Simultaneous Perturbation Stochastic Approximation in Global Optimisation*

Krzysztof Patan[1]

[1] Institute of Control and Computation Engineering, University of Zielona Góra
ul. Podgórna 50, 64-246 Zielona Góra, Poland, email: K.Patan@issi.uz.zgora.pl

Abstract. The paper deals with global property analysis of the Simultaneous Perturbation Stochastic Approximation algorithm. It spite of the fact that the algorithm uses an estimation of a gradient, a global optimisation property can be achieved by using properly shaped sequence $\{c_k\}$ used for generating trial points. The experiments are performed on a number of test functions showing efficiency of the method. The examined algorithm is also used to train a dynamic neural network. A network under consideration is designed using neuron models with internal dynamics. The identification of an industrial process based on real data shows usefullness of the algorithm.

1 Introduction

Many problems arising in engineering practice have a nonlinear, dynamic nature. To model such processes artificial neural networks can be applied. However, to satisfy requirements, a neural network should be trained first using available data. Therefore, optimisation techniques are of a crucial importance in the framework of neural network theory [2].

Unfortunately, the training process of a neural network of the dynamic type, seems to be an optimisation problem which is intrinsically related to a very rich topology of cost (objective) function [3]. Classical methods, e.g. back-propagation based algorithms, usually find one of the local unsatisfactory optima. A multi-start technique very seldom ends successfully [5]. Thus, algorithms of global optimisation should be implemented.

Recently, a large amount of attention was paid to the stochastic algorithms, among them stochastic approximation ones. This paper is devoted to the so-called Stochastic Perturbation Stochastic Apprioximation (SPSA) method. This relatively new optimisation technique posseses a number of advantages compared to other well known stochastic methods [8]. Moreover, it has a propery of global optimisation [4]. The main objective of this paper is to examine the global optimisation property of the SPSA and the application of this to dynamic neural network training.

The paper is organized as follows. First, in Section 2, the SPSA method is presented in detail. The dynamic neural network is described in Section 3. Section 4 contains

*This work was supported by the State Committee for Scientific Research in Poland (KBN) under the grant No. 4T11A01425

a series of experiments showing the global optimisation property of the SPSA and its application to dynamic neural network training. Remarks and conclusions are given in the final section.

2 Simultaneous Perturbation Stochastic Approximation

In recent years there has been observed a growing interest in stochastic optimisation algorithms that do not depend on gradient information or measurements. This class of algorithms is based on an approximation of the gradient of the loss function. The general form of Stochastic Approximation (SA) recursive procedure is as follows [9]:

$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) \tag{1}$$

where $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ is the estimate of the gradient $\partial J/\partial \hat{\boldsymbol{\theta}}$ based on the measurements $L(\cdot)$ of the loss function $J(\cdot)$ (where $L(\cdot)$ is a measurement of the loss function $J(\cdot)$ affected by noise). In the context of neural network training, the loss function can be in the form of the sum of squared errors between the desired and network outputs, calculated using the entire set of input patterns (batch or off-line learning). The essential part of this equation is the gradient approximation. The SPSA has all elements of $\hat{\boldsymbol{\theta}}$ randomly perturbed to obtain two measurements $L(\cdot)$, but each component $\hat{\boldsymbol{g}}_{ki}(\hat{\boldsymbol{\theta}}_k)$ is formed from a ratio involving the individual components in the perturbation vector and the difference in the two corresponding measurements. For two-sided simultaneous perturbation, estimation of gradient is obtained according to the formula [10]:

$$\hat{\boldsymbol{g}}_{ki}(\hat{\boldsymbol{\theta}}_k) = \frac{L(\hat{\boldsymbol{\theta}}_k + c_k\Delta_k) - L(\hat{\boldsymbol{\theta}}_k - c_k\Delta_k)}{2c_k\Delta_{ki}} \tag{2}$$

where the distribution of the user-specified p-dimensional random perturbation vector, $\Delta_k = (\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kp})^T$ is independent and symmetrically distributed about 0 with finite inverse moments $E(|\Delta_{ki}|^{-1})$ for all k, i. One of the possible distributions that satisfies these conditions is the symmetric Bernoulli ± 1. Two commonly used distributions that not satisfy these conditions are the uniform and the normal. The rich bibliography presents sufficient conditions for convergence of the SPSA ($\hat{\boldsymbol{\theta}}_k \to \boldsymbol{\theta}^\star$ in the stochastic almost sure sense). However, the efficiency of the SPSA depends on the shape of the $J(\boldsymbol{\theta})$, the values of gain sequences $\{a_k\}$ and $\{c_k\}$ and distribution of the $\{\Delta_{ki}\}$. The choice of the gain sequences is critical to the performance of the algorithm. In the SPSA the gain sequences are calculated as follows [9]:

$$a_k = \frac{a}{(A+k)^\alpha}, \quad c_k = \frac{c}{k^\gamma} \tag{3}$$

where a, c, A, α and γ are non-negative coefficients. In spite of the fact that the SPSA uses the gradient estimate, it it possibility to apply the method to global optimisation. Two solutions are reported in the literature [1, 4, 8]: (a) using an injected noise, (b) using a stepwise (slowly decaying) sequence $\{c_k\}$. In this work the second solution is used.

3 Dynamic neural network

The neural network under consideration exhibits dynamic characteristics. Dynamics is introduced into a neuron in such a way that the neuron activation depends on its internal states. This is done by introducing a linear dynamic system – the Infinite Impulse Response (IIR) filter – into the neuron structure [7, 3]. In this way, each neuron in the dynamic network reproduces the past signal values with the input $u_p(k)$, for $p = 1, 2, \ldots, P$, and the output $y(k)$. Three main operations are performed in this dynamic structure. First of all, the weighted sum of inputs is calculated according to the formula:

$$x(k) = \mathbf{w}^T \mathbf{u}(k) = \sum_{p=1}^{P} w_p u_p(k) \tag{4}$$

where $\mathbf{w} = [w_1 \; w_2 \ldots w_P]^T$ denotes the input–weight vector, P is the number of inputs, and $\mathbf{u}(k) = [u_1(k) \; u_2(k) \ldots u_P(k)]^T$ is the input vector (T is the transpose operator). The weights play a role similar to that in static feedforward networks. The weights together with the activation function are responsible for the approximation properties of the model. Then the calculated sum $x(k)$ is passed to the IIR filter. In this paper, the filters under consideration are linear dynamic systems of different orders, viz. the first or second order. The behaviour of this linear system can be described by the following difference equation:

$$\tilde{y}(k) = \sum_{i=0}^{n} b_i x(k-i) - \sum_{i=1}^{n} a_i \tilde{y}(k-i) \tag{5}$$

where $x(k)$ is the filter input, $\tilde{y}(k)$ is the filter output, $\mathbf{a} = [a_1 \; a_2 \ldots a_n]^T$ and $\mathbf{b} = [b_0 \; b_1 \ldots b_n]^T$ are feedback and feedforward paths weighted by the vector weights, n is the filter order and k is the discrete-time index. Finally, the neuron output can be expressed as:

$$y(k) = F\big(g \cdot \tilde{y}(k) + c\big) \tag{6}$$

where $F(\cdot)$ is the nonlinear activation function that produces the neuron output $y(k)$, c is the bias factor, and g is the slope parameter of the activation function. In the dynamic neuron, the slope parameter can change. Thus, the dynamic neuron can model better the biological neuron. A neural network consisting of dynamic neurons can have the same structure as a standard feedforward back-propagation one. All unknown network parameters can be represented by a vector $\boldsymbol{\theta}$. The objective of training is to adjust the elements of the vector $\boldsymbol{\theta}$ in such a way as to minimize some loss (cost) function:

$$\boldsymbol{\theta}^\star = \underset{\theta \in C}{\arg\min} \; J(\boldsymbol{\theta}) \tag{7}$$

where $\boldsymbol{\theta}^\star$ is the optimal network parameter vector, $J : \mathbb{R}^p \to \mathbb{R}$ represents some loss function to be minimized, p is the dimension of the vector $\boldsymbol{\theta}$, and $C \subseteq \mathbb{R}^p$ is the set of admissible parameters constituted by constraints. To minimize (7) the stochastic approximation method presented in Section 2 is applied.

4 Experiments

4.1 Optimisation of sum of two gaussian picks

Let us consider a function composed of two gaussian picks defined as follows:

$$f_1(\boldsymbol{x}) = -\exp\left(-(x_1+1)^2 - x_2^2\right) - \frac{1}{2}\exp\left(-\left(x_1 - \frac{3}{2}\right)^2 - x_2^2\right) \tag{8}$$

where $\boldsymbol{x} = [x_1, x_2]$. Note, that the global minimum is $f_1(\boldsymbol{x}^*) = -1$ at $\boldsymbol{x}^* = [-1, 0]$. Let the initial point $\boldsymbol{x}_0 = [2.5, 1.5]$. The parameters of the SPSA have been selected using the "trial and error" method, and have the following values: $a = 4$, $c = 5$, $\alpha = 0.5$, $\gamma = 0.16$ and $A = 10$. These parameters enable the property of global optimisation of the SPSA. The results are given in Fig. 1. In Fig. 1(a) one can see two dimensional plot of the function f_1 and a sequence of solutions marked by black points. The algorithm slightly approaches the local minimum located at $\boldsymbol{x} = [-1.5, 0]$ and then proceeds toward the global minimum. The Fig. 1(b) presents the track of solutions generated by the SPSA. This result clearly shows that, using suitable parameters, the algorithm can pass saddles beetween local/global minima and find a final solution. A conclusion is that a proper selection of the sequence $\{c_k\}$ is of a crucial importance. For example, let $c = 1$, $a = 0.9$, $\alpha = 0.2$, $\gamma = 0.1$, $A = 10$. In this case, the algorithm possesses strict local optimisation property and finds a local minimum at $\boldsymbol{x} = [-1.5, 0]$ after 33 iterations with the accuracy $\varepsilon = 0.001$. In turn, a large value of c results in the algorithm being unable to calculate a gradient in a proper way and the method is slowly convergent. In spite of the fact that a quite large number of iterations is needed to find a global minimum, the algorithm is not complex. On PC machine with Intel Pentium Centrino 1.7GHz processor and 512 MB RAM the simulation including 5000 iterations lasts ≈ 1.15 seconds.

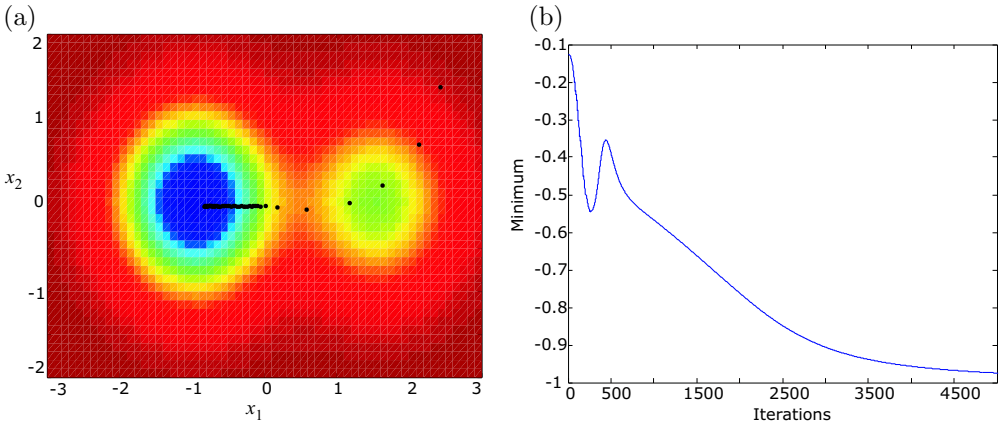

Figure 1. Optimisation of the function f_1. Two-dimensional plot of the function with the sequence of solutions (a), track of solutions (b).

4.2 Optimisation of "drop wave" function

The function considered next has a form:

$$f_2(\boldsymbol{x}) = 2 - \frac{1 + \cos\left(12\sqrt{x_1^2 + x_2^2}\right)}{\frac{1}{2}\left(x_1^2 + x_2^2\right) + 2} \tag{9}$$

where $\boldsymbol{x} = [x_1, x_2]$. The global minimum is $f_2(\boldsymbol{x}^*) \sim 0.5$ at $\boldsymbol{x}^* = [0, 0]$. Let the initial point $\boldsymbol{x}_0 = [3.6, 3.2]$. The property of global optimisation is examined with the following parameters of the SPSA: $a = 9$, $c = 15$, $\alpha = 0.25$, $\gamma = 0.2$ and $A = 1000$. The results are given in Fig. 2. Such established sequences a_k and c_k allow the algorithm to easily cross saddles and locate the global minimum. The searching process is clearly presented in both Figs. 2 (a) and (b). In this case, the simulation including 10000 iterations lasts ≈ 2.5 seconds.

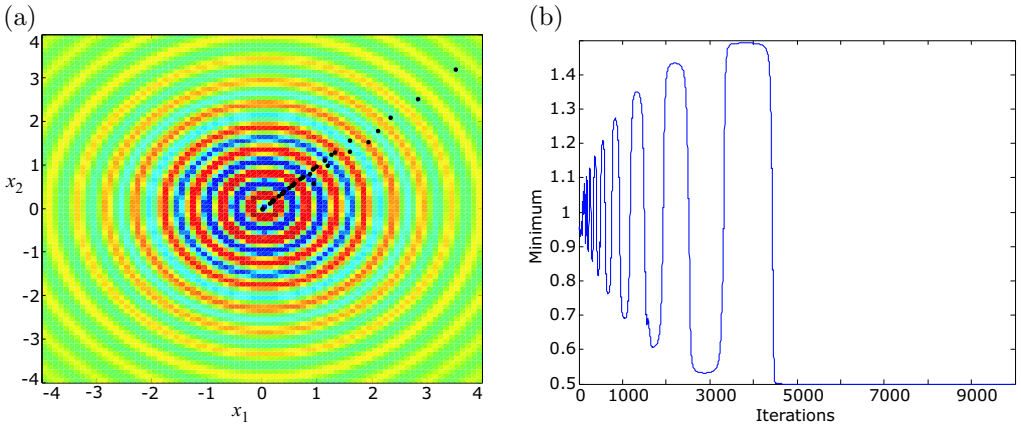

Figure 2. Optimisation of the function f_2. Two-dimensional plot of the function with the sequence of solutions (a), track of solutions (b).

4.3 Optimisation of Rastriagin's function

The last considered function is defined by the formula:

$$f_3(\boldsymbol{x}) = 100 - \left(100 - \left(x_1^2 + x_2^2\right) - 10\left(\cos(2\pi x_1) + \cos(2\pi x_2)\right)\right) \tag{10}$$

where $\boldsymbol{x} = [x_1, x_2]$. In this case, the function has four global minima $f_3(\boldsymbol{x}^*) \sim 19.5$ at $\boldsymbol{x}_1^* = [-0.5, -0.5]$, $\boldsymbol{x}_2^* = [-0.5, 0.5]$, $\boldsymbol{x}_3^* = [0.5, -0.5]$ and $\boldsymbol{x}_4^* = [0.5, 0.5]$. Let the initial point be $\boldsymbol{x}_0 = [4, 4]$. The best founded set of algorithm parameters is: $a = 0.5$, $c = 3$, $\alpha = 0.6$, $\gamma = 0.3$ and $A = 100$. The results of optimisation using this set of parameters is shown in Fig. 3. The global minimum has been located pretty fast (~ 500 iterations). The algorithm quickly passes the searching domain into the region including minima with lower values of the cost function and then tries to find global minimum jumping between two neighbouring minima. This process is continued till the sequencess $\{a_k\}$

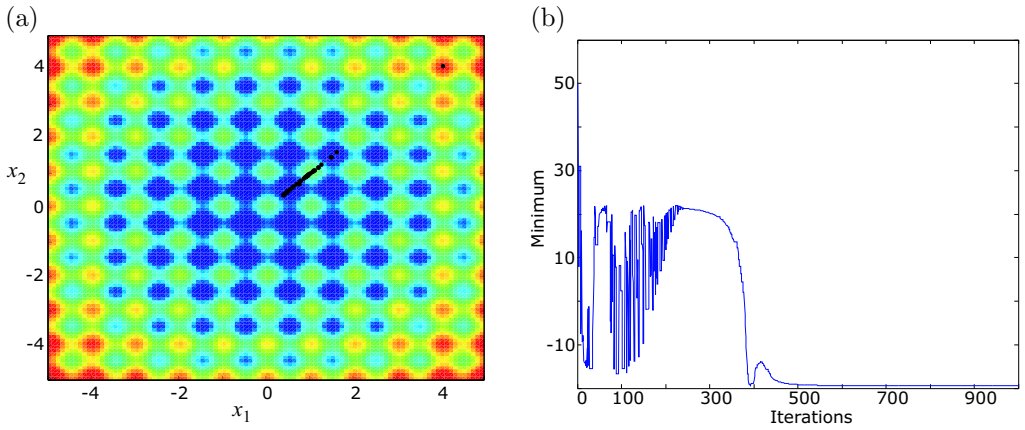

Figure 3. Optimisation of the function f_3. Two-dimensional plot of the function with the sequence of solutions (a), track of solutions (b).

and $\{c_k\}$ acquire proper values. An interesting behaviour is observed for $c = 1$. In this case the algorithm searches the domain in a chaotic manner. Further analysis shows that smaller values of c enable local optimisation properties, while larger c, e.g. $c = 10$ causes the algorithm to search too large un area with negative effect. The searching process consisting of 1000 iterations lasts ≈ 0.36 second.

4.4 Neural network training

In this section, the SPSA is applied to train the dynamic neural network described in details in Section 3. The process under consideration is the actuator of the sugar evaporation station in the Lublin Sugar Factory (Poland) [6]. The actuator to be modelled is the valve located at the inlet of the evaporation station. For this valve, two process variables are available: the control value for the valve on the juice inlet to the evaporation section (the actuator input), and the juice flow on the inlet to the evaporation station (the actuation). With these two signals the neural model of the actuator can be defined as:

$$y = F_N(u) \tag{11}$$

where F_N denotes the nonlinear function. The dynamic neural network of the structure $N^2_{1,5,1}$ (two processing layers, one input, five neurons in hidden layer and one output), is trained using the SPSA method. Taking into account dynamic behaviour of the valve each neuron in the network structure possesses a first order filter. The model of the valve is identified using real process data recorded during the sugar campaign in October 2000. In the sugar factory control system, the sampling time is equal to 10 s. Thus, during one work shift (6 hours) approximately 2 160 training samples per one monitored process variable are collected. To perform experiments two data sets were used. The first set, containing 500 samples, was used for training and another one, containing 1 000 samples, was used to check the generalisation ability of the network.

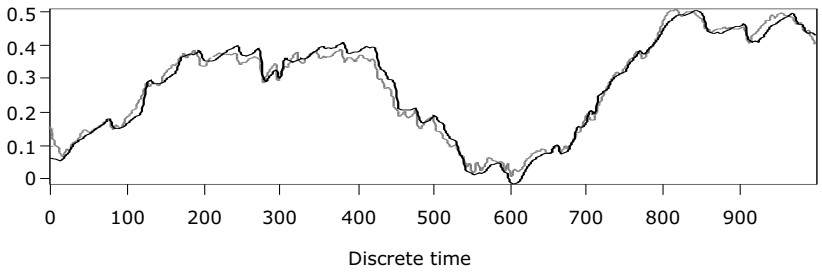

Figure 4. Modelling of the valve, testing phase: output of the valve (black) and output of the neural network (gray).

The experiment was carried out for 7 500 iterations using the following parameters $a = 0.001$, $A = 100$, $c = 0.01$, $\alpha = 0.25$ and $\gamma = 0.05$. The modelling results for the testing set are presented in Fig. 4. The parameter γ controls the decreasing ratio of sequence $\{c_k\}$ and is set to a small value to enable a property of global optimisation. Parameter a is set to a very small value to assure convergence of the algorithm. The dynamic neural network is very sensitive to large changes in parameters values (dynamic filters) and large values of a like 0.4 can result in the learning process becoming divergent. The results achieved suggest that the model is pretty good.

5 Conclusions

The main objective of this work was examining the property of global optimisation of Simultaneous Perturbation Stochastic Approximation algorithm. The one way to enable the property of global optimisation of this method is to use a slowly decaying rate of sequence $\{c_k\}$. This sequence is responsible for generating trial points used for estimation of a gradient. A number of experiments has been performed to show the ability of the SPSA to find a global optimum. Moreover, a discussion on the influence of the value of parameter c on behaviour of the method has been also included. Next, to show efficiency of the algorithm it has been applied to dynamic neural network training. The algorithm itself is not complex but, in the case of off-line training calculation of the cost function, can be time consuming, especially in the case of large data sets used for training. Taking into account that standard SPSA uses two measurements to calculate a gradient estimate, the SPSA can be slower than gradient based methods. A solution is to apply one-sided version of the SPSA. Summarizing, the SPSA is a fast optimisation method, but to start the optimisation procedure, five parameters should be determined. To define these values and use the method properly, a user should possesses a lot of knowledge about this method. Taking into account the property of global optimisation, this stochastic approach can be effectively used for optimisation and modelling of nonlinear processes.

Bibliography

[1] D. C. Chin. A more efficient global optimization algorithm based on Styblinski and Tang. *Neural Networks*, 7:573–574, 1994.

[2] S. Haykin. *Neural Networks. A comprehensive foundation, 2nd Edition.* Prentice-Hall, New Jersey, 1999.

[3] J. Korbicz, K. Patan, and A. Obuchowicz. Dynamic neural networks for process modelling in fault detection and isolation systems. *International Journal of Applied Mathematics and Computer Science*, 9(3):519–546, 1999.

[4] J.L. Maryak and D. C. Chin. Global random optimization by Simultaneous Perturbation Stochastic Approximation. In *Proc. of the American Control Conference, ACC 2001, Arlington VA, USA*, pages 756–762, 2001.

[5] A. Obuchowicz. *Evolutionary Algorithms for Global Optimization and Dynamic System Diagnosis.* University of Zielona Góra Press, Zielona Góra, Poland, 2003.

[6] K. Patan and J. Korbicz. Application of Dynamic Neural Networks in an Industrial Plant. In *Proc. Int. Symp. Fault Detection Supervision and Safety for Technical Processes, SAFEPROCESS 2000, Budapest, Hungary*, pages 186–191, 2000.

[7] K. Patan and T. Parisini. Identification of neural dynamic models for fault detection and isolation: the case of a real sugar evaporation process. *Journal of Process Control*, 15:67–79, 2005.

[8] J. C. Spall. *Introduction to Stochastic Search and Optimization.* John Willey & Sons, New Jersey, 2003.

[9] J.C. Spall. Multivariate stochastic aproximation using a simultaneous perturbation gradient approximation. *IEEE Trans. Automatic Control*, (37):332–341, 1992.

[10] J.C. Spall. Stochastic optimization, stochastic approximation and simulated annealing. In J.G. Webster, editor, *Encyclopedia of Electrical and Electronics Engineering*, New York, 1999. John Wiley & Sons.