Adaptive Wavetable Oscillators

Raman Arora and William Sethares

Abstract

An adaptive oscillator is a system that can lock on to a time-varying input signal, synchronizing its output to both the frequency and phase of the input. A wavetable oscillator generates a periodic output by indexing into a lookup table that stores a single period of the waveform. An adaptive wavetable oscillator (AWO) combines these two ideas in a technique which separates the periodic output waveform from the parameters that control the adaptation of the frequency and phase of the waveform. This separation is advantageous because it decouples the state of the oscillator from the dynamics of the adaptation, allowing the process of sychronization to be interpreted as a simple gradient optimization on a cost function. The oscillations remain stable over a large and easily described range of parameter values, and analysis of the synchronization can proceed along lines familiar from standard adaptive systems. Key issues in the design of AWOs are: the class of admissible inputs, the shape of the wavetable, the parameters that will be controlled, and the adaptive algorithm that adjusts the parameters. This paper examines these issues through analysis and simulation, focusing on conditions that achieve the desired synchronization between output and input.

EDICS: ASP-ANAL, ASP-APPL

Permission to publish this abstract separately is given.

Both authors are with the Department of Electrical and Computer Engineering, University of Wisconsin-Madison, Madison, WI 53706-1691 USA. 608-262-5669 sethares@ece.wisc.edu

I. Introduction

While there are many uses of fixed oscillators (those which operate at a specified fixed frequency), many important applications require greater flexibility. For example, in communication systems, the phase-locked loop generates a sinusoid that adjusts a local oscillator to match the phase and frequency of a sinusoidal carrier embedded in the received signal [2]. Similarly, the timing recovery problem requires that the phase and frequency of a locally generated signal (the pulse shape) be synchronized to the symbols of the baseband received signal [14]. In the active noise cancellation of engine noise, a locally generated oscillatory waveform must smoothly track the rotation of the engine based on an input consisting of n pulses per revolution. In the beat-tracking problem, a locally generated waveform attempts to match phase and period to follow the foot-tapping "beat" of a musical passage [9].

In such applications the oscillator must do more than just mark time, it must also respond to external events. The frequency and phase of the oscillator must be malleable, able to synchronizing to a variety of stimuli. In general, oscillators must generate a periodic waveform, have an input, be capable of synchronizing to the frequency (or period) and phase of the input, and be robust to noisy periods, missing events, and unexpected events. Rather than focusing on particular applications or particular classes of inputs, this paper focuses on the adaptation of the oscillator, the way that the parameters of the oscillator are adjusted so as to achieve synchronization with the input. The key idea is the use of Adaptive Wavetable Oscillators (AWOs), which separate the detailed shape of the periodic waveform from the control signals that specify the phase and frequency of the oscillation. This separation allows the evolution of the control signals to follow a simple gradient strategy on a cost function. Moreover, it is straightforward to tailor the AWO for use with any anticipated class of input waveform. The resulting algorithms are well behaved and easy to understand since gradient techniques are common to many areas of optimization.

This paper begins with a brief review of several basic oscillators in Sect. II. The various ways that inputs can drive the oscillators and the ways that the oscillators can adapt their settings to achieve synchronous actions are discussed in Sect. III. The taxonomy of adaptive oscillators includes a variety of structures related to AWO and phase-reset oscillators. Sect. IV examines the behavior of the oscillators as the parameters adapt to follow various input signals. The adaptation is analyzed in a variety of settings and the optimum (equilibrium) points of the algorithm are shown graphically and derived analytically. As becomes clear, when the wavetable is chosen properly for the class of input signals, the oscillators can adapt quickly and robustly. When the input signals deviate from the expected class, the oscillators may fail to achieve synchronization. Since the wavetable is an easily controlled design parameter, insuring a good match is often straightforward.

II. Fixed Oscillators

Perhaps the simplest oscillator is the second order linear system

$$\ddot{x} + \omega^2 x = 0$$

which has the explicit solution, $x(t) = e^{-j\omega t+\phi}$, thus describing an oscillator with frequency f where $\omega = 2\pi f$. The equivalent discrete-time linear oscillator is

$$x[k] = \beta x[k-1] - x[k-2]. \tag{1}$$

Depending on the values of β and the initial conditions $x[0]$ and $x[1]$, (1) generates a sampled sinusoidal signal. Detailed investigation of the numerical properties of such oscillators may be found in [1] and [6].

Classical oscillators such as a pendulum regulated by an escapement [7] or the Van der Pol oscillator [21] operate by wrapping a nonlinearity around a linear oscillator. This nonlinear feedback injects energy to counteract the inevitable dissipative forces and stabilizes the oscillator about its operating frequency. Accordingly, when the feedback term is small, the output of the oscillator is approximately sinusoidal. Other kinds of oscillators model systems which grow and/or decay rapidly. For example, the Fitzhugh-Nagumo oscillator [5] is a two-variable differential equation $\left(\frac{dx}{dt} = -x(x-\tau_1)(x-\tau_2) - y - \Omega,\ \frac{dy}{dt} = \epsilon(x - \tau_3 y)\right)$ that models the cyclical action potential of neurons. The model gradually accumulates energy until a threshold is reached; it then “fires” and releases the energy, beginning a new cycle.

Phase-Reset Oscillators I

The stabilizing nonlinearities in the Van der Pol and Fitzhugh-Nagumo oscillators are smooth and differentiable. Other oscillators use more severe, discontinuous nonlinearities. For example, the “phase reset” methods contain an integrator (or a summer) which accumulates the input until it reaches a predefined positive threshold value τ. Whenever the output crosses the threshold, it is immediately reset to a fixed value y_0. To be concrete, let $y[k]$ be the output of the oscillator at time k, let $b \geq 0$ be the amount of increase, and $a > 0$ the decay (or growth) factor. Then the oscillator is

$$y[k] = f(ay[k-1] + b) \tag{2}$$

where the nonlinearity $f(\cdot)$ is

$$f(s) = \begin{cases} s & 0 < s < \tau \\ y_0 & \text{otherwise} \end{cases} .$$

The oscillator is said to be periodic if there is a positive integer P such that $y[k] = y[k + mP]$ for all m and k, and the smallest such P is the period of the oscillator. For the phase reset oscillator above, if $a = 1$ and $y_0 = 0$, then the output of the integrate and fire oscillator is a ramp or sawtooth wave with a period equal to $P = \frac{\tau}{b}$ as shown in Fig. 1(a). In general, for $a \neq 1$, the period is given by $P = \frac{ln\left(\frac{\tau(1-a)-b}{(1-a)y_0-b}\right)}{ln(a)}$ if this exists and is positive, as shown in Appendix -A. For some values of a, y_0, τ and b, the period may not exist and there is no oscillation. Typical trajectories are shown in Fig. 1. The parameters have been chosen so that the period is $\frac{1}{2}$ sec and the oscillation has a fundamental frequency of 2 Hz.

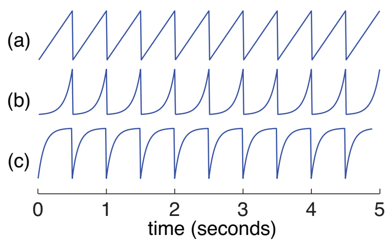

Fig. 1. The trajectories of three phase-reset oscillators: (a) a ramp oscillator with $a = 1$, $\tau = 1$, $b = 0.002$, $y_0 = 0$ (b) a growth oscillator with $a = 1.1$, $\tau = 1$, $b = 0$, $y_0 = 0.0085$ and (c) a decay oscillator with $a = 0.9$, $\tau = 1$, $b = 0.1$, $y_0 = 0$.

Wavetable Oscillators

In classical oscillators, the shape of the output waveform is linked to the values of the state and to the dynamics of the generating system. A modern innovation is the wavetable oscillator [17], which stores one or more periods of the oscillation in a table and reads values from the table to generate its output. The data may be resampled or read at different speeds in order to change frequency. This provides complete flexibility in the shape of the output because the data in the wavetable is decoupled from the state of the oscillator.

Suppose that $w(t)$ is a periodic signal with period T. If the bandwidth of $w(t)$ is B and if the sampling rate $T_s < \frac{1}{2B}$, it is possible to exactly reconstruct $w(s)$ at any point s using only $N = \lceil \frac{T}{T_s} \rceil$ samples $w[i], i = 1, 2, \ldots, N$ of a single period. Though such Shannon reconstruction is computationally intensive, it can be approximated using relatively simple interpolation methods. It is always possible to resample at a faster rate (to lower the frequency of the oscillator). But resampling slower can be done without distortion only as long as the resampled rate αT_s remains less than $\frac{1}{2B}$. For large α, this will eventually be violated. These limits are discussed in [20] for a variety of standard waveshapes.

Indexing into the $N \times 1$ vector wavetable w may be accomplished using the recursion

$$s[k] = (s[k-1] + \alpha) \mod N \tag{3}$$

where mod N is the remainder after division by N and the index is initialized as $s[0] = 0$. The output of the oscillator at time k is then

$$o[k] = w((s[k] + \beta) \mod N) \tag{4}$$

where β specifies the phase. The period of the oscillator is $\frac{NT_s}{\alpha}$ sec. Equivalently, its frequency is $\frac{\alpha}{NT_s}$ Hz. If $s[k] + \beta$ happens to be an integer then the output of the oscillator can be read directly from the table. Otherwise, interpolation is required.

Such wavetable synthesis is common in applications such as the sample playback of electronic musical instruments. The stored waveform might be a multiple - period sampling of an acoustic instrument such as a flute playing the note C. Whenever a key is pressed on the keyboard, the recording of the flute note plays back but with a different α to transpose the frequency (and hence the pitch). One advantage of the wavetable approach is that it allows straightforward control over the frequency (via α) and phase (via β), and cleanly separates these control parameters from the shape of the oscillatory waveform. It is also possible to index smoothly through a wavetable using the technique in Appendix -B.

Phase-Reset Oscillators II

At first glance it may appear as if the wavetable equation (4) is the same as the phase-reset equation (2) with the mod function serving the same role as $f(\cdot)$. However the two are fundamentally different. The phase reset oscillator is like an accumulator with a threshold whereas the wavetable oscillator stores the output values of the waveform and updates the index into the table.

A threshold τ can be incorporated into the wavetable approach by indexing into w using the recursion

$$s[k] = \begin{cases} 0 & o[k] \geq \tau \\ (s[k-1] + \alpha) \text{ mod } N & \text{otherwise} \end{cases} \tag{5}$$

where $o[k] = w(s[k])$ is the output of the oscillator at time k. If $o[k]$ never exceeds the threshold, the period of the oscillator is $\frac{NT_s}{\alpha}$. If $o[k]$ does exceed the threshold, the period is $\frac{mT_s}{\alpha}$ where m is the (first) index at which $w(m) \geq \tau$. Phase-reset oscillators provide flexibility when incorporating inputs.

III. Adaptive Oscillators

The oscillators of the previous section are unforced: they have no inputs other than the fixed parameters of the model. In order for an oscillator to respond to signals in the world, there must be a way for it to incorporate external signals into its operation: this is a *forced oscillator*. In order to be useful, there must be a way for the parameters of the oscillator to change in response to external events: this is an *adaptive oscillator*. There are four key issues in the design of an adaptive oscillator: the kind of oscillator (Van der Pol, phase-reset, wavetable, etc.), the kinds of inputs it is designed to accommodate (sinusoidal, impulsive, random, etc.), which parameters will be adapted, and the mechanism (or algorithm) implementing the adaptation. The first example considers the phase locked loop, which addresses these issues using a sinusoidal oscillator, operating with sinusoidal inputs, and adjusting the phase of its internally generated sinusoid using a hill-climbing strategy [8]. The remainder of this section examines a number of different adaptive oscillators; subsequent sections examine their behavior with a variety of inputs.

A. The Phase Locked Loop

Many modern telecommunication systems operate by modulating a message sequence (usually a long string of binary encoded data) with a sinusoidal “carrier” wave. In order to recover the data, the receiver must track this carrier wave. To accomplish this, a local oscillator at the receiver generates and adapts an internally generated sinusoid to match the received carrier. The phase locked loop is one way of accomplishing this goal. Suppose that the input signal is a sinusoid $r(t) = \cos(2\pi ft + \phi)$ with known frequency f but unknown phase ϕ. The local oscillator generates $s(t) = \cos(2\pi ft + \theta)$. The product of $r(t)$ and $s(t)$ can be used to define a cost function

$$\begin{aligned} J(\theta) &= \text{LPF}\{r(t)s(t)\} = \text{LPF}\{\cos(2\pi ft + \phi)\cos(2\pi ft + \theta)\} \\ &= \frac{1}{2}\text{LPF}\{\cos(\phi - \theta)\} + \frac{1}{2}\text{LPF}\{\cos(4\pi ft + \theta + \phi)\} \\ &\approx \frac{1}{2}\cos(\phi - \theta) \end{aligned} \tag{6}$$

where the approximation holds as long as the bandwidth of the linear low-pass filter (LPF) is less than $2f$. Accordingly, the values of θ that maximize $J(\theta)$ occur when $\theta = \phi + 2n\pi$ for any integer n.

One way to find the maximum of such a function is to use a gradient strategy that iteratively moves the value of the argument in the direction of the derivative. The derivative of (6) with respect to θ can be approximated

$$\frac{dJ(\theta)}{d\theta} \approx -\text{LPF}\{r(t)\sin(2\pi ft + \theta)\}.$$

The LPF and the derivative commute because they are both linear operations. (The primary requirement for a close approximation is that the stepsize μ of (7), which governs the speed of adaptation of θ, be small. Details can be found in Appendix G3 of [8].) The corresponding hill-climbing algorithm is $\dot{\theta} = \mu\frac{dJ(\theta)}{d\theta}$, or, in discrete time

$$\theta[k+1] = \theta[k] + \mu \left.\frac{dJ(\theta)}{d\theta}\right|_{\theta=\theta[k]}.$$

Substituting for the derivative and evaluating at the appropriate θ yields

$$\theta[k+1] = \theta[k] - \mu\text{LPF}\{r(kT_s)\sin(2\pi fkT_s + \theta[k])\} \tag{7}$$

where T_s is the sampling interval and k is the iteration counter. This is the same as the standard PLL as in [10]. The phase parameter θ in the PLL plays the same role that the phase parameter β plays in the wavetable oscillator (4). Using a wavetable w defined to be the N samples of a sinusoid, the evolution of θ acts to synchronize the phase of the local oscillator (that is, of w) to the phase of the input $r(t)$.

The cost function (6) correlates the input with the waveshape of the oscillator. A variety of alternative cost functions can also be used. For example, the least squares cost

$$J_{LS}(\theta) = \text{LPF}\{(r(t) - \cos(2\pi f t + \theta))^2\} \tag{8}$$

leads to the algorithm

$$\theta[k+1] = \theta[k] - \mu\text{LPF}\{(r(kT_s) - \cos(2\pi f k T_s + \theta[k]))\sin(2\pi f k T_s + \theta[k])\}.$$

Similarly, the cost function

$$J_C(\theta) = \text{LPF}\{(r(t)\cos(2\pi f t + \theta))^2\}$$

leads to the “Costas loop” [2]

$$\theta[k+1] = \theta[k] - \mu\text{LPF}\{r(kT_s)\cos(2\pi f k T_s + \theta[k])\}\text{LPF}\{r(kT_s)\sin(2\pi f k T_s + \theta[k])\}. \tag{9}$$

Analysis of these variations of the PLL (as in [8]) show that they are more-or-less equivalent in the sense that they have the same answers in the ideal case, though the algorithms may react differently to noises.

B. Adaptive Wavetable Oscillators

A wavetable oscillator (as defined in Sect. II) consists of an array w containing N stored values of a waveform. The output of the oscillator at time k is (4)

$$o_1[k] = w((s[k] + \beta_1) \text{ mod } N)$$

where the indices into w are given by equation (3). The parameter α in (3) specifies the frequency of the oscillation while β_1 in (4) defines the phase. AWOs achieve adaptivity by adjusting the parameters to align the oscillator with an external input. This can be accomplished in several ways. Suppose that the input to the oscillator is $i[k]$. An analogy with the PLL uses the correlation-style cost function

$$J(\beta_1) = \text{LPF}\{i[k]o_1[k]\}$$

that directly parallels (6). The β_1 that maximizes J provides the best fit between the input and the oscillator. It can be adapted using a gradient descent strategy

$$\beta_1[k+1] = \beta_1[k] + \mu \frac{dJ}{d\beta_1} = \beta_1[k] + \mu \text{LPF}\left\{ i[k] \left.\frac{dw}{d\beta_1}\right|_{\beta_1=\beta_1[k]} \right\}. \tag{10}$$

Since w is defined by a table of values, $\frac{dw}{d\beta_1}$ is another table, the numerical derivative of w. Observe that $\frac{dw}{d\beta_1}$ is a scalar times $\frac{dw}{dt}$. Several candidate wavetables and their derivatives are shown in Fig. 2.

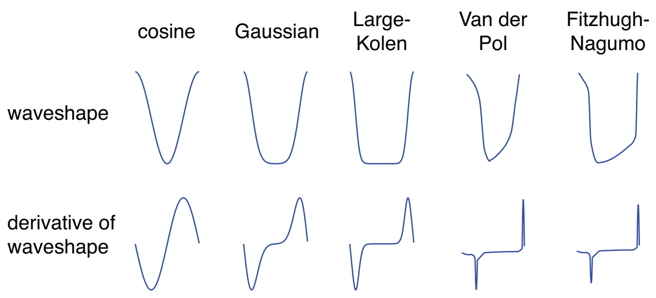

Fig. 2. Five common wavetables and their derivatives. The cosine wavetable is used in the PLL. The Gaussian shape is shifted so that the largest value occurs at the start of the table. The wavetable for the Large-Kolen oscillator is defined by $1 + \tanh(\gamma(\cos(2\pi f t) - 1))$. The Van der Pol and Fitzhugh-Nagumo waveshapes are defined using waveforms culled from numerical simulations.

The update for the frequency can be accomplished using the integral/derivative relationship between frequency and phase. [1] To be specific, let

$$o_2[k] = w((s[k] + \beta_1[k] + \beta_2[k]) \bmod N)$$

be a second oscillator. A gradient update for β_2 proceeds as above by maximizing $J(\beta_2) = \text{LPF}\{i[k]o_2[k]\}$ with respect to β_2. The formula is the same as (10) with the appropriate substitutions.

The AWO using β_1 and β_2 can follow changes in the frequency of the input, but the parameter α in (3) is never changed. A consequence of this is that if the input ceases, the oscillator returns to its original frequency.

[1] Analogous updates for the PLL are defined in [8].

It is also possible to adapt the frequency parameter. This allows the oscillator to "learn" the frequency of the input and to continue at the new frequency even if the input stops. Perhaps the simplest technique for adapting the frequency is to use a gradient strategy that maximizes $J(\alpha) =$ LPF$\{i[k]o_1[k]\}$. This is:

$$\begin{aligned} \alpha[k+1] &= \alpha[k] + \mu_\alpha \frac{dJ}{d\alpha} \\ &= \alpha[k] + \mu_\alpha \text{LPF}\{\ i[k] \left.\frac{dw}{ds}\frac{ds}{d\alpha}\right|_{\alpha=\alpha[k]} \}. \end{aligned} \tag{11}$$

Since $s[k]$ is defined by the recursion (3), the derivative with respect to α cannot be expressed exactly. Nonetheless, when the stepsizes μ_α are small, it can be approximated by unity, and so the update has the same form as the update for β. Because the frequency parameter is more sensitive, its stepsize μ_α should be chosen smaller than the stepsize used to adapt the phase.

In adapting the β's and α's of the AWO, other cost functions may be used. For example, minimizing $J_{LS} =$ LPF$\{(i[k] - o[k])^2\}$ leads to an update that optimizes a least-squares criterion while maximizing $J_C =$ LPF$\{(i[k]o[k])^2\}$ leads to a "Costas loop" method of updating the oscillator parameters. These parallel directly the sinusoidal methods associated with the PLL in (8)-(9).

Besides the PLL, many common oscillators are captured by the AWO structure. For example, for Large and Kolen's [9] oscillator, the wavetable is defined by $1 + \tanh(\gamma(\cos(2\pi f t) - 1))$ as shown in Fig. 2 and the intended input is a spike train. Both the phase β and the frequency α of the local oscillator are adapted using a "hill climbing" strategy.

When attempting to locate the position of a train of spikes in time, oscillators that use pulses (such as Large and Kolen's or the Gaussian) are intuitively plausible. The pulse can be thought of as providing a window of time over which the oscillator "expects" another spike to occur. If the spike occurs at exactly the right time, the derivative is zero and there is no change. If the spike occurs slightly early, the derivative is positive and the phase increases. If the spike occurs late, the derivative is negative and the phase decreases. This process of adjustment actively aligns the oscillator with the spike train. Just as importantly, there is a zone between pulses where the value of the waveshape and its derivative are both small. In this region, the update term is small and the oscillator is (relatively) insensitive to extraneous spikes and noisy data.

Fig. 3 shows how the AWO responds to an input

$$i(t) = \begin{cases} 1 & t = nT,\ n = 1, 2, \dots, M \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

that is a regular train of spikes spaced $T = 500$ ms apart. The simulation uses the Gaussian pulse oscillator with phase and frequency parameters adapted according to (10) and (11). In (c), α was initialized with period 550 ms, corresponding to a 10% error. The phase and frequency converge within a few periods and the pulses align with the spikes. The oscillator continues at the adapted frequency even after the input ceases. The same oscillator may synchronize in various ways to the same input depending on the initial values. The figure also shows 1:2, 2:1, and 3:4 synchronizations where n:m means that n periods of the oscillator occur in the same time as m periods of the input. While such nonunity synchronizations are common in the mode locking of oscillators, they are encouraged by the specifics of the waveshape. The dead (zero) region between pulses means that the adaptation is insensitive to spikes that occur far away from expected location. For example, in the 1:2 synchronization, the “extra” spikes occur at precisely the point where they will have the least influence on the adaptation. On the other hand, in the 2:1 synchronization, the input is zero for the duration of the “off beat” pulse and hence does not affect adaptation. The final (bottom) simulation in Fig. 3 shows a stable 3:4 synchronization. These simulations (using the Gaussian pulse shape) are effectively the same as using the Large-Kolen oscillator. The convergence is similar with a cosine wave (as suggested in [12]) suggesting that the details of the waveshape are not particularly crucial.

The stepsize parameters μ and μ_α affect the adaptation and synchronization in a direct way: larger stepsizes allow faster convergence but also may result in overshoot or unstable behaviors. Smaller stepsizes allow smoother adaptation at the cost of longer time to converge. Useful values lie somewhere between. (In the simulations of Fig. 3 the stepsizes were 0.01 and 0.001.) The behavior of AWO is discussed further in Sect. IV.

C. Adaptive Phase-Reset Oscillators

The input to the AWO is used only in the adaptation of the control parameters. The input to the adaptive phase-reset oscillator is incorporated into the thresholding process. In some cases, this allows the adaptive

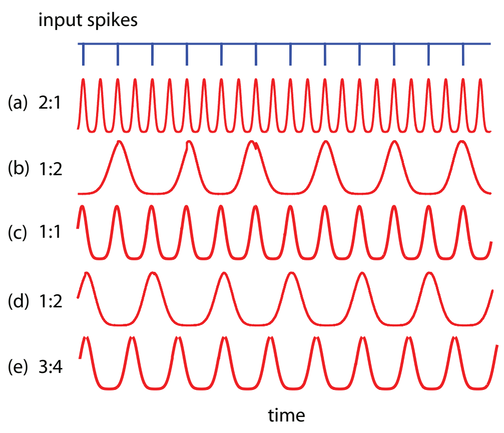

Fig. 3. The input spike train (12) excites the AWO and the parameters β and α are updated according to (10) and (11). The initial value in (a) was $\alpha = 240$ ms and the oscillator synchronizes to a 2:1 rate (two oscillator pulses occur for each input spike). (b) and (d) were initialized at $\alpha = 1050$ ms. The oscillator synchronizes to a 1:2 rate (one oscillator output for every two input spikes). Depending on the initial value of β, the oscillator can lock onto either the odd or the even spikes. (c) was initialized at $\alpha = 550$ ms. Other synchronizations such as 3:4 are also possible.

phase-reset oscillator to synchronize more rapidly to the input without suffering from the effects of a large stepsize.

As in (5), the threshold τ defines the indexing into a wavetable w using a recursion

$$s[k] = \begin{cases} 0 & o[k] + i[k] \geq \tau \\ (s[k-1] + \alpha) \bmod N & \text{otherwise} \end{cases} \tag{13}$$

where the input to the oscillator at time k is $i[k]$ and the output from the oscillator is $o[k] = w(s[k])$. The adaptation of the frequency parameter α can proceed in any of the usual ways. McAuley [12] suggests[2] minimizing the least square error $J_{LS}(\alpha) = \text{LPF}\{i[k](\tau - o[k])^2\}$ using a gradient strategy. An alternative is to maximize the correlation cost $J_{cor}(\alpha) = \text{LPF}\{i[k]o[k]\}$. In either case, the update for α is

$$\alpha[k+1] = \alpha[k] \pm \mu \frac{dJ}{d\alpha}$$

where the $+$ is used to maximize and the $-$ is used to minimize. Both cost functions result in performance similar to that in Fig. 4. The top plot shows α converging to its final value of one within the time span of a few input spikes. Observe that the convergence is not necessarily monotonic: the first step moves in the "wrong"

[2]McAuley's algorithm has other features (such as activation sharpening) that distinguish it from the present discussion.

direction because of an unfortunate initial relationship between the phase of the output of the oscillator and the onset of the input spikes. Despite the false start, convergence is rapid. The bottom plot shows the output of the oscillator as it resets its phase with the onset of each input spike. Synchronization aligns the output with the input spikes. This simulation used the sinusoidal wavetable; others such as the Gaussian, the Large-Kolen, Van der Pol, and Fitzhugh-Nagumo shapes (recall Fig. 2) behave approximately the same.

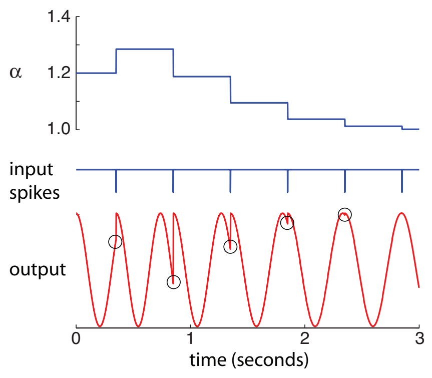

Fig. 4. The phase-reset oscillator (13) adapts to maximize the correlation between the input (12) and the output of the oscillator, shown in the bottom plot. The frequency parameter α converges, within only a few periods of the input, to its desired value of 1, despite initially moving in the wrong direction. Each time an input spike arrives, the oscillator resets. The small circles annotate the output of the oscillator at the points of reset, which move to align the maximum points of the output with the input spikes.

IV. Behavior of Adaptive Oscillators

As suggested by the simulations, the behavior of adaptive oscillators are dependent on several factors: the kind of oscillator, the waveshape, the fixed parameters (e.g., stepsizes, threshold values), the method of adaptation, and the input. This section shows how the synchronization can be understood by careful consideration of the cost function, and analyzes the convergence of the algorithms under suitable simplifying assumptions.

Regular Pulse Trains

Adaptive oscillators are often designed to synchronize to specific classes of input sequences. When the input is indeed of this form, it is reasonably straightforward to understand the convergence of the oscillator by plotting the cost functions for all possible values of the parameters. For example, [8] shows plots of the cost surfaces for the various forms of the PLL (7)-(9) when the input is the expected sinusoid. The "cost surfaces" show how the algorithm behaves: when maximizing, the parameters climb the surface. When minimizing, the parameters descend the surface. The summits (or the valleys) are the values to which the algorithm converges.

Many oscillators easily synchronize to inputs consisting of trains of isolated pulses such as (12), as are used here to demonstrate the various properties of the cost function. The cost function $J_{cor}(\alpha) = \text{LPF}\{i[k]o[k]\}$ for the phase-reset adaptive oscillator of (13) is shown in Fig. 5(a) and (b). The input is assumed to be a pulse train (12) with $T = 500$ ms, the threshold is $\tau = 1$, and the cost function $J(\alpha)$ is plotted for α between 0.1 and 5. Time is scaled so that a value of $\alpha = 1$ is the "correct" answer where one period of the oscillator occurs for each input spike. In Fig. 5(a), the waveshape is the cosine while (b) uses the Gaussian waveshape (recall Fig. 2). In both cases, if the α is initialized between 0.5 and 1.5 then it will climb the surface until it reaches the desired value of one. If it is initialized between 1.5 and 2.5, it converges to $\alpha = 2$, which is when two periods of the oscillator occur for each single input spike. Similarly, it can also converge to any integer multiple of the input spike train.

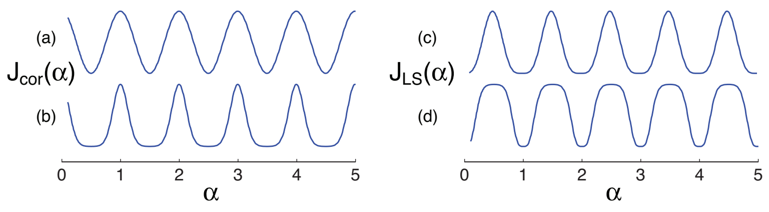

Fig. 5. Cost surfaces for the adaptive phase-reset oscillator. In all four cases, (Gaussian and cosine waveshapes, least squares and correlation costs) the α parameter is optimized at the "correct" value of 1, or at a simple integer multiple.

Figs. 5(c) and (d) show the cost surfaces for the Least Squares $J_{LS}(\alpha) = \text{LPF}\{i[k](\tau - o[k])^2\}$. Part (c) uses the cosine waveshape and (d) uses the Gaussian waveshape; all other values are the same. In these cases, if the α is initialized between 0.5 and 1.5, it descends the surface (because it is minimizing rather than maximizing) until it reaches the bottom at the desired value of unity. If it is initialized between 1.5 and 2.5, it converges to $\alpha = 2$. In all cases, the adaptive parameter converges to an integer multiple of the input spike train.

The AWO of Sect. III-B adapts both the phase β and the frequency α, and so the cost function $J(\alpha, \beta)$ is two dimensional. The surface for the correlation cost is shown in Fig. 6. The algorithm is initialized at some α, β pair and then evolves on this surface, climbing at each step. As before, α is normalized so that unity corresponds to one period of the oscillator for each input spike. The parameter β must range from zero to α to cover all possible phases.

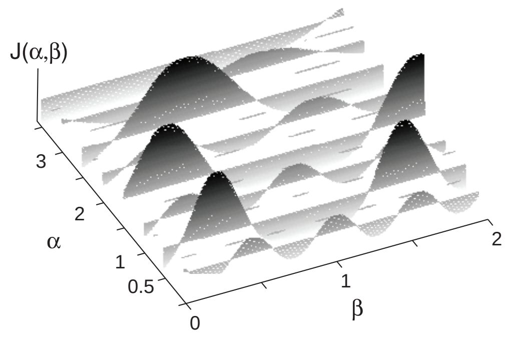

Fig. 6. The cost surface for the phase and frequency updates (10)-(11) of the AWO. Depending on the initial values, the period may converge to $\alpha = 1, 2, 3, \ldots$ or any of the other peaks at integer multiples. There are also stable regions of attraction surrounding $\alpha = 0.5, 1.5, 2.5,$ and 3.5, which correspond to various $n{:}m$ synchronizations.

Observe that the oscillator may converge to different values depending on its initialization. If started near $\alpha = 1$, it inevitably synchronizes so that each period of the oscillator is aligned with an input spike. But other values are possible: α may converge to an integer multiple, or to a variety of $n{:}m$ synchronizations. Some of the behaviors that are suggested by the cost surfaces can actually be proven. For example, Appendix -C analyzes the cost function for the correlation cost when the input and output are spike trains. As expected, the result is that the cost is maximized exactly when the frequency of the oscillator matches the rate of the input, or at some simple integer multiple (or submultiple). Similarly, Appendix -D shows that similar stable points result when the output is an arbitrary wavetable. Note that the cost surface (in Fig. 6) reflects the behaviour of the oscillator for small stepsizes.

Obviously, the wavetable $w(t)$ may assume a large variety of different shapes. But it cannot be arbitrary. Assume that the output of the AWO is

$$o(t) = \sum_{m=1}^{N} w(t - m\alpha - \beta) \tag{14}$$

and let the input be a pulse train with period α^* and phase β^*

$$i(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\alpha^* - \beta^*). \tag{15}$$

The cost function is then

$$J(\alpha, \beta) = \sum_{m=1}^{N} \sum_{n=1}^{N} w(n\alpha^* + \beta^* - m\alpha - \beta). \tag{16}$$

If the wavetable (a) has a global maximum at $t = 0$, (b) has support $(-\frac{\alpha}{2}, \frac{\alpha}{2})$, (c) is monotonic increasing in $(-\frac{\alpha}{2}, 0)$, and (d) is symmetric about $t = 0$, then the cost function achieves its extremal points at one of the simple integer multiples $n{:}m$ which correspond to the points at which the algorithm achieves synchronization. Appendix -D provides the details and a proof.

Cost surfaces like Figs. 5 and 6 only show the behavior of the algorithm when the input spikes are completely regular, that is, when the spikes occur at a fixed rate. Suppose that the rate of the spike train changes. As long as the change is slow enough so that the generic shape of the surface does not change dramatically from step to step, the frequency values will be able to track the changes. Suppose that the frequency parameter has converged to the summit of one of the humps in Figs. 5 and 6. As the underlying rate of the input changes, the height and the width of the hump varies. As long as the variation is not large enough to throw the frequency estimates onto another hump (or to keep it perpetually oscillating between diverse humps) then the algorithm can track the changes in the input. For example, the spikes may represent the rotation pulses of an engine or the events in a musical passage. When the engine changes speed or the tempo changes (as long as they do not change too rapidly) there is reason to hope that the adaptive oscillators can follow.

A. Phase-Reset Oscillators and Irregular Pulse Trains

There are many ways that a regular pulse train may become irregular: spikes in the input may fail to occur, extra spikes may be present, or noise may be added to the input. This section examines the shape of the cost function of the adaptive phase-reset oscillator under these various irregularities. The discussion focuses on the Gaussian waveshape and the correlation cost function, but the general outlines of the results are the same for other wavetables and other cost functions.

First, the adaptation is robust to the deletion of events. Converging takes longer and is more tenuous, but once in the neighborhood of the correct answer, the phase-reset algorithm can tolerate more than 90% deletions and remain synchronized. Fig. 7 shows three cases where 50%, 80%, and 95% of the input spikes have been removed. In all three cases, there are still prominent peaks in the cost function at the expected integer values of α. The implication of the narrow peaks is that the region of convergence is reduced, that is, the range of initial α values that will converge properly is smaller than without the deletions. Initial values that are not near the peaks may converge to one of the local maxima, the smaller peaks that occur between the integer values.

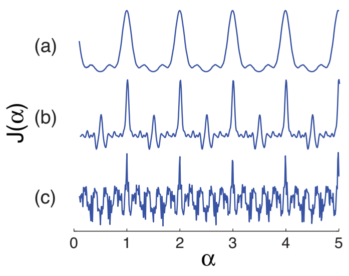

Fig. 7. The cost functions for the adaptive phase-reset oscillator when (a) 50% of the input spikes have been (randomly) deleted, (b) 80% have been deleted, and (c) 95% have been deleted. Despite the high percentages, all still retain their optima at $\alpha = n$ for integers n. As the number of deletions increases, the region of attraction of the optimal values shrinks and local maxima occur.

The algorithm is also fairly robust to the addition of spikes that are randomly placed in time. Fig. 8 shows three situations. In the first, an average of one extra spike is added per period (at random times). The convergent points of the algorithm are unchanged from the perfectly regular case. As the percentage of correct spikes decreases, so does the ability of the algorithm to synchronize; the randomly spaced spikes succeed in hiding the underlying regularity of the spike train. As the percentage of random spikes increases, the algorithm develops a tendency to converge to $\alpha = 0$, which is a degenerate situation where the oscillator continuously repeats the first value in the wavetable. This represents a divergence (towards infinity) of the frequency of the oscillator. As is clear from the figures, the oscillator is more robust to deletions than inserions. This is because there is more erroneous data to drive the updates with insertions than with deletions.

The Achilles heel of the adaptive phase-reset oscillator is its sensitivity to additive noise; even small amounts of noise can bias and/or destabilize the iteration. Fig. 9 shows the cost function when Gaussian

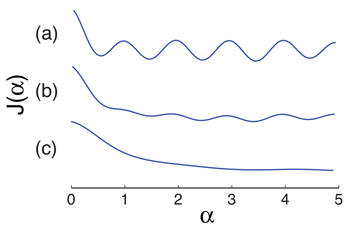

Fig. 8. The cost function for the adaptive phase-reset oscillator when (a) an average of one extra (randomly located) spike is inserted per period, (b) an average of two extra spikes, and (c) an average of three extra spikes. As the density of spikes increases, the optimal values cease to be maxima of the cost function and the oscillator cannot be expected to converge except to the degenerate solution $\alpha = 0$.

noise is added. In case (a), the standard deviation of the added noise is 0.004, in (b) it is 0.01 and in (c) it is 0.1. Even in (a), the convergent value for α is significantly biased away from one (the peak occurs at approximately $\alpha = 1.2$). In the modest noise case, the peak near one disappears completely, and in the high noise case, the algorithm is unstable for any initial values.

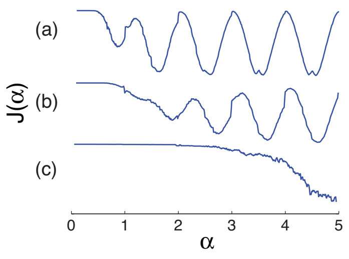

Fig. 9. The cost function for the adaptive phase-reset oscillator when (a) the input is subjected to additive Gaussian noise with standard deviation 0.004, (b) with standard deviation 0.01, and (c) with standard deviation 0.1. As the amount of noise increases, the optimal values cease to be maxima of the cost function and the oscillator cannot be expected to converge.

Why is the phase-reset adaptation so sensitive to such small additive perturbations? The heart of the problem lies in the thresholding. Any additive noise tends to cause the oscillator to fire early (because the threshold tends to be exceeded in each cycle when the noise adds to the oscillator output). Since the threshold is crossed early, the oscillator tries to compensate by increasing the period. This is why Fig. 9(a) and (b) shows such severe biasing of the α. Eventually, the noise gets large enough and the thresholding occurs irregularly enough that synchronization is lost completely.

B. Adaptive Wavetable Oscillators and Irregular Pulse Trains

The behavior of the AWO is similar to that of the phase-reset oscillator when the input is a train of spikes. For example, both are robust to random additions and deletions of spikes in the input. The AWO is more resilient to additive noises than the adaptive phase-reset oscillator: it can withstand more than 0.04 standard deviation without losing synchrony. Fig. 10 shows the cost function when the input is corrupted by additive noise. The region of attraction of the humps is smaller than in the noise-free case and the noise floor of the figure is much higher. Nonetheless, the humps in the cost function occur at the desired integer locations and the oscillator can still achieve unbiased synchronization with the input.

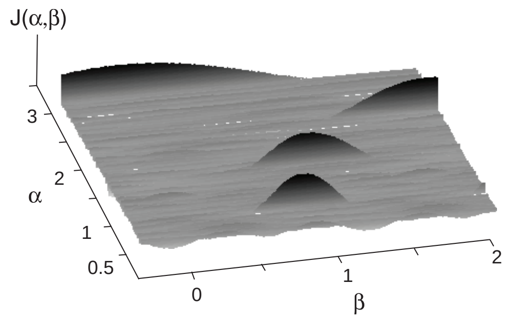

Fig. 10. The cost surface for the phase and frequency updates (10)-(11) of the AWO using the Gaussian waveshape. Additive noise with standard deviation of 0.04 is added to the input, which is a regular train of spikes. Depending on the initial values, the period may converge to $\alpha = 1, 2, 3, \ldots$

The discussion so far has focussed on the use of spike train inputs because this is the framework in which much of the research has been done. For instance, [4], [9], [11], [13], [16], and [18] consider spike train inputs exclusively. Indeed, the phase-reset oscillator requires inputs of this form. (Otherwise, the thresholding operation fails to function properly.) But the AWO can operate with a variety of different inputs and there is nothing inevitable about the use of spikes to represent events. A regular succession can be represented equally well by a sequence of Gaussian shaped pulses, by a series of triangular waveforms, or by a collection of equally spaced sinusoids. To see why this may be advantageous, observe that the update terms in the adaptive oscillators are multiplied by $i[k]$. When this is zero, no update occurs. When the input is a spike train, updates can only occur at the exact time of the spikes. But with other pulse shapes, updates can occur throughout the cycle. This can have the effect of averaging out the noise and making the adaptation less sensitive to additive noises. Appendix -D examines the optimum points of the cost function when the wavetable is symmetric, convex, and has a unique global maximum at its midpoint. Not surprisingly, the algorithm converges to the same kinds of n:m synchronization ratios as with the spike train models.

To see this, suppose that the input is a regular succession of Gaussian pulses contaminated with a significant amount of noise, as shown in Fig. 11(a). Using the Gaussian pulse waveshape in the AWO, synchronization is readily achieved. Fig. 11(b) shows the output of the oscillator, which is essentially a "cleaned up" version of the input.

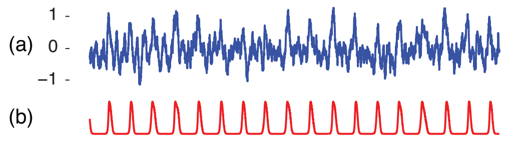

Fig. 11. Changing the input to a train of Gaussian pulses allows the AWO to tolerate additive disturbances, shown here as additive white Gaussian noise with a standard deviation of 0.04.

The cost function for this case is shown in Fig. 12. The large mountain at $\alpha = 1$ represents the proper operation of the oscillator as it synchronizes to the input from Fig. 11. But something odd has happened: the oscillator no longer has stable points at integer multiples of the correct period. This may be undesirable behavior for some applications (it would never be advantageous for a receiver to synchronize to twice the rate of the transmitter) but for other applications such as rhythmic analysis, it is desirable to have oscillators that are capable of locking onto different rates that correspond to the different hierarchical levels of musical meter.

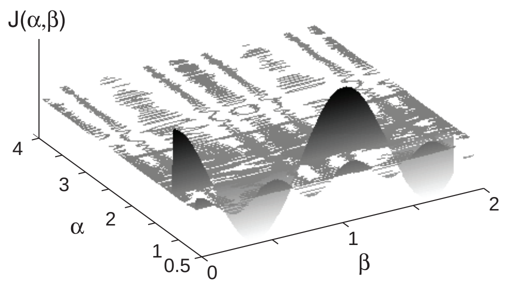

Fig. 12. With a significant amount of noise (standard deviation of 0.04), the AWO can still synchronize to a regular succession of Gaussian pulses. However, it loses the ability to synchronize with higher ratios. The only peaks occur at $\alpha = 1$ and $\alpha = 0.5$.

V. Conclusions

This paper has formalized a general class of oscillators that consist of a lookup table (which specifies the detailed shape of the output waveform) and a set of control paramaters (that specify the frequency and/or phase of the output). This separation between the waveshape and the control parameters in the AWO has several advantages. First, it is easy to choose the shape of the output waveform to match that of the input, and this shape remains fixed irrespective of the amplitude and frequency of the oscillation. (In contrast, in standard oscillators such as Van der Pol's, the output waveform changes shape as a function of amplitude, frequency, and coupling parameters.) Second, it is often straightforward to choose the cost function so as to achieve certain kinds of performance goals. For instance, in a noisy situation a "Costas Loop" cost (9) may be preferable to a least squares cost (8). In a situation with many outliers, an absolute deviation cost might be better. AWOs allow such design choices, which are absent from oscillators not derived from an optimization perspective. Third, there is less concern about chaotic instabilities in an AWO since the algorithm follows a simple, relatively well understood gradient strategy, at least for small stepsizes.

Fourth, the separation of state and control parameters in the AWO allows standard analytical techniques from adaptive system theory to be applied. For example, this paper analyzed an ideal AWO adapting to follow a periodic pulse train; generic conditions on the possible waveshapes were given and stable equilibria were shown to depend on the derivatives of the waveshape. Even when a complete analysis is not feasible, examples show that it is often straightforward to "compare and contrast" the behaviors of various candidate AWOs under a variety of operating conditions by plotting the cost functions. The number and kind of equilibria can be seen in the plots, which also give an idea of the size of the region of convergence of the various equilibria. When using a bank of oscillators, such information may be useful in estimating the number of oscillators needed to guarantee coverage. Finally, AWOs provide a common framework in which a number of seemingly unrelated oscillators can be considered together. This may help the system designer to choose the most appropriate from among the various possibilities.

References

[1] M. M. Al-Ibrahim, "A multifrequency range digital sinusoidal oscillator with high resolution and uniform frequency spacing," *IEEE Trans. on Circuits and Systems-II,* Vol. 48, No. 9, Sept. 2001.

[2] Costas, J.P., "Synchronous Communications," Proceedings of the IRE, Dec. 1956, pp. 1713-1718.

[3] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *J. New Music Research,* Vol. 30, No. 1, 2001.

[4] D. Eck, "Finding downbeats with a relaxation oscillator," *Psychological Research*, 66(1), 18-25, 2002.

[5] R. Fitzhugh, "Impulses and physiological states in theoretical models of nerve induction," *Biophysical Journal*, 1, 455-466, 1961.

[6] N. J. Fliege and J. Wintermantel, "Complex digital oscillators and FSK modulators," *IEEE Trans. Signal Processing,* Vol. 40, No. 2, Feb. 1992.

[7] C. Huygens, *Horologium* 1658.

[8] C. R. Johnson, Jr. and W. A. Sethares, *Telecommunication Breakdown: concepts of communications transmitted via software-defined radio,* Prentice-Hall, 2004.

[9] E. W. Large and J. F. Kolen, "Resonance and the perception of musical meter," *Connection Science* 6: 177-208, 1994.

[10] E. A. Lee and D. G. Messerschmitt, "Digital Communication," Boston, MA, Kluwer Academic Publishers, 1994.

[11] H. C. Longuet-Higgins and C. S. Lee, "The perception of musical rhythms," *Perception,* Vol. 11, 115-128, 1982.

[12] J. D. McAuley, Time as phase: a dynamic model of time perception," *Proc. Sixteenth Conf. of Cognitive Science Society,* Lawrence Erlbaum, 607-612, 1994.

[13] J. D. McAuley, *Perception of time as phase: toward an adaptive-oscillator model of rhythmic pattern processing*, Ph.D Thesis, Indiana University, 1995.

[14] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers,* Wiley, 1998.

[15] M. Niedźwiecki and P. Kaczmarek, "Identification of quasi-periodically varying systems using the combined nonparametric/parametric approach," *IEEE Trans. Signal Processing,* vol. 53, No.12, 2005.

[16] D. J. Povel, *Internal representation of simple temporal patterns, J. of Experimental Psychology* Vol. 7, No. 1, 3-18, 1981.

[17] M. Puckette, "Theory and Techniques of Electronic Music", March 2006, http://www.crca.ucsd.edu/∼msp/techniques/v0.08/book-html/

[18] D. Rosenthal, "Emulation of rhythm perception," *Computer Music Journal,* Vol. 16, No. 1, Spring 1992.

[19] W. A. Sethares, R. D. Morris and J. C. Sethares, "Beat tracking of audio signals," *IEEE Trans. on Speech and Audio Processing*, Vol. 13, No. 2, March, 2005.

[20] T. Stilson and J. Smith, "Alias-free digital synthesis of classic analog waveforms," http://www-ccrma.stanford.edu/∼stilti/papers

[21] B. van der Pol and J. van der Mark, "The heartbeat considered as a relaxation oscillation and an electrical model of the heart," *Phil. Mag. Suppl.* No. 6, 763-775, 1928.

[22] Wikipedia, "Equidistribution theorem", http://en.wikipedia.org/wiki/Equidistribution_theorem

Appendix

A. *Period of Phase-Reset Oscillator I*

Over the linear region, the output of (2) can be rewritten $y[k] = a^k y_0 + b \sum_{i=0}^{k-1} a^i$. The oscillation resets when $y[k]$ equals or exceeds the threshold τ. Thus the period P is given by the first k for which $y[k] \geq \tau$. For $a = 1$ and $y_0 = 0$, $y[k] = bk$. Then solving for $y[k] = \tau + \epsilon$, where $\epsilon \geq 0$, gives $P = \tau/b + \epsilon/b$. As $\epsilon \to 0$, $P = \tau/b$. For $a \neq 1$, the geometric series replaces the sum, yielding $y[k] = a^k y_0 + b\frac{1-a^k}{1-a}$. Substitute $y[k] = \tau + \epsilon$, where $\epsilon \geq 0$, and solve for a^k as $\epsilon \to 0$ to get $a^k = \frac{\tau(1-a)-b}{(1-a)y_0-b}$. Taking natural logs of both sides shows that this occurs when $k = \frac{ln\left(\frac{\tau(1-a)-b}{(1-a)y_0-b}\right)}{ln(a)}$, which gives the desired period P.

B. *Smooth indexing into Wavetable Oscillators*

Indexing into the wavetable oscillator using the mod(·) function as in (3) is discontinuous and difficult to analyze. If the wavetable is symmetric, the indexing can be accomplished using a sinusoid

$$s[k] = e^{j\theta} e^{2\pi j f k} \tag{17}$$

where f specifies the frequency and θ the phase. The sinusoid can be used to index into a wavetable w using

$$o[k] = w\left(\text{Re}\{\frac{N}{4}(1 + s[k])\}\right) \tag{18}$$

where Re$\{\cdot\}$ takes the real part of its argument. Thus the index of w sweeps both forwards (from 0 to $\frac{N}{2}$) and backwards (down to zero) through a table of length $\frac{N}{2}$ (rather than sweeping forward through a table of length N as in (3)-(4)). As before, interpolation is needed whenever the argument is not an integer.

Just as the wavetable oscillator of Sect. II can synchronize to an input by adapting the α and β parameters, the wavetable oscillator of (17)-(18) can synchronize by adjusting its frequency and phase parameters f and θ. Let $f[k]$ be the (time varying) frequency at time k, and let $\phi[k] = e^{2\pi j f[k]} \phi[k-1]$. Then $s[k]$ of (17) becomes $s[k] = e^{j\theta}\phi[k]$. Accordingly, given an objective such as the maximization of the correlation function

$J(f,\theta) = \text{LPF}\{i[k]o[k]\}$, hill-climbing updates for the parameters can be formed

$$\begin{aligned} f[k+1] &= f[k] + \mu \frac{dJ}{df}\bigg|_{f=f[k]} \\ &= f[k] + \mu \text{ LPF}\{\ i[k]\ \frac{dw}{ds}\ \frac{ds}{d\phi}\ \frac{d\phi}{df}\bigg|_{f=f[k]}\} \\ &= f[k] + \mu \text{ LPF}\{\ i[k]\ \frac{N}{4} w'\left(\frac{N}{4}(1+s[k])\right)\ e^{j\theta}\ \phi[k-1]\ 2\pi j e^{2\pi j f[k]}\} \\ &= f[k] - \bar{\mu} \text{ LPF}\{\ i[k]\ w'\left(\frac{N}{4}(1+s[k])\right)\ \text{Im}\{e^{j\theta}\phi[k]\}\}. \end{aligned}$$

The transition from the second to the third line occurs because the $\frac{d\phi}{df}$ term does not depend explicitely on $f[i]$ for $i < k$ (this parallels a similar derivation in [15]). The $\bar{\mu}$ in the final step consolidates the constants. The update for the phase proceeds analogously. Observe also that the waveshape output (18) from such an indexing scheme differs from a periodic repetition of w. If a given output shape is desired, the appropriate wavetable must be calculated.

C. Synchronization with Spike Train Inputs and Outputs

The analysis of the cost function is clearer in continuous time. Let

$$J(\alpha,\beta) = \lim_{T\to\infty} \frac{1}{T}\int_{t=-\frac{T}{2}}^{\frac{T}{2}} i(t)o(t)dt \tag{19}$$

where the input is a spike train with period α^* and phase β^* as in (15). The simplest case to analyze is when the output is also a train of pulses with estimated period α and phase β

$$o(t) = \sum_{m=-\infty}^{\infty} \delta(t - m\alpha - \beta). \tag{20}$$

Combining (19)-(20) gives

$$\begin{aligned} J(\alpha,\beta) &= \lim_{T\to\infty} \frac{1}{T}\int_{t=-\frac{T}{2}}^{\frac{T}{2}} \sum_{n=-\infty}^{\infty} \delta(t - n\alpha^* - \beta^*) \sum_{m=-\infty}^{\infty} \delta(t - m\alpha - \beta)dt \\ &= \lim_{T\to\infty} \frac{1}{T}\int_{t=-\frac{T}{2}}^{\frac{T}{2}} \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} \delta(t - n\alpha^* - \beta^*)\delta(t - m\alpha - \beta)dt \end{aligned}$$

The term $\delta(t - n\alpha^* - \beta^*)\delta(t - m\alpha - \beta)$ is zero except when

$$n\alpha^* + \beta^* = m\alpha + \beta. \tag{21}$$

Since n and m are integers, the integral counts the number of times (21) holds within the interval $(-\frac{T}{2}, \frac{T}{2})$. Let $s = \frac{\alpha}{\alpha^*}$. Dividing both sides of (21) by α^* and solving gives

$$n = ms + \gamma$$

where $\gamma = \frac{\beta - \beta^*}{\alpha^*}$. If s is irrational, there is at most one pair of n, m that fulfill (21). Hence the integral is bounded by 1. Taking the limit shows $J(\alpha, \beta) = 0$. If s is rational, it may be expressed as the ratio of two relatively prime integers p and q, i.e., $s = \frac{p}{q}$. This implies that q α's occur in the same interval as p α^*'s. Within the interval $(-\frac{T}{2}, \frac{T}{2})$, there are at least $\lfloor \frac{T}{p\alpha^*} \rfloor$ and at most $\lceil \frac{T}{p\alpha^*} \rceil$ nonzero terms where $\lfloor x \rfloor$ and $\lceil x \rceil$ represent the smallest and largest integers contained in x. Thus the cost can be bounded above and below by

$$\lim_{T \to \infty} \frac{1}{T} \lfloor \frac{T}{p\alpha^*} \rfloor \leq J(\alpha, \beta) \leq \lim_{T \to \infty} \frac{1}{T} \lceil \frac{T}{p\alpha^*} \rceil.$$

Since both limits converge to $\frac{1}{p\alpha^*}$ (by the technical lemma), this is the value of the cost function. Accordingly, $J(\alpha, \beta)$ has a (local) maximum at every rational s, which represents synchronization of $p : q$ for all integers p and q.

Proposition 1 (Technical Lemma) For any $x, y > 0$, $\lim_{y \to \infty} \frac{1}{y} \lfloor \frac{y}{x} \rfloor = \lim_{y \to \infty} \frac{1}{y} \lceil \frac{y}{x} \rceil = \frac{1}{x}$.

Proof: By definition, the first limit means that $\forall \epsilon > 0$, $\exists N$ such that if $y > N$ then $|\frac{1}{y} \lfloor \frac{y}{x} \rfloor - \frac{1}{x}| < \epsilon$. By definition of the floor function, $|\lfloor \frac{y}{x} \rfloor - \frac{y}{x}| \leq 1$. Dividing both sides by y gives $|\frac{1}{y} \lfloor \frac{y}{x} \rfloor - \frac{1}{x}| \leq \frac{1}{y}$. Thus, given ϵ, choose N such that $\epsilon > \frac{1}{N}$. Then for $y > N$, $\epsilon > \frac{1}{N} > \frac{1}{y} \geq |\frac{1}{y} \lfloor \frac{y}{x} \rfloor - \frac{1}{x}|$. The second limit proceeds analogously.

D. Synchronization with Spike Train Inputs and Wavetable Outputs

This Appendix analyzes the stationary points of the algorithm using the cost (19) when the input is the spike train (15) and where the output

$$o(t) = \sum_{m=-\infty}^{\infty} w(t - m\alpha - \beta) \tag{22}$$

is generated from a differentiable wavetable function $w(t)$ which is assumed to (a) have a global maximum at $t = 0$, (b) have support $(-\frac{\alpha}{2}, \frac{\alpha}{2})$, (c) be monotonic increasing in $(-\frac{\alpha}{2}, 0)$, and (d) be symmetric about 0. The cost function is

$$J(\alpha,\beta)=\lim_{T\to\infty}\frac{1}{T}\int_{t=-\frac{T}{2}}^{\frac{T}{2}}\sum_{n=-\infty}^{\infty}\delta(t-n\alpha^{*}-\beta^{*})\sum_{m=-\infty}^{\infty}w(t-m\alpha-\beta)dt.$$

Since $\sum_m w(t-m\alpha-\beta)$ is periodic with period α, it can be rewritten as a single period modulo α. The $\delta(\cdot)$ function samples this periodic function at $\beta^{*}+n\alpha^{*}$ for all n, and the integral in $J(\alpha,\beta)$ sums the sampled terms between $-\frac{T}{2}$ and $\frac{T}{2}$. For any given T, there are $\frac{T}{\alpha^{*}}$ such samples (drawn from $\frac{T}{\alpha}$ periods of $w(t)$). Thus

$$J(\alpha,\beta)=\lim_{T\to\infty}\frac{1}{T}\sum_{k=-\frac{T}{2\alpha^{*}}}^{\frac{T}{2\alpha^{*}}}w((\beta^{*}-\beta+k\alpha^{*})\mod\alpha).$$

Let $\bar{w}(t)=w(\frac{t}{\alpha}+\frac{1}{2})$ be a time-scaled version of $w(t)$ with support in (0,1). Then the cost can be written in standard form

$$J(\alpha,\beta)=\frac{1}{\alpha^{*}}\lim_{N\to\infty}\frac{1}{N}\sum_{k=-N/2}^{N/2}\bar{w}((x+k\frac{\alpha^{*}}{\alpha})\mod 1)\tag{23}$$

where $x=\frac{\beta^{*}-\beta}{\alpha}+\frac{1}{2}$. If $s=\frac{\alpha^{*}}{\alpha}$ is irrational, Khinchin's equidistribution theorem [22] shows that

$$J(\alpha,\beta)=\frac{1}{\alpha^{*}}\int_{0}^{1}\bar{w}(t)dt,\tag{24}$$

which holds for almost all x and any Lebesgue integrable function $\bar{w}(t)$. More interesting are the special cases when $s=\frac{p}{q}$ is rational for small integers p and q.

Case 1: When $p=q=1$, $\bar{w}((x+k)\mod 1)=\bar{w}(x)$ for every integer k. Hence, from (23), $J(\alpha,\beta)$ is maximized when $x=\frac{1}{2}$, i.e., when $(\alpha,\beta)=(\alpha^{*},\beta^{*})$. Thus $J(\alpha,\beta)=\bar{w}(\frac{1}{2})=w(0)$ and this is the global maximum.

Case 2: When $p>1,q=1$, $\bar{w}((x+pk)\mod 1)=\bar{w}(x)$ for every integer k. Again, $J(\alpha,\beta)$ is maximized when $x=\frac{1}{2}$ and in this case $J(\alpha,\beta)=\frac{\bar{w}(\frac{1}{2})}{p}=\frac{w(0)}{p}$. These represent $p:1$ synchronizations of the input and output.

Case 3: When $p=1,q=2$, the sum (23) contains exactly two terms per period, separated by $\frac{1}{2}$. The maximum occurs at the x that maximizes $\bar{w}(x)+\bar{w}((x+\frac{1}{2}\mod 1))$. Thus the stable points depend on the details of the wavetable function. For example, if the wavetable is sinusoidal $\bar{w}(x)=\sin(\pi x)$, it is necessary to find the minima of $\bar{w}_2(x)=\sin(\pi x)+\sin(\pi(x+\frac{1}{2})\mod 1)$. Taking the derivative, setting it to zero, and solving shows that $\frac{d\bar{w}_2(x)}{dx} = 0$ at $x = \frac{1}{4}$. Thus for $1:2$ synchronization with a sinusoidal wavetable, the algorithm is stable to a point $1/4$ cycle removed from the expected $x = \frac{1}{2}$ and the value of the cost is $\frac{1}{2}(\bar{w}(\frac{1}{4}) + \bar{w}(\frac{3}{4})) = \frac{\sqrt{2}}{2}$. If the wavetable $\bar{w}(x) = e^{-20(x-\frac{1}{2})^2}$ is a narrow Gaussian pulse centered at $\frac{1}{2}$, the optimal point occurs very close to $x = \frac{1}{2}$ with value $\frac{1}{2}(\bar{w}(0) + \bar{w}(\frac{1}{2})) \approx \frac{1}{2}\bar{w}(\frac{1}{2})$. If the wavetable were linear with slope $\frac{1}{2}$ on $(0, \frac{1}{2})$, there would be no optimal point because $\bar{w}(x) + \bar{w}((x + \frac{1}{2} \mod 1)) = 1$ for all x. In this case, the algorithm does not have a stable $2:1$ synchronization.

Case 4: When $p = 1, q > 2$, the sum (23) contains exactly q terms per period $\bar{\bar{w}}(x) = \sum_{n=0}^{q-1} \bar{w}(x + \frac{n}{q} \mod 1)$ and the cost can be found by calculating the x that maximizes $\bar{\bar{w}}(x)$. For example, with a sinusoidal wavetable, the $q = 3$ synchronization occurs at the maximum of $\sin(\pi x) + \sin(\pi(x + \frac{1}{3}) \mod 1) + \sin(\pi(x + \frac{2}{3}) \mod 1)$. The optimum x is $\frac{1}{6}$ and the cost is $\frac{1}{3}(\bar{w}(\frac{1}{6}) + \bar{w}(\frac{1}{2}) + \bar{w}(\frac{5}{6})) = \frac{2}{3}$. The case for general p and q is similar, and the behavior depends on the details of the particular wavetable function used.

There are two factors limiting the complexity of the $p:q$ synchronizations in practical situations. First, the maximum value of the cost at the optimal points becomes smaller (even if it does exist) when q becomes large (because of the $\frac{1}{N}$ term). When p becomes large, the cost becomes smaller because of the $\frac{1}{\alpha^*}$ term in (23). Finally, whenever the maximum value of the $\frac{p}{q}$ cost slips below the value for irrational $\frac{\alpha^*}{\alpha}$ in (24), then it is no longer a local maximum.