Simultaneous Perturbation Stochastic Approximation with Norm-Limited Update Vector

Yosuke Tanaka [a], Shun-ichi Azuma [a], Toshiharu Sugie [a]

[a] *Graduate School of Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto 606-8501, Japan*

Abstract: This paper addresses the convergence of simultaneous perturbation stochastic approximation (SPSA) with a norm-limited update vector. We first illustrate an unstable solution of the standard SPSA algorithm and motivate to consider a modified version, where the norm of the update vector is limited to a certain value. Next, a result on the almost-sure convergence is presented by reducing the modified algorithm into the standard SPSA algorithm and restricting the probability distribution for the perturbation to a Bernoulli distribution. Finally, we apply the modified algorithm to a system identification problem to demonstrate its performance.

Keywords: simultaneous perturbation stochastic approximation, system identification

1 Introduction

The simultaneous perturbation stochastic approximation (SPSA) [1] is a well-known solution to model-free optimization problems. The main feature is that, in solving an optimization problem, the method uses not the closed-form expression of the objective function but a small number of *measurements* of the objective function. By focusing on the practical utility, it has been extensively studied so far. For instance, to enhance its applicability, the original algorithm is extended to the one-measurement version [2], the adaptive version [3], the global version [4], the one-sided version [5], and so on (see [6] for other variations). Also, it has been applied to a wide range of engineering problems, e.g., model-free control [7–9], adaptive control [10–12], image registration [13–15], neural networks [16–18], and multi-agent control [19,20].

For the optimization problem $\min_{x\in\mathbf{R}^n} J(x)$, the standard SPSA algorithm [1] is given by

$$x_{k+1} = x_k + h(x_k, r_k)$$

where x_k is the candidate of the solution, r_k is the random vector, and $h(x_k, r_k)$ is the update vector. If some conditions hold for J, h, r_k, and x_0, the sequence $x_1, x_2, \ldots$, which is generated by the algorithm, converges to a solution to the problem in a stochastic sense. However, since the SPSA algorithm is a solution to *model-free* optimization (i.e., where the closed-form expression of J is unknown), the conditions for J are not always checkable in practice. As the result, one has to use the algorithm without checking the conditions and often encounters an instability phenomenon. This instability problem is involved in not only the standard algorithm but also all the variations including the aforementioned extensions [2–5].

Throughout our experience in engineering applications (e.g., in [7]), there are two main reasons for the instability. First, the update vector $h(x_k, r_k)$ often becomes too large to execute the algorithm, which causes numerical instability. Second, the update vector $h(x_k, r_k)$ grows with $k \to \infty$ and eventually diverges. So it is reasonable to limit the quantity of the update, which motivates us to introduce the following modified algorithm [7] as an alternative:

$$x_{k+1} = x_k + \delta(h(x_k, r_k))$$

where δ is a saturation function. It has been shown in [7,21] that the modified version succeeds in some engineering problems, and thus the modified SPSA algorithm must be more practical. Moreover, this modification can be used for a variety of SPSA algorithms including the aforementioned extensions [2–5].

However, in spite of such usefulness, any theoretical result has never been provided so far. Although the modified version will be used without checking convergence conditions due to the same reason as above, it is quite important to prove that it solves optimization problems

Email addresses: `ytanaka@robot.kuass.kyoto-u.ac.jp` (Yosuke Tanaka), `sazuma@i.kyoto-u.ac.jp` (Shun-ichi Azuma), `sugie@i.kyoto-u.ac.jp` (Toshiharu Sugie).

under reasonable conditions which are (hopefully) similar to those for the standard SPSA algorithm.

In this paper, we thus analyze the convergence of the modified algorithm. First, we illustrate an unstable solution and motivate to consider the modified algorithm. Next, a result on the almost-sure convergence is presented. The key ideas to prove the convergence are (a) reducing the algorithm into the standard SPSA algorithm and (b) restricting the probability distribution for the perturbation to a Bernoulli distribution. Finally, the modified algorithm is demonstrated by applying to a system identification problem.

Notation: Let $\mathbf{R}$, $\mathbf{R}_+$, and $\mathbf{N}$ be the real number field, the set of positive real numbers, and the set of non-negative integers. We denote by 0 and $\mathbf{1}$ the zero scalar/vector and the vector whose all elements are one. For the vector x, we use $\|x\|$ and $\mathrm{sgn}(x)$ to express the Euclidean norm and the vector obtained by applying the signum function to each element. For example, for $x = [3 \ \ -4 \ \ 0]^\top$, $\|x\| = 5$ and $\mathrm{sgn}(x) = [1 \ \ -1 \ \ 0]^\top$. If x contains no zero-valued element, we use $x^{(-1)}$ to represent the elementwise inverse, e.g., $x^{(-1)} = [1/4 \ \ -1/3]^\top$ for $x = [4 \ \ -3]^\top$. We denote by $A \otimes B$ the Kronecker product of the matrices A and B. Finally, let $\mathbb{P}(a)$ and $\mathbb{E}(\alpha)$ respectively denote the probability and the expectation for an event a and the random variable α.

2 Simultaneous Perturbation Stochastic Approximation [1]

In this section, we briefly review the SPSA algorithm [1] and illustrate its unstable solution.

2.1 Standard SPSA Algorithm

Consider the optimization problem

$$\min_{x \in \mathbf{R}^n} J(x) \tag{1}$$

where $x \in \mathbf{R}^n$ is the parameter to be optimized and $J : \mathbf{R}^n \to \mathbf{R}$ is the objective function.

A solution is given by the following algorithm:

$$x_{k+1} = x_k - a_k g(x_k) \tag{2}$$

where $x_k \in \mathbf{R}^n$ is the solution vector, $a_k \in \mathbf{R}_+$ is the gain, and $g(x_k) \in \mathbf{R}^n$ is the direction vector defined as

$$g(x_k) := \frac{J(x_k + c_k r_k) - J(x_k - c_k r_k)}{2c_k} r_k^{(-1)} \tag{3}$$

for a positive number $c_k \in \mathbf{R}_+$ and a random vector $r_k \in (\mathbf{R} \setminus \{0\})^n$. This algorithm is called the *Simultaneous perturbation stochastic approximation* (SPSA) [1].

The idea of this algorithm is that the expectation of $g(x_k)$ is nearly equal to $\frac{\partial J}{\partial x}(x_k)$ and thus (2) corresponds to a stochastic version of the steepest descent. Based on this fact, the almost-sure convergence to a local solution has been shown in [1] subject to several conditions.

2.2 Stable and Unstable Solutions

The following example demonstrates the algorithm in (2).

Example 1 *Consider*

$$J(x) := (x - \mathbf{1})^\top (x - \mathbf{1}) \tag{4}$$

where $n = 10$*. This has the unique stationary point at* $x = \mathbf{1}$*, which corresponds to the global minimum point. Fig. 1 shows the time evolution of* $J(x_k)/J(x_0)$ *for the algorithm with* $x_0 := [-0.14 \ \ -0.58 \ \ 1.07 \ \ -0.41 \ \ -0.26 \ \ 2.44 \ \ -1.29 \ \ -1.22 \ \ -0.87 \ \ -0.02]^\top$, $a_k := 0.05/(k + 200)^{0.602}$, $c_k := 0.01/(k+1)^{0.101}$*, and a probability distribution for* r_k*. It turns out that the algorithm solves the minimization problem.* □

However, the algorithm does not always solve the problem in (1).

Example 2 *Consider*

$$J(x) := \left((x - \mathbf{1})^\top (x - \mathbf{1}) \right)^3. \tag{5}$$

This objective function also has the unique stationary point at $x = \mathbf{1}$*, which is the global minimum point. Fig. 2 depicts the time evolution of* $J(x_k)/J(x_0)$ *for the algorithm under the same condition as in Example 1. Unlike Example 1,* $J(x_k)$ *does not converge to the minimum value, which implies that the algorithm could not solve the problem.* □

As shown in the above examples, the standard SPSA algorithm does not always give a stable solution, which motivates us to consider a modified version of the SPSA algorithm.

3 SPSA with Norm-limited Update Vector And Its Convergence

The main reason for the unstable solution is that the sequence $\|a_k g(x_k)\|$ $(k = 0, 1, \ldots)$ increases as $k \to \infty$ in some cases and x_k diverges as the result.

A solution to avoid such instability is to limit the amount of the update, i.e., $a_k g(x_k)$. So we consider the following algorithm, originally proposed in [7], as an alternative of (2):

$$x_{k+1} = x_k - \delta\left(a_k g(x_k)\right) \tag{6}$$

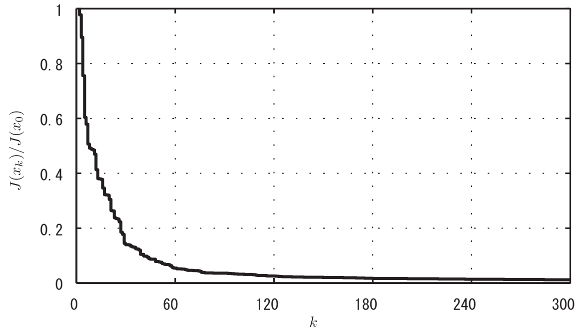

Fig. 1. Time evolution of $J(x_k)/J(x_0)$ in Example 1.

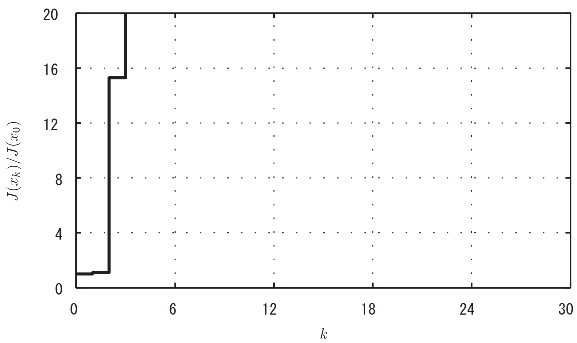

Fig. 2. Time evolution of $J(x_k)/J(x_0)$ in Example 2.

where $\delta : \mathbf{R}^n \to \mathbf{R}^n$ is given by

$$\delta(\theta) := \begin{bmatrix} \mathrm{sgn}(\theta_1)\min(|\theta_1|, d) \\ \mathrm{sgn}(\theta_2)\min(|\theta_2|, d) \\ \vdots \\ \mathrm{sgn}(\theta_n)\min(|\theta_n|, d) \end{bmatrix} \tag{7}$$

for $\theta \in \mathbf{R}^n$, the i-th element θ_i of θ, and a pre-specified positive number $d \in \mathbf{R}_+$. Note here that each element of the vector $\delta\left(a_k g(x_k)\right)$ is not greater than d.

For the proposed algorithm, the following result is obtained.

Theorem 1 *For the algorithm in (6), suppose that $x_0 \in \mathbf{R}^n$ and $d \in \mathbf{R}_+$ are given. Let $x^* \in \mathbf{R}^n$ be a local solution to the problem in (1). Then,*

$$\lim_{k\to\infty} x_k = x^* \quad \text{w.p.1} \tag{8}$$

subject to the following assumptions:

(C1) *For almost all x_k (at each $k \geq K$ for some $K < \infty$) and some $\alpha_0 > 0$, $J^{(3)}(x) := \partial^3 J/\partial x^\top \partial x^\top \partial x^\top$ exists continuously with individual elements bounded by α_0 for all x in an open neighborhood of x_k. Moreover, there exists an $\alpha_1 > 0$ such that $\mathbb{E}\left(J(x_k \pm c_k r_k)^2\right) \leq \alpha_1$ for all $k \in \mathbf{N}$.*

(C2) *$\|x_k\| < \infty$ for all $k \in \mathbf{N}$ w.p.1.*

(C3) *x^* is an asymptotically stable solution of the differential equation*

$$\frac{d\xi(t)}{dt} = -\frac{\partial J}{\partial x}(\xi(t)). \tag{9}$$

(C4) *Let $\mathbf{D}(x^*)$ be the domain of attraction for the point x^*, i.e., $\mathbf{D}(x^*) := \{\xi_0 \in \mathbf{R}^n \mid \lim_{t\to\infty} \xi(t, \xi_0) = x^*\}$ where $\xi(t, \xi_0)$ denotes the solution to the differential equation (9) for $\xi(0) = \xi_0$. Then, there exists a compact set $\mathbf{S} \subseteq \mathbf{D}(x^*)$ such that $x_k \in \mathbf{S}$ infinitely often for almost all sample points.*

(C5) *$a_k, c_k > 0$ for all $k \in \mathbf{N}$, $\lim_{k\to\infty} a_k = 0$, $\sum_{k=0}^{\infty} a_k = \infty$, $\lim_{k\to\infty} c_k = 0$, and $\sum_{k=0}^{\infty} a_k^2/c_k^2 < \infty$.*

(C6) *r_{ki} $(i = 1, 2, \ldots, n)$ are the i.i.d. random numbers drawn from the Bernoulli distribution*

$$\begin{cases} \mathbb{P}(r_{ki} = 1) = 0.5, \\ \mathbb{P}(r_{ki} = -1) = 0.5, \end{cases} \tag{10}$$

where r_{ki} is the i-th element of r_k.

Proof: *This is a consequence of the following three facts.*

*(i) Consider the standard SPSA algorithm in (2). Assume that a_k $(k = 0, 1, \ldots)$ are random variables. Then the solution sequence converges to x^* w.p.1 subject to **(C1)**–**(C4)**, **(C6)**, and*

(C5') *$a_k, c_k > 0$ for all $k \in \mathbf{N}$, $\lim_{k\to\infty} a_k = 0$, $\lim_{k\to\infty} c_k = 0$, and $\sum_{k=0}^{\infty} a_k^2/c_k^2 < \infty$. Moreover, $\sum_{k=0}^{\infty} a_k = \infty$ w.p.1.*

(ii) The algorithm in (6) is rewritten as

$$x_{k+1} = x_k - \tilde{a}_k g(x_k) \tag{11}$$

for

$$\tilde{a}_k := \begin{cases} \dfrac{2c_k d}{|J(x_k + c_k r_k) - J(x_k - c_k r_k)|} \\ \qquad \text{if } |J(x_k + c_k r_k) - J(x_k - c_k r_k)| > \dfrac{2c_k d}{a_k}, \\ a_k \quad \text{otherwise.} \end{cases} \tag{12}$$

Equation (11) is in the standard SPSA form in (2).

*(iii) Conditions **(C1)**, **(C2)**, and **(C5)** imply*

(C5") *$\tilde{a}_k, c_k > 0$ for all $k \in \mathbf{N}$, $\lim_{k\to\infty} \tilde{a}_k = 0$, $\lim_{k\to\infty} c_k = 0$, and $\sum_{k=0}^{\infty} \tilde{a}_k^2/c_k^2 < \infty$. Moreover, $\sum_{k=0}^{\infty} \tilde{a}_k = \infty$ w.p.1.*

Fact (i) is a straightforward result from [1] and [22] (pp. 17, line 1–9) (see Lemma 2 in Appendix A for further details), while (ii) and (iii) are nontrivial facts which have to be proven. So we show (ii) and (iii) in the following part.

*(ii) Let $\Delta J_k := J(x_k + c_k r_k) - J(x_k - c_k r_k)$. Then it follows from (6), (7), and $r_k^{(-1)} \in \{-1, 1\}^n$ (from **(C6)**) that*

$$\begin{aligned}\delta(a_k g(x_k)) &= \delta\left(a_k \frac{\Delta J_k}{2c_k} r_k^{(-1)}\right)\\ &= \min\left(a_k \frac{|\Delta J_k|}{2c_k}, d\right) \operatorname{sgn}\left(\Delta J_k r_k^{(-1)}\right)\\ &= \begin{cases} d \operatorname{sgn}\left(\Delta J_k r_k^{(-1)}\right) & \text{if } |\Delta J_k| > \dfrac{2c_k d}{a_k},\\ a_k g(x_k) & \text{otherwise.}\end{cases}\end{aligned} \tag{13}$$

Note here that

$$\begin{aligned}d \operatorname{sgn}\left(\Delta J_k r_k^{(-1)}\right) &= \frac{2c_k d}{|\Delta J_k|} \frac{|\Delta J_k|}{2c_k} \operatorname{sgn}\left(\Delta J_k r_k^{(-1)}\right)\\ &= \frac{2c_k d}{|\Delta J_k|} \frac{\Delta J_k}{2c_k} \operatorname{sgn}\left(r_k^{(-1)}\right)\\ &= \frac{2c_k d}{|\Delta J_k|} g(x_k)\end{aligned} \tag{14}$$

if $|\Delta J_k| > \frac{2c_k d}{a_k}$. From (13) and (14),

$$\delta(a_k g(x_k)) = \tilde{a}_k g(x_k). \tag{15}$$

So we have (ii).

*(iii) From (12) and **(C5)**, we have*

$$0 < \tilde{a}_k \leq a_k \tag{16}$$

*for every $k \in \mathbf{N}$. This and **(C5)** imply*

$$\lim_{k\to\infty} \tilde{a}_k = 0, \qquad \sum_{k=0}^{\infty} \frac{\tilde{a}_k^2}{c_k^2} < \infty. \tag{17}$$

*So the first condition of **(C5")** holds.*

*Next, we prove the second condition of **(C5")**, i.e., $\sum_{k=0}^{\infty} \tilde{a}_k = \infty$ w.p.1.*

*Conditions **(C1)** and **(C2)** imply that there exists a $\gamma_0 > 0$ satisfying*

$$\left\| \frac{\partial J}{\partial x}(x_k) \right\|_\infty \leq \gamma_0 \quad \textit{w.p.1} \tag{18}$$

for every $k \in \mathbf{N}$.

*On the other hand, by using the Taylor expansion, the triangle inequality, **(C1)**, **(C6)**, and (18), we have*

$$\begin{aligned}&|J(x_k + c_k r_k) - J(x_k - c_k r_k)|\\ &= \left| 2c_k \left(\frac{\partial J}{\partial x}(x_k)\right)^\top r_k \right.\\ &\qquad \left. + \frac{c_k^3}{6}\left(J^{(3)}(x_k^+) + J^{(3)}(x_k^-)\right) r_k \otimes r_k \otimes r_k \right|\\ &\leq 2c_k \left| \left(\frac{\partial J}{\partial x}(x_k)\right)^\top r_k \right|\\ &\quad + \frac{c_k^3}{6}\left| \left(J^{(3)}(x_k^+) + J^{(3)}(x_k^-)\right) r_k \otimes r_k \otimes r_k \right|\\ &\leq 2c_k n \gamma_0 + \frac{c_k^3 n^3 \alpha_0^3}{3} \quad \textit{w.p.1}\end{aligned} \tag{19}$$

where x_k^+ and x_k^- are some vectors on the line segments between x_k and $x_k + c_k r_k$ and between x_k and $x_k - c_k r_k$, respectively. It follows that

$$\frac{2c_k d}{|J(x_k + c_k r_k) - J(x_k - c_k r_k)|} \geq d\left(n\gamma_0 + \frac{c_k^2 n^3 \alpha_0^3}{6}\right)^{-1} \quad \textit{w.p.1.} \tag{20}$$

Finally, from (12), (16), (20), $d > 0$, $n > 0$, $\gamma_0 > 0$, and $\alpha_0 > 0$, we have

$$\begin{aligned}\sum_{k=0}^{\infty} \tilde{a}_k &\geq \sum_{k=0}^{\infty} \frac{2c_k d}{|J(x_k + c_k r_k) - J(x_k - c_k r_k)|}\\ &\geq \sum_{k=0}^{\infty} d\left(n\gamma_0 + \frac{c_k^2 n^3 \alpha_0^3}{6}\right)^{-1} \quad \textit{w.p.1}\\ &= \infty \quad \textit{w.p.1,}\end{aligned} \tag{21}$$

which completes the proof. □

Example 3 *Consider again $J(x)$ in (5). Now, let us apply the algorithm in (6) with $d := 0.5$ and the same condition as in Example 2.*

Fig. 3 illustrates the time evolution of $J(x_k)/J(x_0)$. Unlike the standard SPSA algorithm, the SPSA algorithm achieves the minimization of $J(x)$. □

In this way, the SPSA algorithm in (6) is more stable and so can be applicable to a wide range of real problems.

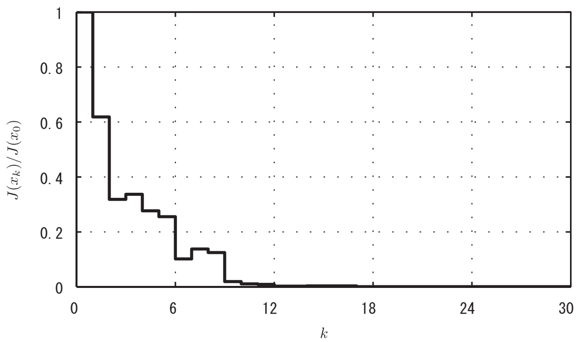

Fig. 3. Time evolution of $J(x_k)/J(x_0)$ for the modified SPSA algorithm.

4 Application to System Identification with Partially Known Model Structure

In this section, we apply the SPSA algorithm in (6) to a system identification problem.

Consider the system described by

$$y(t) = b_1 y(t-1) + b_2 y(t-2) + b_3 u(t-1) y(t-1) + b_4 f(b_5 u(t-1)) \tag{22}$$

where $t \in \mathbf{N}$ is the discrete time, $u(t) \in \mathbf{R}$ is the input, $y(t) \in \mathbf{R}$ is the output, $b_i \in \mathbf{R}\,(i = 1, 2, \ldots, 5)$ are constants. Moreover, $f : \mathbf{R} \to \mathbf{R}$ is a smooth function but it is assumed to be unknown. For the system, we address here the problem of identifying the parameters $b_i \in \mathbf{R}\,(i = 1, 2, \ldots, 5)$ with the input data $\hat{u}(0), \hat{u}(1), \ldots, \hat{u}(N)$ and the output data $\hat{y}(0), \hat{y}(1), \ldots, \hat{y}(N)$, assuming that a blackbox simulator of f is available, that is, the output value of f can be obtained once an input value is given.

This problem is reduced into the optimization problem in (1), whose objective function is given by

$$J(x) := \sum_{t=0}^{N} \left(\hat{y}(t) - y(t, \hat{u}(t), x)\right)^2 \tag{23}$$

where $x := [b_1 \;\; b_2 \;\; \cdots \;\; b_5]^\top \in \mathbf{R}^5$ and $y(t, \hat{u}, x)$ is the output at time t of the system (22) for the parameters x and the input sequence $\hat{u}(0), \hat{u}(1), \ldots, \hat{u}(N)$. Note here that the explicit function form of the gradient of $J(x)$ cannot be obtained due to the unknown function f, and so an SPSA method is a suitable solution.

Now, we apply the SPSA algorithm in (6) to this problem. Suppose that the original parameter is given by $b := [1 \;\; -0.5 \;\; 0.5 \;\; 1 \;\; 2]^\top$, $N = 1000$, and $\hat{u}(0), \hat{u}(1), \ldots, \hat{u}(1000)$, $\hat{y}(0), \hat{y}(1), \ldots, \hat{y}(1000)$ are given as Figs. 4 and 5. We set $x_0 := 0$, $a_k := 0.1/(k + 1000)^{0.602}$, $c_k := 0.01/(k + 1)^{0.101}$, and $d := 0.01$ for the algorithm in (6). Then we execute the algorithm for $t = 0, 1, \ldots, 999$ and obtain $x_{1000} = [0.99 \;\; -0.48 \;\; 0.51 \;\; 1.05 \;\; 1.96]^\top$, which is nearly equal to the original value of b. Fig. 6 shows the time evolution of $J(x_k)/J(x_0)$ and Fig. 7 illustrates the comparison of the original system and the identified model in terms of the output for another input sequence. These demonstrate that the SPSA algorithm is stable and solves the above system identification problem.

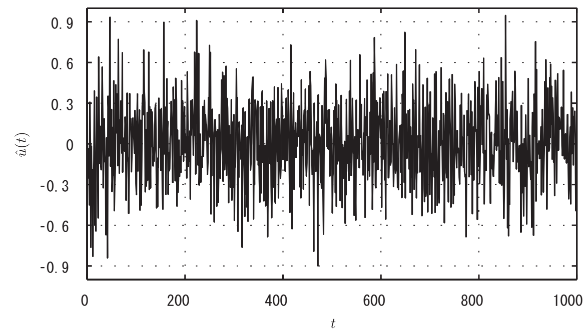

Fig. 4. Input data $\hat{u}(0), \hat{u}(1), \ldots, \hat{u}(1000)$ used for identification.

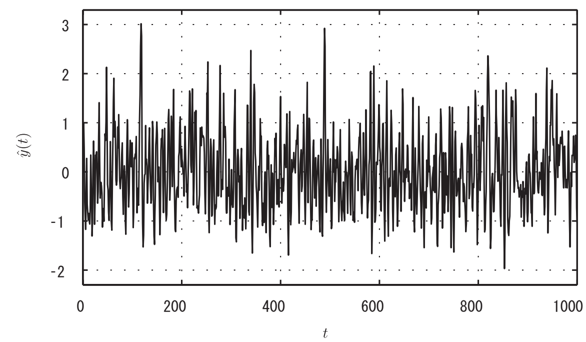

Fig. 5. Output data $\hat{y}(0), \hat{y}(1), \ldots, \hat{y}(1000)$ used for identification.

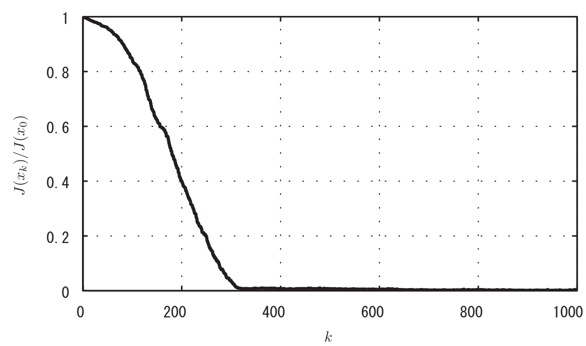

Fig. 6. Time evolution of $J(x_k)/J(x_0)$ for the modified SPSA algorithm.

On the other hand, Fig. 8 depicts the time evolution of

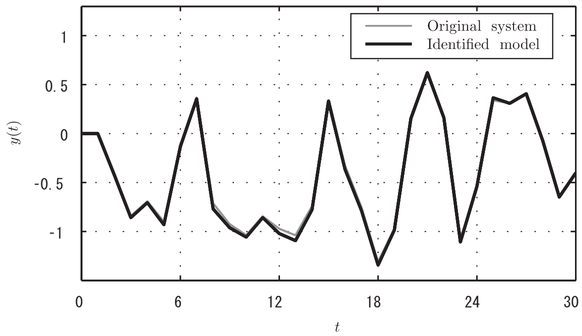

Fig. 7. Comparison between original system and identified model.

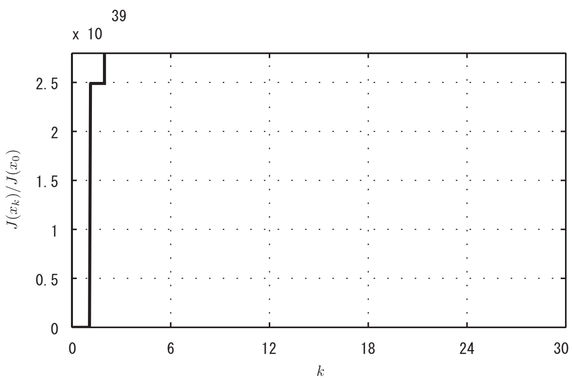

Fig. 8. Time evolution of $J(x_k)/J(x_0)$ for the standard SPSA algorithm.

$J(x_k)/J(x_0)$ for the standard SPSA algorithm in (2). This shows that the algorithm is unstable and does not give any solution to the problem. So it is concluded that the SPSA algorithm in (6) is a useful tool for system identification.

5 Conclusion

This paper has addressed the simultaneous perturbation stochastic approximation with norm-limited update vector to avoid some unstable solutions unlike the standard SPSA algorithm. As a theoretical result for the SPSA algorithm, the almost sure convergence has been proven. Finally, the proposed algorithm has been applied to a system identification problem with a partially known model structure.

Acknowledgements

This work was partly supported by Grant-in-Aid for Scientific Research (A) 25249058 and Grant-in-Aid for Young Scientists (A) 24686051 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

Appendix A: Convergence of Standard SPSA Algorithm

As a straightforward consequence of [1] and [22] (pp. 17, line 1–9), the following convergence result is provided for the standard SPSA algorithm given by (2) and (3).

Lemma 2 *For the standard SPSA algorithm given by (2) and (3), suppose that $x_0 \in \mathbf{R}^n$ is given and a_k ($k = 0, 1, \ldots$) are random variables. Let $x^* \in \mathbf{R}^n$ be a local solution to the problem in (1). If **(C1)**–**(C4)**, **(C6)** (given in Theorem 1), and **(C5')** hold for the algorithm, x_0, and x^*, then*

$$\lim_{k\to\infty} x_k = x^* \quad \text{w.p.1.}$$

Proof: Since it is proven in the same way as [1], we show here the sketch. If **(C1)** and **(C6)** hold, the expectation of $g(x_k)$ is nearly equal to $\frac{\partial J}{\partial x}(x_k)$, i.e.,

$$\mathbb{E}(g(x_k)|x_k) = \frac{\partial J}{\partial x}(x_k) + O(c_k^2) \quad (c_k \to 0).$$

By using this relation, the standard SPSA algorithm is reduced into the so-called Robbins-Monro algorithm [23]. Then **(C2)**–**(C4)** and **(C5')** imply the convergence conditions (see [23]) of the Robbins-Monro algorithm. This completes the proof. $\square$

References

[1] J.C. Spall: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation, *IEEE Transactions on Automatic Control*, 37(3), pp. 332–341, 1992.

[2] J.C. Spall: A one-measurement form of simultaneous perturbation stochastic approximation, *Automatica*, 33(1), pp. 109–112, 1997.

[3] J.C. Spall: Adaptive stochastic approximation by the simultaneous perturbation method, *IEEE Transactions on Automatic Control*, 45(10), pp. 1839–1853, 2000.

[4] J.L. Maryak and D.C. Chin: Global random optimization by simultaneous perturbation stochastic approximation, *IEEE Transactions on Automatic Control*, 53(3), pp. 780–783, 2008.

[5] S. Azuma, M. Selman Sakar, and G.J. Pappas: Stochastic source seeking by mobile robots, *IEEE Transactions on Automatic Control*, 57(9), pp. 2308–2321, 2012.

[6] S. Bhatnagar, H.L. Prasad, and L.A. Prashanth: *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods*, *Lecture Notes in Control and Information Sciences 434*, 2013.

[7] I. Baba, S. Azuma, and T. Sugie: Controller design for optimizing fuel consumption of hybrid electric vehicles: A model-free approach based on simultaneous perturbation stochastic approximation, *Transactions on Society of Instrument and Control Engineers*, 49(9), pp. 887–894, 2013.

[8] F. Rezayat: On the use of an SPSA-based model-free controller in quality improvement, *Automatica*, 31(6), pp. 913–915, 1995.

[9] J.C. Spall and J.A. Cristion: Model-free control of nonlinear stochastic systems with discrete-time measurements, *IEEE Transactions on Automatic Control*, 43(9), pp. 1198–1210, 1998.

[10] B. Hahn and K.R. Oldham: A model free on-off iterative adaptive controller based on stochastic approximation, *IEEE Transactions on Control Systems Technology*, 20(1), pp. 194–204, 2012.

[11] J.C. Spall and J.A. Cristion: Nonlinear adaptive control using neural networks: estimation with a smoothed form of simultaneous perturbation gradient approximation, *Proceedings of the 1994 American Control Conference*, pp. 2560–2564, 1994.

[12] A. Vakhitov, V. Vlasov, and O. Granichin: Adaptive control of siso plant with time-varying coefficients based on random test perturbation, *Proceedings of the 2010 American Control Conference*, pp. 4004–4009, 2010.

[13] A.A. Cole-Rhodes and K.L. Johnson: Multiresolution registration of remote sensing imagery by optimization of mutual information using a stochastic gradient, *IEEE Transactions on Image Processing*, 12(12), pp. 1495–1511, 2003.

[14] Q. Li, I. Sato, and Y. Murakami: Simultaneous perturbation stochastic approximation algorithm for automated image registration optimization, *IEEE International Conference on Geoscience and Remote Sensing Symposium*, pp. 184–187, 2006.

[15] S. Martin, G. Morison, W. Nailon, and T. Durrani: Fast and accurate image registration using Tsallis entropy and simultaneous perturbation stochastic approximation, *Electronics Letters*, 40(10), pp. 595–597, 2004.

[16] Y. Maeda, H. Hirano, and Y. Kanata: A learning rule of neural networks via simultaneous perturbation and its hardware implementation, *Neural Networks*, 8(2), pp. 251–259, 1995.

[17] J.I.M. Martinez, K. Nakano, and K. Higuchi: Parameter estimation in neural networks by improved version of simultaneous perturbation stochastic approximation algorithm, *Proceedings of the 2009 ICROS-SICE International Joint Conference*, pp. 4567–4572, 2009.

[18] Q. Song, J.C. Spall, and J. Ni: Robust neural network tracking controller using simultaneous perturbation stochastic approximation, *IEEE Transactions on Neural Networks*, 19(5), pp. 817–835, 2008.

[19] C. Antal, O. Granichin, and S. Levi: Adaptive autonomous soaring of multiple UAVs using simultaneous perturbation stochastic approximation, *49th IEEE Conference on Decision and Control*, pp. 3656–3661, 2010.

[20] S. Azuma, R. Yoshimura, and T. Sugie: Broadcast control of multi-agent systems, *Automatica*, 49(8), pp. 2307–2316, 2013.

[21] M.A. Ahmad, S. Azuma, I. Baba, and T. Sugie: Switching controller design for hybrid electric vehicles, *SICE Journal of Control, Measurement, and System Integration*, 7(5), pp. 273–282, 2014.

[22] V.S. Borkar: *Stochastic approximation: A dynamical systems viewpoint*, Cambridge University Press, 2008.

[23] H.J. Kushner and D.S. Clark: *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer, 1978.