Controlling spike timing and synchrony in oscillatory neurons

Tyler Stigen, Per Danzl, Jeff Moehlis and Theoden Netoff

J Neurophysiol 105:2074-2082, 2011. doi:10.1152/jn.00898.2011

J Neurophysiol 105: 2074–2082, 2011.
First published January 27, 2011; doi:10.1152/jn.00898.2011.

Controlling spike timing and synchrony in oscillatory neurons

Tyler Stigen,[1] Per Danzl,[2] Jeff Moehlis,[2] and Theoden Netoff[1]
[1]*Department of Biomedical Engineering, University of Minnesota, Minneapolis, Minnesota; and* [2]*Department of Mechanical Engineering, University of California, Santa Barbara, California*

Submitted 22 October 2010; accepted in final form 19 January 2011.

Stigen T, Danzl P, Moehlis J, Netoff T. Controlling spike timing and synchrony in oscillatory neurons. *J Neurophysiol* 105: 2074–2082, 2011. First published January 27, 2011; doi:10.1152/jn.00898.2011.— We describe an algorithm to control synchrony between two periodically firing neurons. The control scheme operates in real-time using a dynamic clamp platform. This algorithm is a low-impact stimulation method that brings the neurons toward the desired level of synchrony over the course of several neuron firing periods. As a proof of principle, we demonstrate the versatility of the algorithm using real-time conductance models and then show its performance with biological neurons of hippocampal region CA1 and entorhinal cortex.

synchrony controller; deep brain stimulation; dynamic clamp; phase-response curve; spike time controller

IN THIS ARTICLE we present a method of controlling a neuron via dynamic clamp to drive the neuron to fire with prescribed spike timing. The control is based on the functional relationship between a stimulus pulse amplitude, applied at a fixed phase of a periodically firing neuron, and the phase advance. This function can be inverted and used as a control function to determine what amplitude stimulus is needed to achieve a particular spike advance. This controller is made possible by a real-time dynamic clamp platform. We demonstrate two applications: the first is to make a neuron fire in a random, but preselected, pattern of interspike intervals (ISI); the second is to make the neuron phase lock to a periodic system (e.g., another periodically firing neuron). The ability to control spike timing and synchrony may play an important role in treatment of many neurological diseases. This work is primarily motivated by the importance of neuronal synchrony in Parkinson's Disease, epilepsy, and essential tremor where it is thought that synchronous neuronal activity is critical to the pathological activity (Benabid et al. 2009; Fisher et al. 2010; Kühn et al. 2009). Moreover, this controller may be useful in developing closed-loop stimulation protocols for deep brain stimulation (DBS) to help synchronize or desynchronize a population using DBS pulses. This algorithm could also be used to control spike timing in a central pattern generator to generate a motor pattern.

Previously, we have demonstrated that a proportional-integral (PI) controller can be used to maintain a neuron at a stable ISI by adjusting the constant current applied to the neuron (Miranda and Netoff, in press). The results presented focus on controlling the neuron around a changing ISI. The basis for the event-based control method proposed is novel and different from the PI controller.

We first describe how the controller works. We then demonstrate the controller for a model neuron, then in a biological neuron. Next, we show how the controller can be extended to a Leader-Follower control paradigm where a neuron (the follower) is made to phase lock with another periodic system (the leader). Finally, we investigate how noise, frequency difference between the leader and follower, and the phase offset between them affects the phase locking between the two.

METHODS

Biological Preparation

Long-Evans rats age postnatal *day 14–21* were deeply anesthetized using isoflurane. The brain was extracted and bathed in a chilled artificial cerebral spinal fluid (aCSF; composition in mM: 124 NaCl, 2KCl, $2MgSO_4$,1.25 NaH_2PO_4, 2 $CaCl_2$, 26 $NaHCO_3$, and 10 D-glucose at pH 7.4, 295 mosM) (1). Transverse slices of the ventral horn of the hippocampal region were sectioned 400 μm thick on a vibratome (Leica Microsystems, Bannockburn, IL). Neurons were visualized using differential interference contrast optics (Olympus, Center Valley, PA). Whole cell patch-clamp recordings were performed in the CA1 region of hippocampus and medial entorhinal cortex (MEC) using both pyramidal and stellate cells. Borosilicate capillary pipettes were pulled to 8 MΩ and filled with intracellular recording fluid (ICF; composition in mM: 120 K-glucose, 10 HEPES, 1 EGTA, 20 KCl, 2 $MgCl_2$, 2 Na_2ATP, and 0.25 Na_3GTP at pH 7.3, 290 mosM). The neuron's membrane potential was amplified and low-pass filtered at 2.4 kHz (Axon 700B; Molecular Devices, Sunnyvale, CA) and digitized on a real-time Linux computer (NiDAQ 6259; National Instruments, Austin TX) . The data was recorded using the RTXI system (described below). Only neurons that required holding currents less than -300 pA to maintain a resting potential of -65 mV were utilized. In addition, only periodically firing neurons were used because the control algorithm requires a well-defined phase to determine when to apply the stimulus pulse. All experiments were conducted as approved by the University of Minnesota Institutional Animal Care and Use Committee.

Dynamic Clamp

The dynamic clamp allows low-latency closed-loop experiments by interfacing a data acquisition card (DAQ) to a patch-clamp amplifier (Dorval et al. 2001). We use the Real-Time eXperiment Interface (RTXI; rtxi.org), which is available online. RTXI operates on the Real-Time Application Interface real-time Linux nanokernel (RTAI; rtai.org). RTXI can be used with many different DAQ cards through the Comedi project (comedi.org). The modules used in this project (as well as many others) are freely available through the RTXI software repository. RTXI can provide closed-loop control up to 100 kHz, and we did all experiments at 5 kHz with latency of one time step $=$ 0.2 ms and maximum jitter of $<$0.01 ms.

Address for reprint requests and other correspondence: T. Stigen, Dept. of Biomedical Engineering, Univ. of Minnesota, Minneapolis, MN 55455 (e-mail: tstigen@umn.edu).

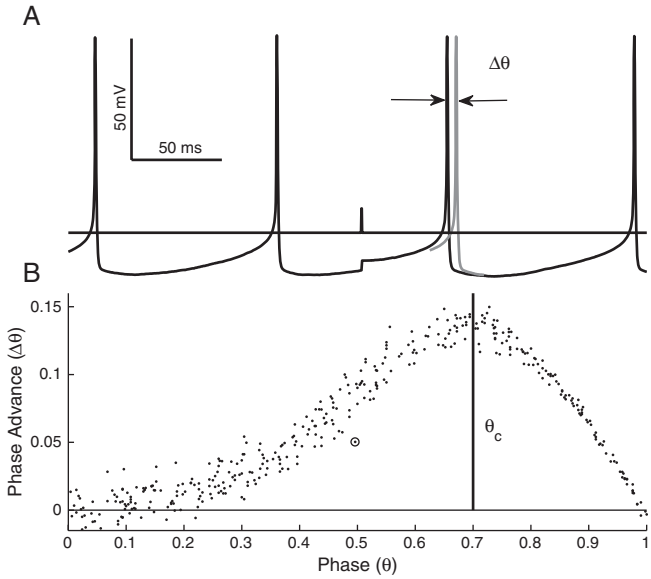

Fig. 1. Phase-dependent sensitivity. The phase-dependent sensitivity to stimulus is measured by injecting a stimulus into the neuron at a random phase in its period. *A*: plot of a neuron's membrane potential (solid), unperturbed voltage trace (shaded), and current pulses (offset). The stimulus is applied on every sixth cycle to allow the neuron to return to its natural period and to avoid higher order interactions from the neuron's response. The current pulse causes a phase advance of $\Delta\theta$. The response of this current pulse is shown in *B* as a circle. *B*: spike advance plotted against stimulus phase showing that the peak advance occurs at $\theta = 0.7 \equiv \theta_c$; stimulating at θ_c will yield the largest dynamic range.

Model Neuron

We use a conductance-based neuron model (Hodgkin and Huxley 1952) called GACell. It is based on the model from Golomb and Amitai (1997) and incorporates three potassium ($I_{\mathrm{Kdr}}, I_{\mathrm{KA}}, I_{\mathrm{K,slow}}$), two sodium ($I_{\mathrm{Na}}, I_{\mathrm{NaP}}$), and leak conductances ($I_{\mathrm{L}}, I_{\mathrm{app}}$):

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -I_{\mathrm{Na}}(V,h) - I_{\mathrm{NaP}}(V) - I_{\mathrm{Kdr}}(V,n) - I_{\mathrm{KA}}(V,b) - I_{\mathrm{K,slow}}(V,z) - I_{\mathrm{L}}(V) + I_{\mathrm{app}}.$$

GACell is a model of a pyramidal neuron and is designed with the parameters to allow for stable periodic firing. Noise in the model is simulated by adding a scaled gaussian white noise to the applied current. Noise current was applied at 5 kHz or 0.2 ms. The model was integrated at 100 kHz using an fourth-order Runge-Kutta integrator (Press et al. 2007). The lowest noise level (1×) is based on the coefficient of variation statistic. In this case, we measured the coefficient of variation (CV) of the ISI of a stable periodically firing CA1 pyramidal neuron, giving CV = 0.075. By trial and error, we determined what parameter values were needed in our white noise injection module to recreate approximately the same CV with the GA neuron model. We then repeated this process for 2×, 3×, and 4× multiples of the base CV: 0.15, 0.3, and 0.45, respectively.

RESULTS

Single Cell Spike Time Control

Stimulus phase selection. The spike time control method proposed is based on stimulating a periodically firing neuron at a given phase with a pulse where the amplitude is selected to make the neuron fire at a desired time.

Neurons exhibit a phase-dependent sensitivity to stimulus. Selecting a stimulation phase that falls in an area of high sensitivity will yield a larger range of control. We can locate the optimal phase by stimulating the neuron at different phases and measuring the spike advance. To avoid interactions between the stimuli, the neuron is only stimulated every sixth cycle. In Fig. 1, the spike advance is measured at different stimulus phases for the model neuron (Gutkin et al. 2005). The stimulus waveform was a square wave pulse with width 0.2 ms and amplitude 100 pA. We define phase as a fraction of the natural period in the interval $\theta \in (0, 1)$. It can be seen that the greatest advance occurs when the stimulus is applied at $\theta = 0.7$. It should also be noted that the variance of the response early in the phase is higher than later in the phase, meaning that stimulating later in the phase yields a response that is more reliable (Beverlin et al. 2011). In general, we find that the phase-response curves in pyramidal neurons in hippocampus and entorhinal cortex tend to peak around a phase of 0.6 and 0.7 phase as well (data not shown). Therefore, we have chosen the stimulus phase $\theta_c = 0.7$, which offers a balance between response variance and higher dynamic range.

Determining spike advance. After selecting the stimulus phase, θ_c, we characterize the phase advance as the stimulus amplitude is varied. Figure 2*A* shows the voltage trace from a neuron while being stimulated with a 0.2-ms current pulse, and Fig. 2*B* shows the phase advance as a function of the stimulus amplitude. To avoid interactions between the stimuli, the neuron is only stimulated every sixth cycle. We will assume in this controller that each stimulus does not have significant effect on subsequent cycles, but it is possible to make a controller that would account for these higher order effects (Talathi et al. 2009). The spike advance as a function of current pulse amplitude has a distinct sigmoidal shape. The upper limit (the maximum spike advance) is bounded by causality, where the stimulus immediately elicits a spike. The longest spike delay (negative spike advance) has no well-defined limit, but the variance increases as the interval gets longer with large negative current pulses. We fit the data using a sigmoidal function of the form

$$\Delta s(u) = A + \frac{B - A}{1 + e^{\frac{-u+C}{D}}} \equiv f_s(u), \qquad (1)$$

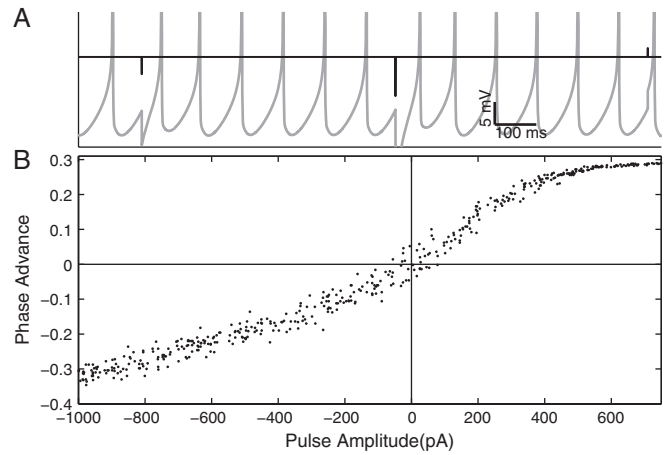

Fig. 2. Spike time advance curve. *A*: voltage trace of neuron (shaded) and current stimulus pulses (solid) at phase $= \theta_c$. *B*: spike advance plotted against amplitude of current pulse. The phase advance as a function of stimulus pulse amplitude has a sigmoidal shape. A negative phase advance corresponds to a delay in the spike timing.

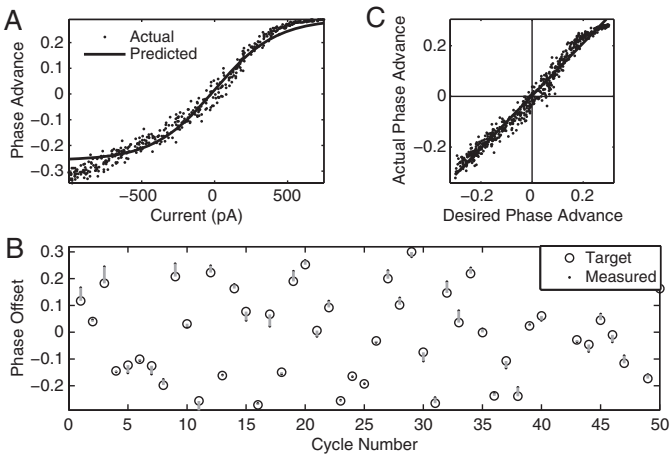

Fig. 3. Model neuron spike time control. *A*: the spike advance curve and the sigmoidal fit for a model neuron. White noise current is injected at a level comparable to that of a stable biological neuron. *B*: the time series of both the target phase offset (circles) and the measured phase offset (dots) for a model neuron. Shaded lines connect each pair for clarity. *C*: plot of the target phase offset against the measured phase offset for the model neuron. The line has a slope of unity and indicates exact control of spike timing.

where Δs is the desired phase advance; A, B, C, and D are constants; and u is the stimulus amplitude. A determines the maximum delay, B determines the maximum advance, C determines the inflection point of the sigmoid, and D determines the slope at the inflection point. The coefficients are determined by fitting this equation to the measured data to give the least squared error. We chose a sigmoid function for two reasons, first because the data are well fit by this function, to a first approximation, and second because the function is invertible. Although the sigmoid may not fit perfectly at the extremes of stimulus amplitude, this is generally outside of the working range of the controller. Most important is that the function fits the data well around the origin; we have found that even a linear function can serve well to model the cell's response if the stimulus range is limited enough.

For the control algorithm, we select a target Δs and invert this sigmoid to determine the necessary stimulus amplitude, using the function

$$u = C - D \cdot \log\left(\frac{B - A}{\Delta s - A} - 1\right) \equiv f_s^{-1}(\Delta s). \qquad (2)$$

The spike time advance curve fitting $f_s(u)$: $(u_{\min}, u_{\max}) \mapsto (\Delta s_{\min}, \Delta s_{\max})$, where, $|\Delta s_{\min}|$ corresponds to the maximum possible spike time delay and $\Delta s_{\max}$ corresponds to the maximum possible spike time advance. We cannot control the spike time advance beyond these values with a single stimulus per period.

Model neuron. With the Golomb-Amatai neuron model, a PI controller was used to drive the neuron to fire at 10 Hz. The phase of the neuron is based on the target ISI of the PI controller. We chose to stimulate the neuron at $\theta_c = 0.7$. We stimulated the neuron with current pulses of 0.2-ms width and amplitudes selected uniformly at random from −1 to 1 nA, and the spike advances were recorded and plotted against the stimulus amplitude, as shown in Fig. 3*A*. *Equation 1* was fit to the data to generate a spike advance curve. The control function, which determines the input amplitude necessary to make the neuron fire at a desired time, was generated by inverting *Eq. 1* to get *Eq. 2*. We then performed a validation test, where we generated random desired target phase advances in the range −0.3 to 0.3 and measured the spike advance the controller achieved, as shown in Fig. 3, *B* and *C*.

The correlation (R^2) between the desired ISI and the measured ISI, shown in Fig. 3, *A* and *B*, was used as a statistical measure of control efficacy. $R^2 = 1$ if all the variability in the neuron's ISI can be explained by the desired ISI (i.e., under perfect control). For the model neuron driven with noise, $R^2 = 0.99$, indicating that 99% of the neuron's variance could be attributed to the control algorithm.

Biological neuron. Using the method described above, the same process was performed using a pyramidal neuron in the CA1 region of the hippocampus. An example from one sample neuron is shown in Fig. 4. In this particular neuron, the current pulse range used to generate the spike advance curve was −7.5 to 2.5 nA. The controller was then used to control the neuron over a range of phase advances from −0.25 to 0.2. The current range used for control in the real neuron was larger than that used in the model for several reasons, including the resistance of the neuron, the quality of the patch seal, and the neuron's individual dynamics. Therefore, the coefficients for the controller need to be estimated for each neuron. The variance in the real neuron compared to our model is larger. For this neuron, $R^2 = 0.87$, indicating that even in a real neuron the spike timing could be perturbed reliably to make it fire in an arbitrary pattern. This experiment was repeated in 42 neurons with R^2 values ranging from 0.6 to 0.95.

Leader-Follower Control

With the ability to control spike timing, we extended the controller to create phase locking between two periodically firing neurons (or a periodic neuron and any other periodic system). One neuron is the leader, which is periodically firing and is uncontrolled, and the other neuron is the follower, which may have a slightly different period from the leader and is controllable. Our objective is to control the spike time of the follower to

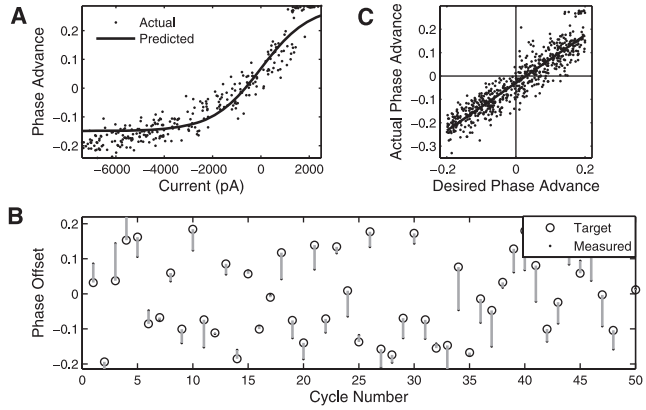

Fig. 4. Biological neuron spike time control. *A*: the spike advance curve and the fit control function for a CA1 pyramidal neuron. The intrinsic noise of a biological neuron may be different from our white noise model. *B*: the time series of both the target phase offset (circles) and the measured phase offset (dots) for a biological neuron. Shaded lines indicate the controller error and connect the actual spike time to the target spike time. *C*: plot of the target phase offset against the measured phase offset for the biological neuron. The solid line has unit slope and represents perfect control of spike timing.

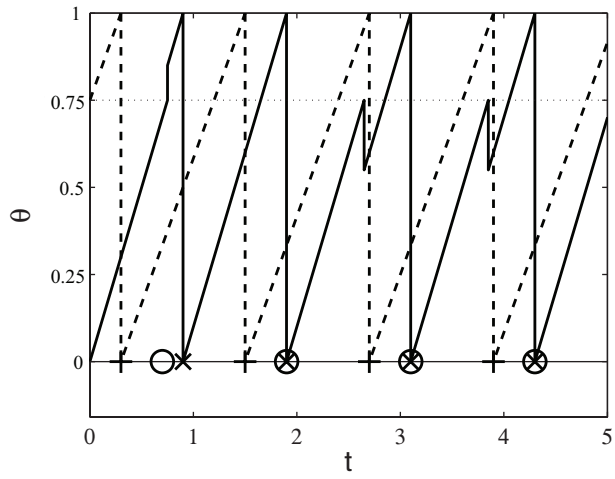

Fig. 5. Leader-Follower control algorithm. The follower phase, θ_s, is shown as the solid line with spike events as $\times$ markers. The leader phase, θ_m, is shown as the dashed line with $+$ markers. The target follower spike times (leader spike times offset by Δt_D) are shown by $\circ$ markers. The dotted line is the time the stimulus pulse is applied, here $\theta_c = 0.75$. In this example, the follower spikes converge to the desired phase offset (relative to the leader spikes) after 2 periods of control, and then the controller simply adjusts the period of the follower each time to match that of the leader.

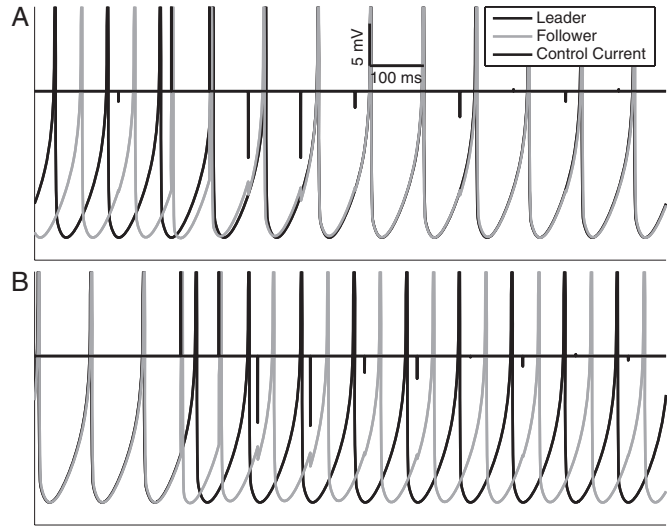

Fig. 6. Leader-Follower control in action. *A*: a time series of the leader follower algorithm controlling 2 model neurons to a target phase offset of 0, so that they are in phase. Leader neuron is solid, follower neuron is shaded, and the control is solid. *B*: a time series of the Leader-Follower algorithm controlling 2 model neurons to a target phase offset of 0.5, or so that they are in anti-phase. The algorithm can control to any arbitrary phase offset desired.

achieve a desired spike time difference or phase offset from the leader's while firing at a commensurate frequency as the leader.

Leader and follower as oscillators. The leader and follower neurons are considered as periodic oscillators where the phase linearly advances from 0 to 1 over the period. The target phase difference is identified as the control goal. Phase for each cell is definied by the target spike time difference divided by the target ISI obtained by the PI controller. When the follower reaches the stimulus phase θ_c, the spike time advance curve is used to correct the difference between the neuron's unperturbed next spike time and the target spike time, as shown in Fig. 5. Because of limitations on stimulus strength, it may not be possible to advance or delay the neuron to the target with one stimulus. Recall that there is a minimum $\Delta s_{\min}$ and maximum $\Delta s_{\max}$ spike advance for a finite current amplitude range. If the neuron cannot be stimulated in a single cycle to achieve the target phase offset, then we may need multiple cycles to achieve the target offset. The Leader-Follower controller determines whether the maximum advance or delay will bring the neuron closer to the target phase and then applies that stimulus. The details of this algorithm are in the APPENDIX.

In Fig. 6, two scenarios of the Leader-Follower control are illustrated: controlling from anti-phase to in-phase (*A*) and in-phase to anti-phase (*B*). In both cases, the controller applies current pulses that rapidly bring the two neurons to the desired phase offset, and then the controller continues to apply small current pulses to maintain the phase locking on subsequent cycles. Once the controller is activated, the desired phase offset is achieved in one or two cycles. However, if the difference in the natural frequencies is greater than the acceptable range of the controller, then control will be intermittent and there will be considerable phase slipping.

Model Neuron. We first applied the Leader-Follower controller to a model neuron (Golomb-Amatai model) with baseline ($1\times$, CV $= 0.075$) noise. Both leader and follower neurons had a period of 100 ms, and the target phase offset was 0.5 (anti-phase). The phase offset between the leader and follower was recorded for 3,000 spikes. Figure 7 shows the histogram of the phase offset for a single 3,000-spike trial. The average spike time difference was $\mu = 50.89$ ms with a standard deviation of $\sigma = 2.85$ ms. Figure 7, *inset*, is a polar plot of the histogram in polar coordinates. As a measure of the controller's efficacy, we treat each phase offset as a unit vector in polar coordinates, $e^{2\pi\theta_i}$ where the angle is determined by the phase difference θ_i on spike i, and $j = \sqrt{-1}$. The average phase difference is measured as the normalized sum of the unit vectors $\hat{\theta} = \frac{1}{N}\sum_i e^{2\pi\theta_i}$. To

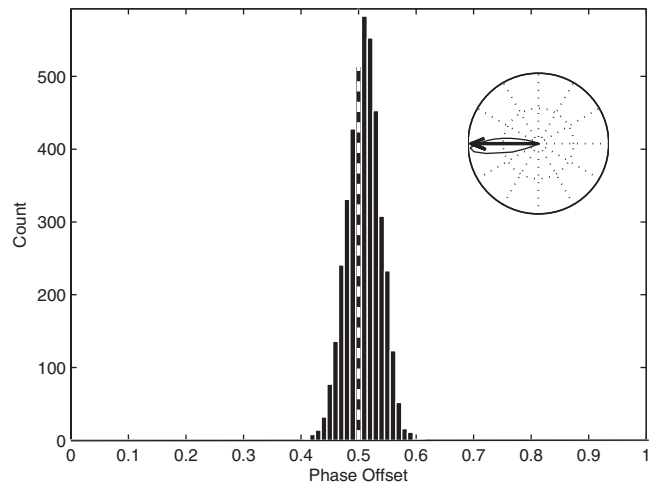

Fig. 7. Leader-Follower control in model neuron. This histogram shows the distribution of phase offset between the leader and follower spike timing. This is a model neuron with $1\times$ noise injection. The targeted phase offset is indicated by the dashed line. The *inset* image is a normalization of this distribution in polar form. The arrow indicates the desired phase offset. In this example, $\mu = 50.89$ ms, $\sigma = 2.85$ ms, and $E = 0.98$.

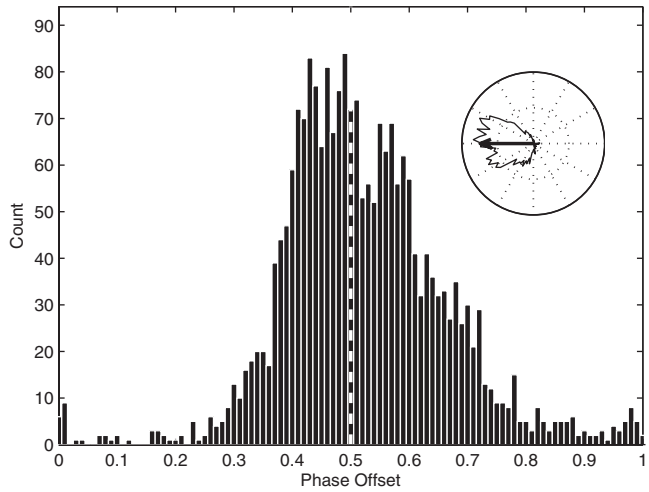

Fig. 8. Leader-Follower control in biological neuron. This histogram shows the distribution of phase offset between the leader and follower spike timing. The leader is a model neuron, and the follower is a pyramidal neuron from CA1. The targeted phase offset is indicated by the dashed line. The *inset* image is a normalization of this distribution in polar form. The arrow indicates the desired phase offset. In this example, $\mu = 52.21$ ms, $\sigma = 14.24$ ms, and $E = 0.69$.

determine the efficacy, E, of the controller, we calculate the vector correlation between the average phase and the target phase. This is the dot product between the average phase and the target phase $E = \hat{\theta} \cdot e^{2\pi\theta_d}$, where θ_d is the target phase (indicated as a dark solid arrow in the polar plot). Empirically, we say that there is poor control if the correlation in the range of 0 to 0.5, moderate control in the range 0.5 to 0.75, and high control if it is greater than 0.75. For the model neuron the vector correlation is 0.98, indicating high control.

Biological neuron. We tested the Leader-Follower control using pyramidal neurons in the CA1 region of hippocampus. Recording from living neurons presents challenges that often limit the ability to maintain long-term stable recordings, such as dramatic changes in cell behavior, fatigue, and death. The leader neuron is a periodically firing model neuron, and the follower is the patch-clamped neuron. Both leader and follower had a period of 100 ms. The target phase offset was 0.5, corresponding to anti-phase locking. We recorded the phase offset between the leader and follower over 2,267 spikes, and the histogram of the data is shown in Fig. 8. The phase offset distribution has a mean $\mu = 52.21$ ms and standard deviation of $\sigma = 14.24$ ms. In the real neuron, the control was moderate ($E = 0.69$) due to the high coefficient of variation with respect to the model neuron.

Algorithm Performance

Using the model neuron, we tested the robustness of the controller as we varied different experimental parameters. We tested the effect of noise amplitude in both leader and follower, the effect of the target phase offset, and the effect of frequency mismatch between the leader and follower.

Effect of noise. In Fig. 9, we illustrate the effect of noise in both the leader and follower on the efficacy of control. The effect of noise on the performance of the controller was measured in two configurations: periodic leader with noisy follower, and noisy leader with noisy follower. For the noisy leader with noisy follower configuration, independent noise was applied to each. Results from these simulations were compared with those from a sham configuration, where no current was applied to the follower. In all configurations, the leader and follower have ISI $= 100$ ms. The target phase offset was set at 0.25 for all configurations. We recorded the phase offset for 10,000 spike events for each configuration. The controller efficacy is measured by vector correlation as described above. For each of the sham configurations, the efficacy is 0, as one would expect. For the periodic leader with noisy follower configuration with noise ranging from $1\times$ to $4\times$, the efficacy decreased as noise increased, and vector correlations were 0.98, 0.89, 0.68, and 0.39, respectively. For the noisy leader with noisy follower configuration, the efficacy was lower, with vector correlations 0.96, 0.79, 0.53, and 0.28, respectively. It is to be expected that increasing noise decreases control efficacy, since the ISI has a larger variance, which makes it more difficult for the controller to accurately predict future control intervals.

Effect of phase offset. In Fig. 10, we illustrate the phase offset independence of this algorithm with noise with a periodic leader and noisy follower. Both leader and follower had a period of 100 ms. We measured the efficacy of the controller over a range of phase offset values from 0 to 0.95 for 3,000 spikes. These values cover intermediate values between in-phase (0 and 1) and anti-phase (0.5) solutions. These values were chosen to demonstrate that there is no bias of the controller to the desired phase offset. The efficacy did not significantly change with relation to the spike time offset, but the average vector correlation decreased with noise, ranging from $1\times$ to $4\times$ noise as 0.98, 0.82, 0.61, and 0.23 (results were not significantly different from control around phase 0.25 as described above, with differences attributed to shorter simulations).

Effect of ISI mismatching. In Fig. 11, we illustrate the effect of mismatched ISIs on the algorithm. We varied the period of the noiseless leader neuron from 70 to 130 ms while fixing the

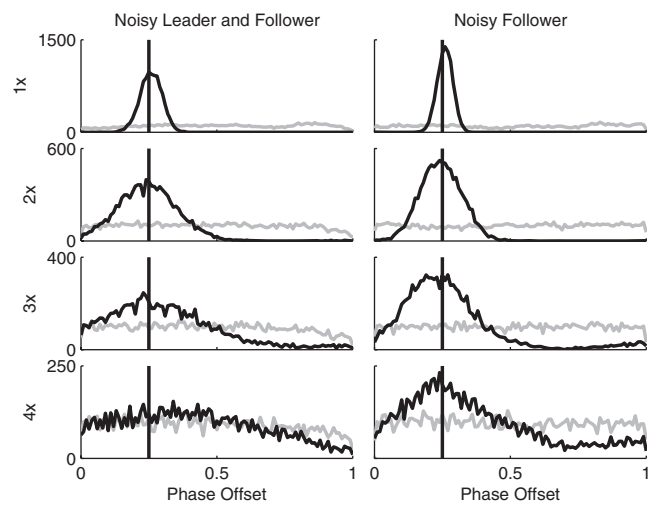

Fig. 9. Effect of noise on Leader-Follower control in model neuron. Shown are histograms of the phase offset between the leader and follower spike timing under $1\times$, $2\times$, $3\times$, and $4\times$ noise levels. Each plot shows the distribution with the control running as a sham with no current applied to the follower (shaded) and in an operational state (solid). The target phase offset is indicated with a vertical line.

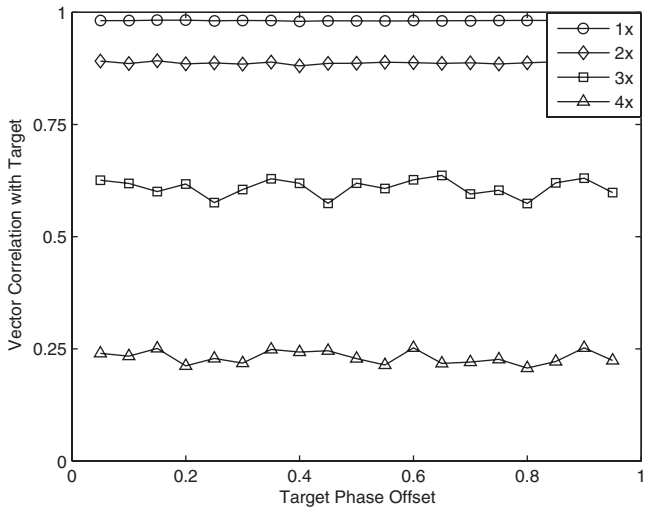

Fig. 10. Effect of target phase offset on control in model neuron. Shown is a plot of the vector correlation of the phase offset distribution vs. the target phase offset. The vector correlation at each target phase offset was measured in a model neuron with 4 noise levels.

period of the follower neuron at 100 ms. The controller maximum advance and delay allowed was 30 ms. The controller was rather robust until the phase offset approached the maximum of range of the controller, and then it dropped off sharply. With higher noise values, the range of control is decreased. We observed phase locking again when the ratio between the leader and follower approached rational ratios, but the behavior was complex and beyond the scope of this report. It is interesting to note that these curves are asymmetric and become more asymmetric as noise increases. This is because phase slipping does not occur as frequently when the leader period is shorter. Since the algorithm is triggered on the follower spike event, when the leader period is shorter, the next leader spike event is normally within the control range. This is not the case when the leader period is longer.

DISCUSSION

In this report, we have demonstrated a method by which a neuron can be controlled around any arbitrary pattern of admissible ISIs. This controller works by stimulating a periodically firing neuron at a selected phase. A model of the neuron's dynamics is generated by fitting a sigmoidal function to the phase advance vs. amplitude data. The control function is then calculated by inverting this function to calculate the necessary stimulus amplitude to achieve a spike at the desired time. In practice, this controller could control nearly 90% of the variance in a hippocampal CA1 pyramidal neuron in vitro. The algorithm was extended to phase lock two neurons at an arbitrary phase offset. In spike timing control and phase locking experiments, controller performance was robust to high levels of noise. In the phase locking experiments, the controller could maintain phase locking even with large differences in the natural frequency of the neurons being controlled.

Spike timing control works best when the neuron's response to the stimulus is reliable and the variance of the ISIs is low. The more accurately the spike times are known and the more reproducible the spike advance is, then the more accurately the controller can achieve the target spike time offsets. Control is best in neurons with a greater dynamic range-to-noise ratio. If a neuron's dynamic range is small, the spike advance can be lost in the noise. In terms of cell viability, a high current-to-spike advance gain is beneficial. Generally, cells with a high gain will be able to stay under control longer before they fail. This might be mitigated by using charge balanced stimulus waveforms.

The control will work better when the spike advance data is well fit by a sigmoid of the form in *Eq. 1*. For small perturbations, we have used a linear fit to the spike advance as a controller with a great deal of success. When larger perturbations are used, nonlinearities in the response begin to emerge. In this report, we have proposed a sigmoidal function to fit the data, but any invertible function fit to the data could be used to correct for these nonlinearities. In general, because neurons are noisy, even a rough fit to the spike advance curve provides sufficient accuracy for control.

In nearly all neurons, the range of spike time delay is much larger than the range of spike time advance. This is due to the selection of the stimulation phase toward the end of the neuron's period, but less trivially, it is because the delay is not bounded by causality. Spike time delay in a neuron is limited only by the maximum current that can be safely injected into the cell. Therefore, we found that for control cases where the mean ISI lengthening is tolerable, the control algorithm can control a much larger dynamical range than when the delays need to be balanced by advances to keep the mean ISI the same as the unperturbed case.

We found that the coefficients of the controller derived from the spike time advance curve were remarkably similar across neurons. This may indicate that the controller optimized for one neuron in a population may work, perhaps suboptimally, for most neurons in the population. If this is the case, it is possible that this controller may be used to control a population of neurons.

In the Leader-Follower algorithm, the stimulus can be applied every cycle, which can lead to significant, but temporary, changes in the average ISI of the follower. In cases when we had a

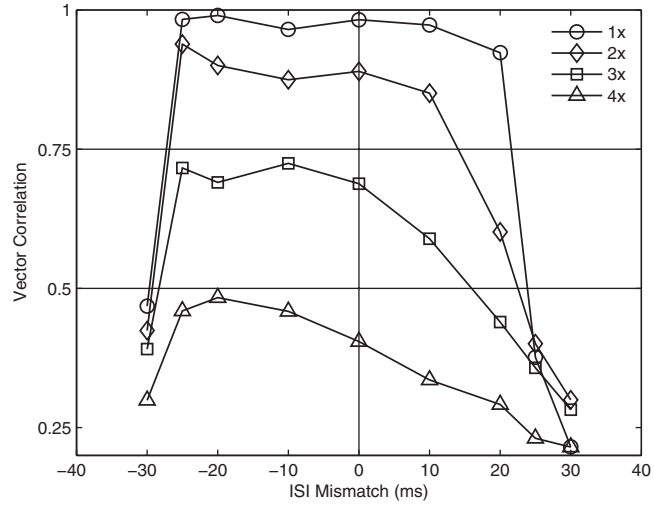

Fig. 11. Effect of interspike interval (ISI) mismatch on control in model neuron. Shown is a plot of the vector correlation of the phase offset distribution with the target phase offset versus the follower neuron ISI. The leader period was fixed at 100 ms, and the follower period was varied between 70 and 130 ms.

controller maintaining the follower's ISI using a PI controller, this could result in competition between the two controllers, resulting in decreased controller performance. Balancing the parameters of the PI controller to correct for slow drift in the ISI but not to respond to transient changes caused by the Leader-Follower controller corrects for this problem.

This framework can be extended to more complicated variants of this problem. Multiple instances of this control system could be implemented with any number of follower neurons, each with their own phase offset. By setting each follower i's phase offset $\Delta t_{D,i} = 0$, we can control to synchrony, or by setting $\Delta t_{D,i} = (i-1)/T_m$, we can desynchronize them.

The algorithm assumes independence of stimuli, and this may not be strictly true. Stimulation on two consecutive periods may introduce some small higher order responses. In general, ignoring these effects will introduce errors, but they are small relative to the inherent noise of the neuron. However, it is possible to take these effects into account and in theory improve the controller (Talathi et al. 2009).

This work is a proof of principle. This control algorithm may have applications in treating neurological diseases characterized by pathological synchronization, such as Parkinson's Diseases, epilepsy, and essential tremor. The controller could be used to divide a synchronous population into subgroups to disrupt pathological synchrony of the network, in an approach similar to coordinated resetting (Tass et al. 2009) or phase desynchronization (Danzl et al. 2009). This algorithm may be used in a closed-loop DBS to determine the subthreshold stimulus based on the physiological response to "gently" perturb the neuron toward the target phase offset. This may increase efficacy and reduce power consumption.

APPENDIX

We present the details of the Leader-Follower spike timing control algorithm. Both the leader and follower neurons are modeled as simple oscillators. The leader, having a natural frequency ω_m (in Hz) and initial phase $\theta_{m,0}$, can be described by

$$\text{Leader neuron} \quad \dot{\theta}_m = \omega_m, \quad \theta_m(0) = \theta_{m,0},$$

and the follower having a natural frequency ω_s, and initial phase $\theta_{s,0}$, can be described by

$$\text{Follower neuron} \quad \dot{\theta}_s = \omega_s + Z(\theta_c, u)u, \quad \theta_s(0) = 0.$$

Before we describe the algorithm itself, we discuss the fundamental limitations on the relationship between the natural periods of the leader and the follower neurons. These limitations are due to the fact that the control magnitude is finite and bounded, and the limitations apply to any such algorithm, not just the one presented in this report. The variables $\Delta s_{\max}$ and $|\Delta s_{\min}|$ are the maximum spike time advance and delay under the maximum control magnitudes $u_{\max}$ and $u_{\min}$. These values are critical because they determine the maximum possible discrepancy between the period of the leader and the follower, which must satisfy

$$T_s - \Delta s_{\max} \le T_m \le T_s + |\Delta s_{\min}|, \tag{3}$$

where $T_m = 1/\omega_m$ is the natural period of the leader neuron and $T_s = 1/\omega_s$ is the natural period of the follower neuron. The challenge is the fact that we cannot achieve any desired advance, only advances within the interval $(\Delta s_{\min}, \Delta s_{\max})$.

Preprocessing

Before the control is activated, the control system calculates some quantities that will be used in the online algorithm. The first is i_m^*, which is the maximum number of follower control iterations necessary to hit a desired spike timing:

$$i_m^* = \left\lceil \frac{T_m}{\Delta s_{\max} + |\Delta s_{\min}|} \right\rceil \tag{4}$$

where $\lceil \cdot \rceil$ is the round-up operator. The rationale behind this claim is that each target spike time will be T_m time units away from the next, and each time the control is activated (at $t = t_0$, the spike of the follower neuron), the control can adjust the timing of the next spike to be anywhere in the time interval $(t_0 + T_s - \Delta s_{\max}, t_0 + T_s + |\Delta s_{\min}|)$. So in n control periods, we can adjust the timing of the nth follower spike to be anywhere in the interval $(t_0 + nT_s - n\Delta s_{\max}, t_0 + nT_s + n|\Delta s_{\min}|)$. For sufficiently large $n \le i_m^*$, $n(\Delta s_{\max} + |\Delta s_{\min}|) \ge T_m$, which means the control window is larger than the period of the target spikes, a situation that guarantees success in hitting a single target spike. As stated before, once synchrony to a single target spike is achieved, we then take the difference of the interspike intervals $T_s - T_m$ as our desired spike advance and maintain this phase offset indefinitely.

Event-Based Algorithm

At each event (follower spike), we need to determine whether to command the maximum advance, maximum delay, or some intermediate spike advance. This task is accomplished by two nested algorithms: **event_control**, which is the high-level driver; and **spike_advance**, which is called to calculate the desired spike time advance each control period. We assume that our implementation-level control system tracks current time in the variable t and the most recent spike time of the leader neuron in the variable t_{last}.

Algorithm 1 event_control(t, t_{last})

1: $t_0 = t$
2: $\theta_m(t_0) = \frac{(t_0 - t_{\text{last}})}{T_m}$
3: $\Delta s =$ **spike_advance**$(t_0, \theta_m(t_0))$
4: $u = f_s^{-1}(\Delta s)$
5: apply pulse with amplitude u at time $t_0 + \theta_c T_s$

Explanation: Algorithm 1 ***event_control***

The control system is triggered when the follower neuron spikes. At this instant, we sample the current time, t, and the time of the last recorded leader neuron spike, t_{last}.

Line 1: Assign the current value of t to the variable t_0. This will serve as a time offset for future calculations.

Line 2: Estimate the phase of the leader neuron based on its last recorded spike time.

Line 3: Call the **spike_advance** algorithm to determine the optimal follower spike time advance to command.

Line 4: Calculate the current amplitude needed for the stimulus to cause the desired spike advance.

Line 5: Apply the stimulus pulse when we expect the follower neuron's phase to be at the optimal stimulus point, θ_c.

Explanation: Algorithm 2 ***spike_advance***

This algorithm is called from the **event_control** algorithm and is used to compute the desired spike advance. This function would be trivial if we could command any arbitrary spike advance. Since we are restricted in our choice of spike advance to the range $(\Delta s_{\min}, \Delta s_{\max})$,

Algorithm 2 spike_advance$(t_0, \theta_m(t_0))$

1: initialize $i = 1$, term $= 0$
2: $\mathcal{T}_t = \left\{ \begin{array}{l} t_t = t_0 + [1 - \theta_m(t_0)]T_m + \Delta t_D + kT_m, \\ k \in \mathbb{Z}_{\geq 0} \quad | \quad t_0 < t_t < t_0 + i_m^* T_s \end{array} \right\}$
3: $\mathcal{I}_C = (t_0 + T_s - \Delta s_{\max}, t_0 + T_s + |\Delta s_{\min}|)$
4: **if** $\mathcal{T}_t \cap \mathcal{I}_C \neq \emptyset$ **then**
5: $\quad \Delta s = T_s - (\min\{\mathcal{T}_t \cap \mathcal{I}_C\} - t_0)$
6: **else**
7: $\quad i$++
8: $\quad$ **while** term $== 0$ **do**
9: $\quad\quad \mathcal{I}_A = (t_0 + iT_s - i\Delta s_{\max}, t_0 + iT_s)$
10: $\quad\quad \mathcal{I}_D = (t_0 + iT_s, t_0 + iT_s + i|\Delta s_{\min}|)$
11: $\quad\quad$ **if** $\mathcal{T}_t \cap \mathcal{I}_A \neq \emptyset$ **then**
12: $\quad\quad\quad \Delta s = \Delta s_{\max}$, term $= 1$
13: $\quad\quad$ **else if** $\mathcal{T}_t \cap \mathcal{I}_D \neq \emptyset$ **then**
14: $\quad\quad\quad \Delta s = \Delta s_{\min}$, term $= 1$
15: $\quad\quad$ **else**
16: $\quad\quad\quad i$++
17: $\quad\quad$ **end if**
18: $\quad$ **end while**
19: **end if**
20: **return** Δs

we must take care in deciding what to do if we cannot reach the first desired spike time in one round of control.

The dynamic information passed to this function includes the current follower spike time offset, t_0, and the estimated phase of the leader, $\theta_m(t_0)$.

Line 1: Initialize an integer counter, i, and a boolean flag, term.

Line 2: Calculate a set of target spike times, $\mathcal{T}_t$. These are based on when the leader neuron will spike next and what the desired time interval is between the leader and follower spikes, Δt_D. The set is truncated using our estimate of the maximum possible control periods, i_m^*, necessary to reach any T_m-periodic spike train, as calculated in *Eq. 4*.

Line 3: Calculate our first interval of control, $\mathcal{I}_C$. This represents the amount we can change the next follower spike time by applying up to the maximum advance or delay in this control period.

Line 4: Check to see if a target spike time lies within the first interval of control. If not, this will return $\emptyset$ (the null or empty set).

Line 5: If the intersection of the set of target spike times, $\mathcal{T}_t$, and the first interval of control, $\mathcal{I}_C$, is nonempty, choose the first target spike time in the intersection set and compute the necessary spike advance to apply this control period. At this point, the *Line 4* **if** statement is done; proceed to *Line 20*.

Line 6: If the intersection of $\mathcal{T}_t$ and $\mathcal{I}_C$ is empty, we cannot achieve our control objective in one control period. We must now enter into an iterative portion of the algorithm that will decide whether to apply the maximum spike advance $\Delta s_{\max}$ or maximum spike delay $\Delta s_{\min}$.

Line 7: Increment the counter, i. This counter tracks how many (hypothetical) control periods must be used to result in enough cumulative spike advance or delay for a follower spike to coincide with a leader spike.

Line 8: This **while** loop runs until we have found how many periods of successive maximal advances or delays are necessary to achieve our objective, codified by setting the terminal flag, term, to 1 (true). This **while** loop is guaranteed to terminate within i_m^* iterations.

Line 9: Compute the ith advance control interval. This represents the interval of control that can be achieved in i control periods by applying the maximum advance, $\Delta s_{\max}$, each time.

Line 10: Compute the ith delay control interval. This represents the interval of control that can be achieved in i control periods by applying the maximum delay, $\Delta s_{\min}$, each time.

Line 11: Check to see if any target spike times lie within this ith interval of maximally advancing control, $\mathcal{I}_A$.

Line 12: If the intersection of the set of target spike times and the ith interval of maximally advancing of control is nonempty, this means that by continuing to apply to maximum spike advance, $\Delta s_{\max}$, we can eventually achieve our control objective. Set $\Delta s = \Delta s_{\max}$ and switch the boolean termination flag, term, to 1 (this will get us out of the **while** loop).

Line 13: Check to see if any target spike times lie within this ith interval of maximally delaying control, $\mathcal{I}_D$.

Line 14: If the intersection of the set of target spike times and the ith interval of maximally delaying of control is nonempty, this means that by continuing to apply to maximum spike delay, $\Delta s_{\min}$, we can eventually achieve our control objective. Set $\Delta s = \Delta s_{\min}$ and switch the boolean termination flag, term, to 1 (this will get us out of the **while** loop).

Lines 15 and 16: If both interval intersection checks return the null set, increment the counter i by one and return to *Line 8*.

Line 20: The algorithm will return the desired spike advance Δs to the **event_control** algorithm that called it.

GRANTS

This work is supported by a National Institutes of Health National Research Service Award Training Fellowship, National Science Foundation Grants NSF-0547606, NSF-1000678, and NSF-0954797, and a National Science Foundation Integrative Graduate Education and Research Training Fellowship.

DISCLOSURES

No conflicts of interest, financial or otherwise, are declared by the authors.

REFERENCES

Benabid AL, Chabardes S, Mitrofanis J, Pollak P. Deep brain stimulation of the subthalamic nucleus for the treatment of Parkinson's disease. *Lancet Neurol* 8: 67–81, 2009.

Beverlin B, Ermentrout GB, Troyer T, Netoff T. The variance of phase-resetting curves. *J Comput Neurosci*. In press.

Danzl P, Hespanha J, Moehlis J. Event-based minimum-time control of oscillatory neuron models: phase randomization, maximal spike rate increase, and desynchronization. *Biol Cybern* 101: 387–399, 2009.

Dorval AD, Christini DJ, White JA. Real-time linux dynamic clamp: a fast and flexible way to contruct virtual ion channels in living cells. *Ann Biomed Eng* 29: 897–907, 2001.

Fisher R, Salanova V, Witt T, Worth R, Henry T, Gross R, Oommen K, Osorio I, Nazzaro J, Labar D, Kaplitt M, Sperling M, Sandok E, Neal J, Handforth A, Stern J, DeSalles A, Chung S, Shetter A, Bergen D, Bakay R, Henderson J, French J, Baltuch G, Rosenfeld W, Youkilis A, Marks W, Garcia P, Barbaro N, Fountain N, Bazil C, Goodman R, McKhann G, Krishnamurthy KB, Papavassiliou S, Epstein C, Pollard J, Tonder L, Grebin J, Coffey R, Graves N, SANTE Study Group. Electrical stimulation of the anterior nucleus of thalamus for treatment of refractory epilepsy. *Epilepsia* 51: 899–908, 2010.

Golomb D, Amitai Y. Propagating neuronal discharges in neocortical slices: compuational and experimental study. *J Neurophysiol* 78: 1199–1211, 1997.

Gutkin BS, Ermentrout GB, Reyes AD. Phase-response curves give the responses of neurons to transient inputs. *J Neurophysiol* 94: 1623–1635, 2005.

Hodgkin AL, Huxley AF. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol* 117: 500–544, 1952.

Kühn AA, Tsui A, Aziz T, Ray N, Brcke C, Kupsch A, Schneider GH, Brown P. Pathological synchronisation in the subthalamic nucleus of patients with Parkinson's disease relates to both bradykinesia and rigidity. *Exp Neurol* 215: 380–387, 2009.

Miranda O, Netoff T. PI control for neuronal firing. *J Neural Eng.* In press.
Moyer JR, Brown TH. Methods for whole-cell recording from visually preselected neurons of perirhinal cortex in brain slices from young and aging rats. *J Neurosci Methods* 86: 35–54, 1998.
Press W, Flannery B, Teukolsky S, Vetterling W. *Numerical Recipes in C* (3rd ed.). New York: Cambridge University Press, 2007.
Talathi SS, Hwang DU, Miliotis A, Carney PR, Ditto WL. Predicting synchrony in heterogeneous pulse coupled oscillators. *Phys Rev E Stat Nonlin Soft Matter Phys* 80: 021908, 2009.
Tass PA, Silchenko AN, Hauptmann C, Barnikol UB, Speckmann BJ. Long-lasting desynchronization in rat hippocampal slice induced by coordinated reset stimulation. *Phys Rev E Stat Nonlin Soft Matter Phys* 80: 011902, 2009.