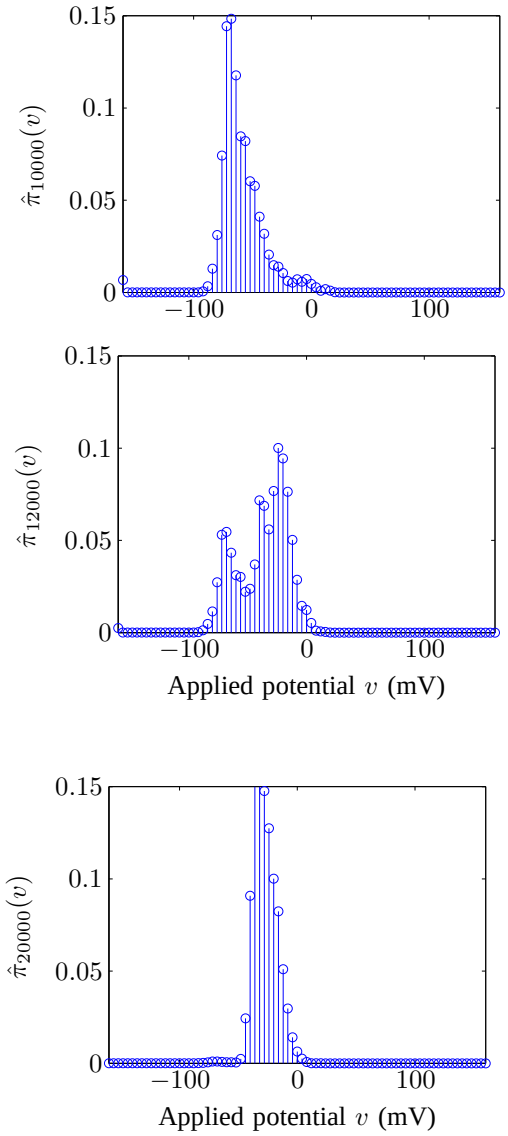

Fig. 10. Snapshots of Occupation probabilities of adaptive Algorithm 2 at batch-times $N = 10,000$, 12,000 and 20,000 for Example 2 in Sec.V. These snapshots illustrate how Algorithm 2 adapts to the change in Nernst potential at batch-time $N = 10,000$ and learns the new Nernst potential.

Forward recursion: $\alpha_1(j) = 1/3$,

$$\alpha_{k+1}(j) = \sum_{i=1}^{3} \alpha_k(i) a_{ij} b_j(y_{k+1}), \quad k = 1, \ldots, \Delta \quad (25)$$

Backward recursion: $\beta_\Delta(i) = 1$ and

$$\beta_{k+1}(i) = \sum_{j=1}^{3} \beta_k(i) a_{ij} b_j(y_{k+1}), \quad k = \Delta, \ldots, 1.$$

$$\gamma_k(i) = \frac{\alpha_k(i)\beta_k(i)}{\sum_{i=1}^{3} \alpha_k(i)\beta_k(i)}, \quad k = 1, \ldots, \Delta$$

$$\zeta_k(i,j) = \frac{\alpha_k(i) a_{ij} \beta_{k+1}(j) b_j(y_{k+1})}{\sum_{i=1}^{3} \alpha_k(i) a_{ij} \beta_{k+1}(j) b_j(y_{k+1})}, \quad k = 1, \ldots, \Delta$$

Here $b_j(y_{k+1})$ denotes the HMM observation probability density function: $b_j(y_{k+1}) \stackrel{\triangle}{=} p(y_{k+1}|i_{k+1} = l_j)$

$$b_j(y_{k+1}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_{k+1} - l_j)^2}{2\sigma^2}\right)$$

$l_j \in \{0_{\mathrm{g}}, 0_{\mathrm{b}}, I(v)\}$

- `Step 2:` (Maximization Step.) At iteration J, update parameter estimate $\lambda^{(J+1)}$ as

$$a_{ij}^{(J+1)} = \frac{\sum_{k=1}^{\Delta} \zeta_k(i,j)}{\sum + k = 1^{\Delta} \gamma_k(i)}, \quad i,j \in \{1,2,3\}$$

$$I^{(J+1)} = \frac{1}{\Delta} \sum_{k=1}^{\Delta} \gamma_k(3) y_k$$

$$\sigma^{2^{(J+1)}} = \frac{1}{\Delta} \sum_{k=1}^{\Delta} \Big((\gamma_k(1) + \gamma_k(2)) y_k^2 + \gamma_k(3)(y_k - I^{(J+1)})^2 \Big)$$

- `Step 3:` (Termination.) If $I <$ `Niter`, set $J \rightarrow J+1$ and go to Step 1.

  Else set MLE estimate $\hat{\lambda}(v) = \lambda^{(J)}$.

In the Expectation (E) step, α_k is called the *forward* variable and β_k is called the *backward* variable [19]. The update equations in the Maximization (M) step are collectively called the *Baum Welch* equations. The nice property of the EM algorithm is that the sequence of estimates $\lambda^{(J)}$ generated by the EM algorithm yield mononotically increasing likelihoods until converges to the MLE.

In actual implementation it is necessary to scale the variables α_k and β_k to avoid numerical underflow, see [19] for details. Finally the initialization α_1 in (25) above is any arbitrary positive vector – α_k is geometrically ergodic and forgets its initial condition geometrically fast.

REFERENCES

[1] E. Neher and B. Sakmann, "Single-channel currents recorded from membrane of denervated frog muscle fibres," *Nature*, vol. 260, pp. 799–802, 1976.

[2] O. Hamill, A. Marty, E. Neher, B. Sakmann, and F. Sigworth, "Improved patch-clamp techniques for high-resoluton current recodring from cells and cell-free membrane patches," *Pflüders Arch.–Eur. J. Physiol.*, vol. 391, pp. 85–100, 1981.

[3] S. Chung, J. Moore, L. Xia, and L. Premkumar, "Characterization of single channel currents using digital signal processing techniques based on hidden Markov models," *Proc. Phil. Trans. Roy. Soc. Lond. B*, vol. 329, pp. 256–285, 1990.

[4] S. Chung, V. Krishnamurthy, and J. Moore, "Adaptive processing techniques based on hidden Markov models for characterising very small channel currents buried in noise and deterministic interferences," *Proc. Phil. Trans. Roy. Soc. Lond. B*, vol. 334, pp. 357–384, 1991.

[5] L. Venkataramanan, R. Kuc, and F. Sigworth, "Identification of hidden Markov models for ion channel currents – part ii: Bandlimited, sampled data." *IEEE Trans. Signal Proc.*, vol. 48, no. 2, pp. 376–385, Feb. 2000.

[6] S. Andradottir, "A method for discrete stochastic optimization," *Management Science*, vol. 41, no. 12, pp. 1946–1961, 1995.

[7] ——, "Accelerating the convergence of random search methods for discrete stochastic optimization," *ACM Transactions on Modelling and Computer Simulation*, vol. 9, no. 4, pp. 349–380, Oct 1999.

[8] M. O'Mara, P. Barry, and S. Chung, "A model of the glycine receptor deduced from brownian dynamics studies," *Proceedings of the National Academy of Sciences U.S.A.*, vol. 100, pp. 4310–4315, 2003.

[9] B. Hille, *Ionic Channels of Excitable Membranes*, 3rd ed. Sunderland, MA.: Sinauer Associates, Inc., 2001.

[10] V. Krishnamurthy, J. Moore, and S. Chung, "Hidden Markov model signal processing in the presence of unknown deterministic interferences," *IEEE Transactions on Automatic Control*, vol. 38, no. 1, pp. 146–152, Janaury 1993.

[11] Y. Ephraim and N. Merhav, "Hidden Markov processes," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1518–1569, June 2002.

[12] P. Bickel, Y. Ritov, and T. Rydén, "Asymptotic normality of the maximum-likelihood estimator for general hidden Markov models," *Annals of Statistics*, vol. 26, pp. 1614–1635, 1998.

[13] G. Pflug, *Optimization of Stochastic Models: The Interface between simulation and Optimization*. Kluwer Academic Publishers, 1996.

[14] B. Leroux, "Maximum-likelihood estimation for hidden Markov models," *Stochastic Processes and its Applications*, vol. 40, pp. 127–143, 1992.

[15] J. Swisher, J. S, H, P. Hyden, and L. Schruben, "A survey of simulation optimization techniques and procedures," in *Proc. 2000 Winter Simulation Conference*, Orlando, Florida, 2000.

[16] D. Yan and H. Mukai, "Stochastic discrete optimization," *SIAM J. Control and Optimization*, vol. 30, no. 3, pp. 594–612, 1992.

[17] W. Gong, Y. Ho, and W. Zhai, "Stochastic comparison algorithm for discrete optimization with estimation," *SIAM Journal on Optimization*, vol. 10, no. 2, pp. 384–404, 1999.

[18] G. Yin, V. Krishnamurthy, and C. Ion, "Regime switching stochastic approximation algorithms with application to adaptive discrete stochastic optimization," *SIAM J. Control and Optimization*, 2003, submitted.

[19] M. James, V. Krishnamurthy, and F. LeGland, "Time discretization of continuous-time filters and smoothers for HMM parameter estimation," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 593–605, March 1996.

Vikram Krishnamurthy (SM'99) was born in 1966. He received his bachelor's degree in Electrical Engineering from the University of Auckland, New Zealand in 1988, and Ph.D. from the Australian National University, Canberra, in 1992. He is currently a professor and Canada Research Chair at the Department of Electrical Engineering, University of British Columbia, Vancouver, Canada. He is an adjunct professor at the Department of Electrical and Electronic Engineering, University of Melbourne, Australia. His research interests span several areas including stochastic scheduling and network optimization, time-series analysis, statistical signal processing and wireless telecommunications.

Dr. Krishnamurthy is currently an associate editor for *IEEE Transactions on Signal Processing* and *Systems and Control Letters*. He has served on the technical program committee of several conferences.

Shin Ho Chung is head of the biophysics group in the Research School of Physical Sciences and Engineering, Australian National University, Canberra, Australia. He received his B.Sc. from Stanford University and Ph.D. from Harvard University. His research is aimed at elucidating how membrane ion channels work.