Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

Simultaneous Perturbation Extremum Seeking Method for Dynamic Optimization Problems

Nusawardhana* and S.H. Żak†

*School of Aeronautics and Astronautics, Purdue University, IN 47907, USA. Email: nusaward@ecn.purdue.edu

†School of Electrical and Computer Engineering, Purdue University, IN 47907, USA. Email: zak@purdue.edu

***Abstract*— A method of dynamic optimization using simultaneous perturbation principle is proposed and illustrated with numerical examples. The method is developed from a simultaneous-perturbation stochastic approximation (SPSA) recursive algorithm and is intended for problems with continuous measurements. The method is suitable for large scale dynamic optimization problems. Problems requiring continuous measurements appear, for example, in the area of neurocontrol, where, for some algorithms, the updates of the network gains depend on the integral of the square tracking error with respect to time.**

A method of dynamic optimization using simultaneous perturbation principle is proposed and illustrated with numerical examples. The method is developed from a simultaneous-perturbation stochastic approximation (SPSA) recursive algorithm and is intended for problems with continuous measurements. The method is suitable for large scale dynamic optimization problems. Problems requiring continuous measurements appear, for example, in the area of neurocontrol, where, for some algorithms, the updates of the network gains depend on the integral of the square tracking error with respect to time.

I. Motivation and Problem Statement

A common objective in stochastic approximation problems is to minimize a cost function, $J(x)$, whose exact analytic expression is unknown but whose noisy measurements are available. To solve this class of problems, we usually employ recursive gradient-descent type of algorithms, which are classified based on the methods used to determine descent directions. An algorithm known as the finite-difference algorithm uses $2n$ perturbations of $J(x)$, $x \in R^n$, two perturbations for each of the n variables, to determine a descent direction. On the other hand, a simultaneous perturbation stochastic approximation (SPSA) recursive algorithm needs only two perturbations of the decision vector $x \in R^n$. The perturbation used in this algorithm is random. Descent direction is consequently randomly determined. However, in each step of the search process, the algorithm uses a method to generate a descent direction. Analytical studies have shown that the SPSA algorithm provides a significant improvement in terms of the number of required computations for large scale optimization problems. Impressive results of the use of the SPSA on large scale problems can be found in [10], [11].

In this paper, we present a SPSA-based approach to continuous time dynamic optimization problems. Specifically, we address the question of how to design a network that employs the SPSA principle to solve large scale dynamic optimization problems using continuous time measurements. Our proposed approach can be employed to the design of a neurocontroller or an approximating architecture for solving approximate dynamic programming problems. We will elaborate on the term dynamic optimization in the problem statement below.

We consider a class of dynamic optimization problems commonly referred to as extremum seeking problems. We limit ourselves to the dynamic optimization problems with the following properties: (i) for the cost function $J(x(t))$, when $x(t)$ is held constant there exists a steady-state time T such that for $t \geq T$, J is almost constant, that is, $|\frac{d}{dt}J| \leq \epsilon_{\delta J}$ where $\epsilon_{\delta J}$ is a small positive number. We assume that after T time units, the cost $J(x(t))$ is constant as long as its argument $x(t)$ is held constant; (ii) J is a continuously differentiable function with respect to x, satisfying the condition, $\|\nabla J(x(kT)) - \nabla J(y(kT))\| \leq L\|x(kT) - y(kT)\|$, for some positive constant L. Another property of dynamic optimization problems that is similar to the stochastic approximation problems is that an exact analytic form of J is not known, but the values of J can be measured.

The cost function J can be regarded as a dynamic-map with n-inputs, $x(t) = [x_1(t) \cdots x_n(t)]^T$, and one output. For this reason we sometimes denote the dynamic cost as $J(t)$.

II. Definitions and Notation

The following notation is used throughout the paper. A space of real vectors is denoted R^n. A vector of real parameters is denoted as $x \in R^n$. The $i-$th element of x is denoted as x_i. An inner product of two vectors x and y in R^n is given by $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$. The norm of x, $\|x\|$, is given by $\langle x, x\rangle^{\frac{1}{2}}$.

A sequence $\{x_1, x_2, x_3, \cdots\}$, $x_i \in R$, in which each term is defined sequentially at T time units apart can be regarded as a discrete-time signal/function x_k. An inner product of two discrete time signals x_k and y_k over the interval $(nT, mT]$, $n < m$, is given as $\langle x_k, y_k\rangle = \sum_{k=n+1}^{k=m} x_k y_k$. This signal space is also known as $l_2(nT, mT]$.

Consider a space of continuous integrable functions in an interval $[a, b]$ denoted as $L_2[a, b]$. An element in this space is denoted as $x(t) \in L_2[a, b]$. An inner product of $x(t)$ and $y(t)$ in $L_2[a, b]$ is defined as $\langle x(t), y(t)\rangle = \int_a^b x(t)y(t)dt$. The norm of $x(t)$, $\|x(t)\|$, is given by $\langle x(t), x(t)\rangle^{\frac{1}{2}}$.

Two vectors x and y, where $x \in R^n$ or $x \in L_2[a, b]$ are orthogonal if $\langle x, y\rangle = 0$. Two vectors x and y are almost orthogonal if $|\langle x, y\rangle| \leq \epsilon$, where ϵ is a small positive number.

Two vectors x and y are aligned if $\langle x, y\rangle = \|x\|\|y\|$. Two vectors x and y (or, $x(t)$ and $y(t)$) are almost aligned if $|\langle x, y\rangle - \|x\|\|y\|| \leq \epsilon$, where ϵ is a small positive number.

The notions of inner product, norm, orthogonality and alignment, as well as the parallelogram law are discussed in more detail in [4].

A piecewise constant signal $x_k(t)$ on the interval $[kT, (k+1)T)$ is defined as $x_k(t) = x_k$, $kT \leq t < (k+1)T$. This signal is held constant over this interval. Note that

$x_k \in R^n$. Therefore, $x_k(t)$ is a vector whose n components are piecewise constant functions.

For a function $J : R^n \rightarrow R$, a pseudo-gradient of $J(x)$ at a point $\bar{x}$ is a vector $s(\bar{x}) \in R^n$ such that $\langle \nabla J(\bar{x}), s(\bar{x}) \rangle > 0$, i.e., the vector $s(\bar{x})$ forms an acute angle with the gradient of J at the point of $\bar{x}$. The vector $-s(\bar{x})$ is consequently a descent direction. For a discussion pertaining to the pseudo-gradient and its use in optimization problems, we refer to [7].

A nonlinear dynamical system with the state vector $z \in R^m$, input $u \in R^n$, and output $J \in R$, described by the equations,

$$\begin{aligned} \dot{z}(t) &= f_J(z(t), u(t)) && (1) \\ J(t) &= h_J(z(t), u(t)), && (2) \end{aligned}$$

is in an equilibrium state whenever $f_J(z(t), u(t)) = 0$. At the equilibrium state, the corresponding state and input vectors are denoted z_o and u_o. The equilibrium state z_o can be expressed as a function of u_o, $z_o = \phi_{zu}(u_o)$. Therefore, $0 = f_J(z_o, u_o) = f_J(\phi_{zu}(u_o), u_o)$. When there exists small variation in u_o, and another equilibrium state is reached, the variation of f_J is given by

$$0 = \delta f_J(z_o, u_o) = \frac{\partial f_J}{\partial z}\frac{\partial \phi_{zu}}{\partial u}\delta u + \frac{\partial f_J}{\partial u}\delta u. \quad (3)$$

The gradient of J with respect to the input variable u evaluated at the equilibrium input u_o is denoted $\nabla_u J$ and is given by

$$\nabla_u J(u_o)^T = \frac{\partial h_J}{\partial z}\left(-\frac{\partial f_J}{\partial z}\right)^{-1}\frac{\partial f_J}{\partial u} + \frac{\partial h_J}{\partial u}, \quad (4)$$

where the second equality is obtained by computing $\partial z/\partial u$ from (3). Note that the partial derivatives: $\partial h_J/\partial z \in R^{1\times m}$, $\partial h_J/\partial u \in R^{1\times n}$, $\partial f_J/\partial z \in R^{m\times m}$, $\partial f_J/\partial u \in R^{m\times n}$ are evaluated at z_o and u_o. When u is perturbed by δu, the variation of $J(t)$ due to the perturbation δu is given by

$$J(t) = h_J(z_o, u_o) + \frac{\partial h_J}{\partial z}\delta z + \frac{\partial h_J}{\partial u}\delta u + o(\|\delta u\|^2), \quad (5)$$

where δz is the solution of the differential equation,

$$\frac{d}{dt}\delta z = \frac{\partial f_J}{\partial z}\delta z + \frac{\partial f_J}{\partial u}\delta u. \quad (6)$$

When $\partial f_J/\partial z$ is stable and invertible, the solution is given by

$$\delta_z(t) = \int_{t_o}^{t} e^{\frac{\partial f_J}{\partial z}(t-\tau)} d\left(-\frac{\partial f_J}{\partial z}\tau\right)\left(-\frac{\partial f_J}{\partial z}\right)^{-1}\frac{\partial f_J}{\partial u}\delta u. \quad (7)$$

Note that all partial derivatives in equations (5) and (6) are evaluated at z_o and u_o, and the perturbation δu is constant on the time interval $[t_o, t]$. We mention that the above manipulations follow the arguments that can be found, for example, in [3, pages 52–57].

III. Simultaneous Perturbation Extremum Seeking Network

In this section, we present the development of a *simultaneous perturbation extremum seeking* (SPES) network using continuous measurements.

A. The SPSA Algorithm in an Inner Product Form

The SPSA algorithm that can be used to recursively estimate the minimizer of a cost function $J(x)$ based on its noisy measurements has the form ([15]),

$$x_{k+1} = x_k - a_k \frac{y_k^+ - y_k^-}{2c_k} d_k, \quad (8)$$

where both a_k and c_k are positive scalars, d_k is a vector of the same dimension as x, y_k^+ and y_k^- are one dimensional vectors (that is, scalars) of noisy measurements of the function J perturbed at two distinct points. In the SPSA algorithm, these measurement vectors are defined by

$$\begin{aligned} y_k^+ &= J(x_k + c_k d_k) + e_k^+, && (9) \\ y_k^- &= J(x_k - c_k d_k) + e_k^-, && (10) \end{aligned}$$

where e_k^+ and e_k^- are additive measurement errors. We note that, in the above algorithm, the cost function is static, that is, $J : R^n \rightarrow R$. A distinctive feature of the SPSA algorithm is that $d_k \in R^n$ is a random direction vector with $d_{k_i} = \pm 1$ generated using the symmetric Bernoulli distribution.

Remark 1: In [15] and [9], the measurements $y_k^{+/-}$ are represented as $J(x_k \pm c_k r_k)$, where $r_{k_i} = (d_{k_i})^{-1}$. However, because $d_{k_i} = \pm 1$, we have $d_{k_i} = (d_{k_i})^{-1} = r_{k_i}$.

To proceed, we represent the SPSA algorithm as follows,

$$\begin{aligned} x_{k+1} &= x_k - a_k \frac{y_k^+ - y_k^-}{2c_k} d_k \\ &= x_k - \frac{a_k}{2c_k}\left\langle \underbrace{\begin{bmatrix} y_k^+ \\ y_k^- \end{bmatrix}}_{=y_k}, \underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{=d_{m_k}} \right\rangle d_k. \end{aligned}$$

The recursive algorithm now constitutes a discrete-time dynamic system for parameter update with an inner product term of two vectors in R^2. The elements of y_k, which are y_k^+ and y_k^-, are sequentially measured every T time units.

We next express the recursive algorithm SPSA in terms of an inner product of two discrete-time signals in the interval $(kT, (k+2)T]$, which will then be used to devise an SPSA-type of algorithm for the elements in $L_2[a, b]$ on the interval $(kT, (k+2)T]$. To proceed, consider two discrete-time sequences, $y = \{\cdots y_{k-1}^+, y_{k-1}^-, y_k^+, y_k^-, y_{k+1}^+, y_{k+1}^- \cdots\}^T$, which is a sequence of measurements, and an alternating sequence $d_m = \{\cdots, +1, -1, +1, -1, \cdots\}^T$. The above two sequences constitute discrete-time signals. Let $y_k = \left[y_k^+, y_k^-\right]^T$ be a segment of discrete-time signal y, where y_k^+ is available prior to y_k^-. Similarly, let $d_{m_k} = [+1, -1]^T$ be a segment of d_m. Using the discrete-time segment signals, y_k and d_{m_k}, the recursive algorithm SPSA can be represented in terms of an inner product of two discrete-time signals on the time interval $(kT, (k+2)T]$ as follows,

$$x_{k+1} = x_k - \frac{a_k}{2c_k}\langle y_k, d_{m_k}\rangle d_k,$$

where the inner-product in the above algorithm is defined on $l_2(kT, (k+2)T]$. Note that the parameter update period is $2T$.

We now wish to extend the above paradigm to the case in which we use continuous time measurements and a periodic alternating signal to construct an inner product term in the recursive equation. We define the following signals,

$$d_m(t) = \begin{cases} +1, & \text{for} \quad kT \le t < (k+1)T \\ -1, & \text{for} \quad (k+1)T \le t < (k+2)T \end{cases} \tag{11}$$

and

$$y_k(t) = \begin{cases} y_k^+, & \text{for} \quad kT \le t < (k+1)T \\ y_k^-, & \text{for} \quad (k+1)T \le t < (k+2)T \end{cases} \tag{12}$$

Using the above two signals, we express the SPSA algorithm as

$$x_{k+1} = x_k - \frac{a_k}{2c_k}\eta_k \langle y_k(t), d_m(t)\rangle d_k, \tag{13}$$

where η_k is a correction factor used to maintain equality between equations (8) and (13). In equation (8), the inner-product is on R^2, while in equation (13) the inner-product is in $L_2[kT,(k+2)T]$. The update period of this discrete-time procedure remains $2T$. The subscript k in the signal $y_k(t)$ indicates that it is a piecewise constant signal. (Note that, in the section on the dynamic optimization network development, we use $y(t)$ instead of $y_k(t)$ to emphasize the fact that we deal with continuous signals.)

Because a piecewise constant signal $y_k(t)$ is used in the recursive equation (13), we need to modify the arguments of the cost function J. In equations (9) and (10), the arguments can be viewed as discrete time signals. To generate the piecewise constant signal, $y_k(t)$, the arguments of J must also be piecewise constant signals,

$$y_k(t) = J(x_k(t) + c_k d_k d_m(t)), \tag{14}$$

where $x_k(t)$ is a piecewise constant signal, $d_k \in R^n$ is a vector of random direction whose elements are ± 1 and are generated using the symmetric Bernoulli distribution as before, and $d_m(t)$ is a square wave signal (see equation (11)) with a period $2T$.

The signal $d_k d_m(t)$ is a square wave $d_m(t)$ modulated by d_k, a random direction vector. This randomly modulated signal is a distinctive feature of the proposed network. The signal $d_k d_m(t)$ simultaneously perturbs all arguments of $J(x_k(t))$. The perturbation of $J(x_k(t))$ in this fashion can also be interpreted as a single-frequency perturbation by a vector of square waves with uniform magnitudes and random phase shifts randomly selected using the Bernoulli distribution from the set $\{0, \pi\}$.

In the above discussions, the cost function J was considered a static map that exhibits no transient. As mentioned earlier, problems in dynamic optimization or extremum seeking are no longer concerned with static cost functions. In dynamic optimization problems transients are present, the shape of the cost function, or the performance surface, changes with time. In the next section, we present methods that allow us to employ the framework of the SPSA to the extremum seeking problems using continuous measurements.

B. SPES Network Using Continuous Measurements

The SPSA algorithm belongs to a class of algorithms that employ a pseudo-gradient in search of an optimizer. Perturbations of the cost function are necessary to obtain a proper sign of d_k so that x_k progresses along a descent direction. In the problems of extremum seeking, J is viewed as an output of a dynamic system and, as such, it is a function of time, that is, $J = J(t)$. A dynamic system that generates $J(t)$ is described by equations (1)–(2) where the input is $x_k(t)$. The function $J(t)$ driven by the input $x_k(t) + c_k d_k d_m(t)$ is no longer a piecewise constant signal.

To proceed, we expand $J(t)$ into a Taylor series, on the interval $[kT, t]$, driven by the input $x_k(t) + c_k d_k d_m(t)$,

$$J(t) = J(kT) + c_k d_m(t) d_k^T \nabla J(kT) + \delta_{tr}(t) + o(\|d_m(t)\|^2), \tag{15}$$

where the gradient $\nabla J(kT)$ corresponds to the input $x_k(kT) = x_k$, and is given by $\nabla J(kT)^T = \frac{\partial h_J}{\partial z}\left(-\frac{\partial f_J}{\partial z}\right)^{-1}\frac{\partial f_J}{\partial u} + \frac{\partial h_J}{\partial u}$. The transient term δ_{tr} is given by $\delta_{tr}(t) = -\nabla J(kT)^T c_k d_k d_m(t) + \left[\int_{kT}^{t} e^{\frac{\partial f_J}{\partial z}(t-\tau)} d\left[-\frac{\partial f_J}{\partial z}\tau\right]\left[-\frac{\partial f_J}{\partial z}\right]^{-1}\frac{\partial f_J}{\partial u}\right] c_k d_k d_m(t)$. The above three expressions are derived from equations (5), (4) and (7), where we set $u = x_k$ and $\delta u(t) = c_k d_k d_m(t)$. Note that at $t = kT$ the system generating $J(t)$ is in an equilibrium state. Each term on the right side of (15) is assumed to be continuous. The transient portion of $J(t)$ is lumped in $\delta_{tr}(t)$. Each signal with the transient present in it is decomposed into two components, the transient and steady-state components, that is, $J(t) = J_{tr}(t) + J_{ss}(t)$. We assume that the length of the transient is $T_{tr} = \lambda T$, $\lambda \in (0,1)$. We design a continuous time signal $d_d(t)$ so that it is orthogonal to the signal $J_{tr}(t)$ and aligned with $J_{ss}(t)$. The signal $d_d(t)$ must also be orthogonal to constant signals. Multiplying both sides of equation (15) by $d_d(t)$ and integrating the result from $t_i = kT$ to $t_f = (k+2)T$, we obtain

$$\int_{t_i}^{tf} J(t) d_d(t) dt = 2(1-\lambda) T c_k d_k^T \nabla J(kT) + \Delta_{tr}(t), \tag{16}$$

where the first term is obtained from $c_k d_k^T \nabla J(kT) \int_{t_i}^{tf} d_d(t) d_m(t) dt$ and the second term $\Delta_{tr}(t)$ is an integral of the transient portion of $J(t)$, $\Delta_{tr}(t) = \int_{t_i}^{tf} d_d(t)\delta_{tr}(t) dt$. The magnitude of the integral of the transient is bounded, that is, $|\Delta_{tr}(t)| \le \epsilon_{tr}$. Note that the terms $J(kT)$ and $\nabla J(kT)$ on the right hand side of equation (15) are constant on the interval $[kT,(k+2)T]$ because they are determined from the equilibrium state at $t = kT$ (see equations (5), (4), (7)). Also note that the inner product of $d_d(t)$ with constant signals is zero on the interval $[kT, t]$. For this reason, we exclude $\nabla J(kT)$ from the integral and remove constant terms from equation (16).

The integration in equation (16) constitutes demodulation as well as an inner product of $J(t)$ and $d_d(t)$. Through this inner product, we obtain the information of directional derivative along d_k. Recall that the recursive SPSA algorithm has an inner product term. Substituting the inner product of equation (16), gives an algorithm/network that can be applied to solve continuous time dynamic optimization problems. A block diagram of the simultaneous perturbation extremum network is shown in Figure 1.

The network is described by the equations,

$$\begin{aligned} x_k(t) &= x_k, \qquad kT \le t < (k+2)T && (17) \\ y(t) &= J(t) + v(t) && (18) \\ \widehat{\nabla} J(kT)\Big|_{u=x_k(t)} &= s_k + w_k && (19) \\ s_k &= 2(1-\lambda) T c_k d_k^T \nabla J(kT) d_k \\ w_k &= \Delta_{tr}(t) d_k \end{aligned}$$

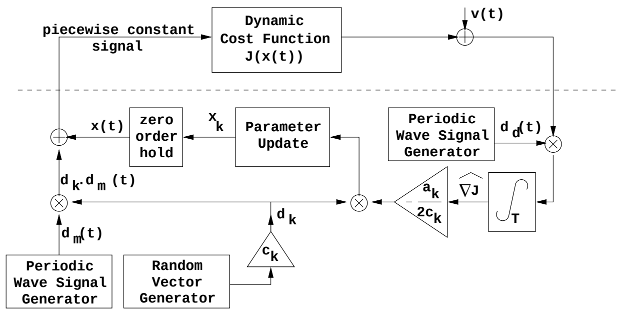
piecewise constant signal
Dynamic Cost Function J(x(t))
v(t)
x(t)
zero order hold
x k
Parameter Update
Periodic Wave Signal Generator
d d(t)
d k.d m (t)
a k / 2c k
∇J
T
d k
d m(t)
c k
Periodic Wave Signal Generator
Random Vector Generator

Fig. 1. A simultaneous perturbation extremum seeking network.

$$+\left(\int_{kT}^{(k+2)T} v(t)d_d(t)dt\right) d_k$$
$$x_{k+1} \quad = \quad x_k - \gamma_k(s_k + w_k), \tag{20}$$

where $\gamma_k = a_k/2c_k$ constitutes parameter update gain, $\widehat{\nabla} J(kT)\Big|_{u=x_k}$ is a descent direction vector of the dynamic cost function $J(kT)$ due to input $u = x_k(t)$ corrupted by the measurement noise $v(t)$ and by the computational errors, s_k is the pseudo-gradient vector, $-s_k$ is the descent direction vector, and w_k lumps both the effect of the measurement noise and δ_{tr} from equation (16).

Remark 2: It was mentioned earlier that the perturbation method used here can be viewed as a single-frequency perturbation of n-variables. This approach differs from the existing extremum-seeking methods (for example, in [1], [6]) which employ multi-frequency sinusoidal signals. Using only one single-frequency wave generator significantly simplifies the design process as well as the resulting network implementation.

IV. Convergence Analysis

In this section, we analyze convergence properties of the proposed algorithm. The algorithm convergence is shown in two stages. First, we show that the method is capable of extracting the pseudo-gradient from the dynamic cost function. Next, we show that the discrete-time recursive parameter update generates a sequence such that x_k approaches a stationary point, that is, $\lim_{k\to\infty} x_k = x^o$, where $\nabla J(x^o) = 0$.

Proposition 1: (Pseudo-Gradient Extraction) If c_k is selected so that $c_k\|\nabla J(kT)^T d_k\| > \frac{\epsilon_{tr}}{2(1-\lambda)T}$, then the demodulation, or the inner-product, of $J(x_k(t) + c_k d_k d_m(t))$ with $d_d(t)d_k$ yields a pseudo-gradient vector.

Proof: See [5] ■

Proposition 1 indicates the minimum value of c_k for a suitable pseudo-gradient generation. When this condition is violated, the demodulation process does not yield a descent direction. It is also important to note that as x_k approaches the neighborhood of $\{x^o : \nabla J(x^o) = 0\}$, ϵ_{tr} also decreases. Theorems on stochastic approximation problems in [8], [10], [15], generally require that as $x_k \to x^o$, $c_k \to 0$. This implies that the gain c_k should not be too large for the sake of convergence, yet at the same time, it should not be too small for the sake of generating a descent direction.

We can perform the convergence analysis of the recursive parameter update using the results from the field of stochastic approximation. Convergence in a recursive stochastic-approximation is determined by proper selection of gains for perturbation as well as parameter update under the influence of the measurement noise/error. Convergence results of stochastic approximation methods require that diminishing gains should be used. Only under certain circumstances, constant update gain is appropriate.

The proposition below gives conditions that guarantee the convergence of the proposed method. The result is of deterministic nature and it is adapted from the deterministic analysis of convergence of the gradient method with the error as presented in [2]. The results in [14], [15] also provide a deterministic convergence analysis of the recursive stochastic approximation algorithms.

Proposition 2: (SPES Convergence Using Sequence of Decreasing Gains) In a simultaneous perturbation extremum seeking method with continuous measurements given by equations (17)–(20) used to minimize the dynamic performance surface $J(t)$, if

1) $J_k = J(x_k) = J(t)|_{t=2kT}$, $\nabla J_k = \nabla J(x_k) = \nabla J(t)\Big|_{t=2kT}$,
2) $\|\nabla J(x_k) - \nabla J(x_{k+1})\| \le L\|x_k - x_{x+1}\|$,
3) $c_1\|\nabla J(x_k)\|^2 \le \nabla J(x_k)^T s_k$,
4) $\|s_k\| \le c_2(1 + \|\nabla J(x_k)\|)$,
5) the update gain γ_k is positive and satisfies $\sum_{k=0}^{\infty}\gamma_k = \infty$, $\sum_{k=0}^{\infty}\gamma_k^2 < \infty$,

then the network generates a sequence of x_k such that either $J(x_k) \to -\infty$ or else $J(x_k)$ converges to a finite value and $\lim_{k\to\infty}\nabla J(x_k) = 0$. Furthermore, every limit point of x_k is a stationary point of $J(x_k)$.

Proof: See [5] ■

We next consider a special case where the magnitude of w_k is bounded by the magnitude of $\nabla J(x_k)$, for which we can use constant gain $\gamma_k = \gamma$.

Proposition 3: (SPES Convergence Using Constant Gain) In a simultaneous perturbation extremum seeking method with continuous measurements given by (17)–(20), $\gamma_k = \gamma$, used to minimize the dynamic performance surface $J(x_k(t))$, if conditions 1–3 from Proposition 2 hold and

4) $\|s_k\| \le c_2(\|\nabla J(x_k)\|)$,
5) $\|w_k\| < c_3\|\nabla J(x_k)\|$, $c_3 < c_1$,
6) the update gain γ satisfies $0 < \gamma \le \frac{c_1 - c_3}{(c_2^2 + c_3^2)L}$,

then the network generates (i) a descent direction, and (ii) a sequence of x_k such that either $J(x_k) \to -\infty$ or else $J(x_k)$ converges to a finite value and $\lim_{k\to\infty}\nabla J(x_k) = 0$. Furthermore, every limit point of x_k is a stationary point of $J(x_k)$.

Proof: See [5] ■

V. Numerical Experiments

In this section, we present two numerical experiments, both involving nonlinear dynamic problems, to illustrate the use of the SPES method to solving dynamic optimization problems. The first example illustrates the design process. The second example shows the use of the SPES method when applied to a large scale dynamic optimization problem.

The first problem uses the nonlinear dynamic system described by equations (5) in [12]. The system state vector is denoted $\xi \in R^4$ with inputs $x \in R^2$ and an output $y \in R$. For fixed x_1, x_2, and any initial condition such that $\xi_3(0) \neq$

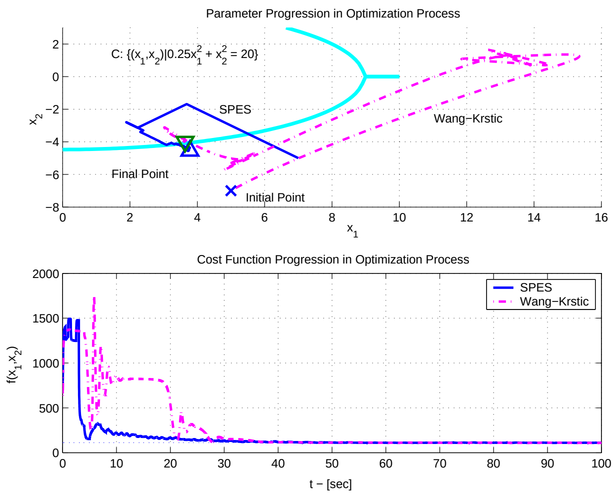

Fig. 2. Comparison of dynamic optimization using two extremum seeking methods.

0 and $\xi_4(0) \neq 0$, the states of the system converge to the attractor $\mathcal{A} = \{\xi \in R^4 : \xi_1 = \frac{1}{2}x_1, \xi_2 = x_2, \xi_3^2 + \xi_4^2 = \left(\frac{1}{4}x_1^2 + x_2^2 - 20\right) + 9\}$ which constitutes level sets. The set of minimizers for the performance surface is given by $\mathcal{C} = \left\{(x_1, x_2) \in R^2 : \frac{1}{4}x_1^2 + x_2 = 20\right\}$. The objective of the design process is to build an optimizing network that generates sequences of variables x_1, x_2 that will drive the output of the nonlinear dynamical system to its minimal set.

To proceed with the design, we first characterize transient properties of the dynamic system. After collecting several step responses, we can determine that the period of $d_d(t)$ is $T = 4$, the period of $d_m(t)$ and the update rate of x_k satisfy $2T = 8$. The signal $d_d(t)$ is designed in such a way that it annihilates J_{tr}. The above observations indicate that the signal

$$d_d(t) = \begin{cases} 0, & kT < t \leq kT + 3 \\ 1, & kT + 3 < t \leq (k+1)T \\ 0, & (k+1)T < t \leq (k+1)T + 3 \\ -1, & (k+1)T + 3 < t \leq (k+2)T \end{cases}$$

is appropriate for demodulation purposes.

To determine the update gain γ_k, we can employ a low gain for γ and gradually increase it until we observe a satisfactory outcome. In this experiment, a fixed gain $\gamma = 1/300$ is chosen after several trials.

The gain c_k is determined as $c_k = 0.1$ because it gives sufficient distinction in the steady state responses, which is useful in determining the sign of the vector d_k.

Using the above design parameters, we perform the closed-loop simulation of the network depicted in Figure 1. As a benchmark, we simulate the closed-loop dynamics using the method of Krstić and Wang [13]. Simulation results of this experiment are summarized in Figure 2. In this figure, the traces of signals generated with the SPES method are denoted using solid lines with the label SPES, while those generated using the method of Krstić and Wang are denoted using the dash-dot lines with the label Wang and Krstić.

The upper plot shows the paths of the parameters in the parameter space starting from the initial condition (denoted "x") towards the set of optimizing parameters (labelled $\mathcal{C}$ and marked with the grey-color oval). Both methods generate sequences of parameters that converge to $\mathcal{C}$. However, it is noticeable that the sequence generated using the method of Krstić and Wang takes a longer path to arrive to the neighborhood of $\mathcal{C}$. The lower plot shows the time history of $y(t) = J(t)$ as the search for the optimizer progresses. As the sequence generated by the SPES method takes a somewhat shorter path to reach the set $\mathcal{C}$, the corresponding value of $y(t)$ converges to the neighborhood of the minimum faster.

The second example is concerned with the minimization of the phase-portrait orbital size. This problem is discussed in detail in [13]. The problem can also be viewed as the problem of the limit-cycle amplitude minimization. We use a second-order nonlinear system, known as the Van der Pol system, which is described by the equation,

$$\ddot{w} + \epsilon\left[(w - w_0)^2 - 1 - f(x)\right]\dot{w} + \mu^2(w - w_0) = 0,$$

where $x \in R^n$, $f(x) = (\sum_{i=1}^n x_i + x_0)^2$. Four different cases are investigated $n = 1, 2, 3, 4$. The setup for this experiment for each case is summarized in Table I.

TABLE I

Gains for different cases in the second experiment.

n	$\sum_{i=1}^n x_i(0)$	x_0	a_k	c_k	T
2	0.5	3	1	0.1	10
10	0.5	3	1/5	0.1	10
30	0.5	3	1/15	0.1	10
100	0.5	3	1/50	0.1	10

The results of numerical experiments are plotted in Figures 3 and 4 for different cases with different dimension of x. Simulation results indicate that, as the number of the parameters increases from 2 to 100, the simultaneous perturbation extremum network can minimize the limit cycle amplitude at a roughly similar rate irrespective of the number of parameters. Figure 3 shows the evolution of the orbit size (in the phase-plane, that is, the w-$\dot{w}$ plane) as the search for the optimum progresses. Figure 4 shows the time history of the limit cycle amplitude, $w(t)$. As the number of the parameters increases, the smoothness of the steady-state minimum limit cycle amplitude decreases when a constant gain is used in the update mechanism. We conjecture that a smoother result can be achieved with decreasing the gain as the limit-cycle magnitude is in the neighborhood of its minimum.

VI. Acknowledgement

We would like to thank Dr. J. C. Spall for his suggestions and encouragement, Professor W. A. Crossley for his support, and Professor M. J. Corless for his feedback on technical issues.

References

[1] K.B. Ariyur and M. Krstic. Analysis and design of multivariable extremum seeking. In *Proceedings of the American Control Conference*, volume 4, pages 2903–2908, May 2002.

[2] D.P. Bertsekas and J.N. Tsitsiklis. Gradient convergence in gradient methods with errors. *SIAM Journal on Optimization*, 10(3):627–642, 2000.

[3] G.C. Goodwin, S.F. Graebe, and M.E Salgado. *Control System Design*. Prentice-Hall, New Jersey, 2001.

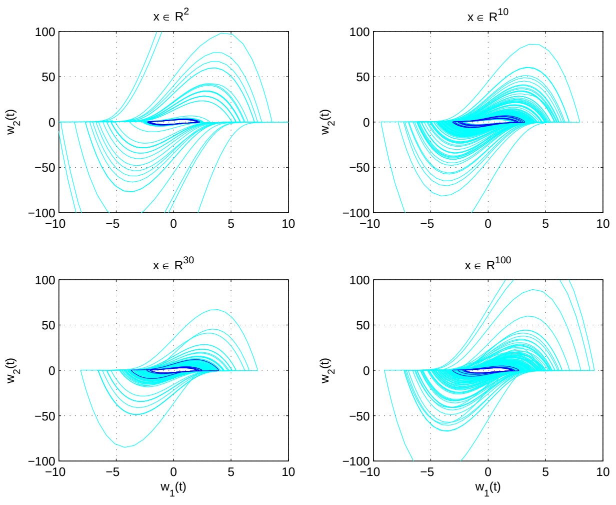

Fig. 3. The orbit size of the Van der Pol system in the w_1-w_2 plane, where $w_1 = w$ and $w_2 = \dot{w}$, for different values of n. The optimization problem is to find a set of parameters $\{x_1 \cdots x_n\}$ that minimize the orbit size. The optimization is performed using the proposed method in this paper. For different values of n, the minimum orbit size is shown using a bold line.

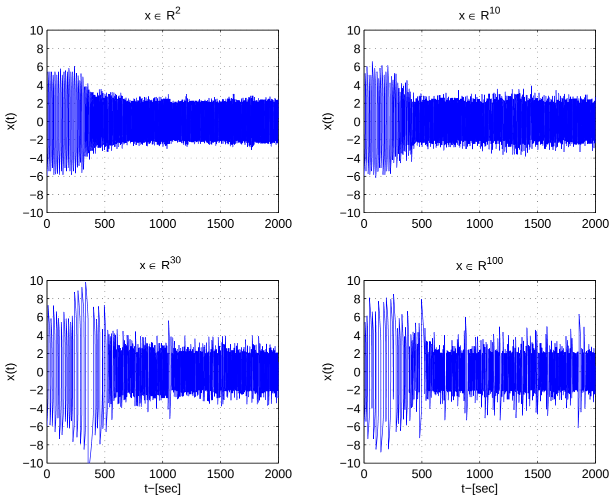

Fig. 4. The limit-cycle amplitude in the minimization problem for different values of n.

[4] D.G. Luenberger. *Optimization by Vector Space Methods,*. John Wiley & Sons, 1969.

[5] Nusawardhana and S.H. Żak. A simultaneous perturbation extremum seeking method for large-scale problems. *(submitted for publication)*, 2004.

[6] V.K. Obabkov. Theory of multichannel extremal control systems with sinusoidal probe signals. *Automation and Remote Control*, 28:48–54, 1967.

[7] B.T. Poljak and Y.Z. Tsypkin. Pseudogradient adaptation and training algorithms. *Automation and Remote Control*, 34(3):377–397, August 1973.

[8] J.C. Spall. A stochastic approximation algorithm for large-dimensional systems in the kiefer-wolfowitz setting. In *Proceeding of the IEEE Conference on Decision and Control*, pages 1544–1548, Austin, TX, December 7-9 1988.

[9] J.C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823, July 1998.

[10] J.C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, 2003.

[11] J.C. Spall. SPSA: Selected references on theory, applications, and numerical analysis. http://www.jhuapl.edu/spsa/Pages/References-List_Ref.htm, updated May, 2003.

[12] A. R. Teel and D. Popović. Solving smooth and nonsmooth multivariable extremum seeking problems by the methods of nonlinear programming. In *Proceedings of the American Control Conference*, pages 2394–2399, Arlington, VA, June 2001.

[13] H.H. Wang and M. Krstić. Extremum seeking for limit cycle minimization. *IEEE Transactions on Automatic Control*, 45(12):2432–2432, 2000.

[14] I.-J. Wang. *Analysis of Stochastic Approximation and Related Algorithms*. PhD thesis, Purdue University, 1996.

[15] I.-J. Wang and E.K.P. Chong. A deterministic analysis of stochastic approximation with randomized directions. *IEEE Transactions on Automatic Control*, 43(12):1745–1749, December 1998.