International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

Pseudo-Gradient Algorithms of Adaptation of Random Search with the Accumulation Information for Determination of the Direction of Working Step

George Filatov

Professor, Doctor of Techn. Sciences, Ukrainian State University of Chemical Technology, Ukraine

Abstract **− Random search - a powerful method of nonlinear programming that used to solve complex applications. But in order to successfully apply this method in practice, the tool required. This tool is the algorithm. In order that this algorithm will could work, you need to configure it properly. For that there is the algorithms of adaptation. This paper is a kind of overview of the algorithms of adaptation of random search methods. The ideas of adaptation adduced in this paper belong to different authors, mainly from the school of L.A. Rastrigin. The engineer, who is familiar with these ideas, may create your own algorithm of random search, which will be able to solve technical problem.**

Keywords **—Random search method, Adaptation, Algorithm, Accumulation of Information, Direction of Working Step, Learning, Self-Learning.**

I. Introduction

In a previous article [1] was studied in detail the question of estimating the direction of the descent by method of statistical gradient. For a linear objective function by numerically method on a computer it was analyzed the dynamics of the estimation of the vector of gradient based on the number of trials and the dimension of the search space, was adduced approximating expression for the density distribution of the angle between the statistical gradient and the gradient of objective function.

In this paper, we present a few of pseudo-gradient algorithms of random search, which realize the idea of the accumulation of information for determination of the direction of working step.

II. Algorithms Of Adaptation

A. Algorithm of random search with accumulation

The algorithm of random search with the accumulation is as follows [2]. From the initial state $\mathbf{X}=(x_1,x_2,....,x_n)$ are fulfilled the m random steps $g\Xi^{(1)}, g\Xi^{(2)},....., g\Xi^{(m)}$ and calculated the corresponding increment of the objective function:

$$\Delta F^{(j)} = F\left(\mathbf{X}+g\Xi^{(j)}\right)-F(\mathbf{X}), \qquad (1)$$

where $\Xi=(\xi_1,\xi_2,....,\xi_n)$ – n – demensional uniformly distributed unit random vector; g – the value of test step. After that formed the vector sum of:

$$\mathbf{Y}=\sum_{j=1}^{m}\Xi^{(j)}\Delta F^{(j)}, \qquad (2)$$

Which, as can be shown for a linear objective function $F(\mathbf{X})$, which in the limiting state at $m\to\infty$ coincides with the direction of function gradient $F(\mathbf{X})$. On the other hand, the average value of the vector $\mathbf{Y}$ throughout realization of the test steps also coincides with the direction of the gradient of function $F(\mathbf{X})$.

Thus, the direction of the vector $\mathbf{Y}$ at the final value of m is a statistical estimate of the gradient direction. Consequently, the direction of working step is naturally determined in accordance with this assessment:

$$\Delta\mathbf{X}=-a\mathbf{Y}, \qquad (3)$$

where a – the value of the working step. When $n=m$ and when the test steps are orthogonal and are non-random and directed along the coordinates:

$$\Xi^{(1)}=(1,0,...,0),.....,\Xi^{(2)}=(0,0,...,1), \quad (4)$$

The considered algorithm degenerates in gradient method. This shows that the gradient method is a special case of the method of statistical gradient.

An essential feature and the distinction of the method of statistical gradient is the ability of deciding on the direction of the working steps at any value m, in particular, when $m<n$, while the method of gradient search to decide the direction of the working step, requires not less than of n trial steps.

Let's investigate the adaptation of accumulation volume m in the described above statistical gradient algorithm. At this the aim of adapting will be the stabilization of the probability of the choice of erroneous step at a given level [3].

Consider the case of small trial steps of search, i.e., when $g\approx 0$, and in the absence of constraints imposed on the objective function $F(\mathbf{X})$.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

In this case we can confine ourselves to the linear objective function:

$$F(\mathbf{X}) = F_0 + \langle \mathbf{X},\ \text{grad}\ \mathbf{F} \rangle, \qquad (5)$$

The expression in square brackets – scalar product. In the case of minimization of the linear function (5), the probability of erroneous step is stipulated by the formula

$$P = \text{Prob}\left[0 \le \varphi \le \frac{\pi}{2}\right] = \int_o^{\frac{\pi}{2}} p(\varphi) d\varphi, \qquad (6)$$

where φ – the angle of inclination of the working step $\Delta\mathbf{X}$ to the direction of the gradient stipulated by the formula (3); $p(\varphi)$ – the density of distribution of the angle defined by the volume of accumulation m.

Let the working step is normalized in accordance with [4]:

$$\Delta\mathbf{X} = -\frac{an}{2gm}\sum_{j=1}^{m}\Xi^{(j)}\left[F\left(\mathbf{X} + g\Xi^{(j)}\right) - F\left(\mathbf{X} - g\Xi^{(j)}\right)\right]. \qquad (7)$$

For the mathematical expectation and dispersion of the projection $\Delta\mathbf{X}$ on the gradient vector for the case of a linear objective function in work [3], are adduced the formulas:

$$M(\Delta\mathbf{X}_g) = -ak\ , \qquad (8)$$

$$D(\Delta\mathbf{X}_g) = \frac{a^2}{m}\left[2k^2\frac{n-1}{n+2} + \frac{n\sigma^2}{2g^2}\right], \qquad (9)$$

where $k = \|\text{grad}\,Q(\mathbf{X})\|$ – the modulus of the gradient; $\Delta\mathbf{X}_g$ – the projection of vector $\Delta\mathbf{X}$ on the direction of the gradient. Assuming that the distribution density of projection $\Delta\mathbf{X}_g$ is normal, and using equations (8)-(9) using the formula (6) we can determine the probability of an erroneous step:

$$P = \frac{1}{2}\left[1 - \Phi\left(\frac{m}{2\sqrt{\frac{n-1}{n+2}}}\right)\right], \qquad (10)$$

Where $\Phi(z)$ – the Laplace function. If we preliminarily know a specific value of error probability $P = P_0$, then using the formula (10) we can obtain an equation for optimum accumulation, which ensures the following probability:

$$m^* = \left(4\frac{n-1}{n+2}\right)\left[\Phi^{-1}(1-2P_0)\right]^2, \qquad (11)$$

where Φ^{-1} – the inverse Laplace function. This formula makes it possible to determine the volume of accumulation to provide the specified probability of erroneous step search, depending on the modulus of the gradient k.

B. Algorithm of adaptation of the directions of step

Let's describe an algorithm to adapt the search step directions using the recurrence formula [5]:

$$p_N(\Delta\mathbf{X}) = p(\Delta\mathbf{X}_N / \mathbf{U}_N); \qquad (12)$$

$$\mathbf{U}_{N+1} = \psi(\mathbf{U}_N, \Delta\mathbf{X}_N, \Delta F_N); \qquad (13)$$

$$|\mathbf{U}_N| \le d\ . \qquad (14)$$

Here: $\mathbf{U}_N = \left(u_1^{(N)}, u_2^{(N)}, ..., u_n^{(N)}\right)$ – vector memory; N – search step number; $p(\Delta\mathbf{X}_N / \mathbf{U}_N)$ – conditional density of distribution of the vector $\Delta\mathbf{X}$ on the $\Delta\mathbf{X}$ – th step of search, which depends on the magnitude of the vector $\mathbf{U}_N$ on the same step of the search; ψ – vector function, which determines the change of the vector memory from N-th till N+1 – th search step, depending on the experience gained at N – th search step. To limit unwanted determinization of search process, on the modulus of vector increment $\mathbf{U}$ is superimposed the restriction (14), where d – a positive number. The process of adaptation of the density distribution $p(\Delta\mathbf{X})$ is carried out by changing the vector memory $\mathbf{U}$.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

The geometric contents of vector memory $\mathbf{U}$ is such that it indicates the direction of the greatest decrease in minimized objective function $F(\mathbf{X})$ in space $\mathbf{X}$. Formula (13) in the process of search brings nearer vector of memory $\mathbf{U}$ to this direction and formula (12) makes possible reorganize the distribution density $p(\Delta\mathbf{X})$ in such a way that with the greatest probability was chosen direction of search $\Delta\mathbf{X}$ which would closer to direction of vector memory $\mathbf{U}$.

C. Algorithms for continuous adaptation [2]

Conventionally can distinguish between several types of adaptation, depending on out of which multiplicities the random vectors $\Delta\mathbf{X}$ and $\mathbf{U}$ take the values. If the vector of step $\Delta\mathbf{X}$ and the vector of memory $\mathbf{U}$ assume the values out of continuous sets of vectors, the algorithms of adaptation are called continuous. The density distribution of probability of step $\Delta\mathbf{X}$ for such algorithms are described by formula (2), at this at $\mathbf{U} = (0,0,....,0)$ the distribution must be uniform. In these algorithms, the direction of random step $\Delta\mathbf{X}$ conveniently represent in the form of a vector function [6]:

$$\Delta\mathbf{X} = aQ(\Xi, \mathbf{U}), \qquad (15)$$

Where a – the step length; $\Xi = (\xi_1, \xi_2, ..., \xi_n)$ – random vector $(|\Xi| = 1)$, uniformly distributed in all directions of the space $\mathbf{X}$; $Q(\Xi, \mathbf{U})$ – a continuous vector function of two variables Ξ and $\mathbf{U}$ $(|Q(\Xi, \mathbf{U})| = 1)$. For different algorithms, this feature may not be identical, but it is convenient that at the same time it satisfies the following property:

1. $Q(\Xi, 0) = \Xi$, (16)

Where $0 = (0,0,....,0)$. This means that at zero value of vector memory the search should be equally probable;

2. $M[Q(\Xi, \mathbf{U})] = \dfrac{\mathbf{U}}{|\mathbf{U}|}$ (17)

The mathematical expectation of direction of random step $\Delta\mathbf{X}$ must coincide with the direction of the vector memory $\mathbf{U}$;

3. $D[Q(\Xi, \mathbf{U})] = \dfrac{d_1}{|\mathbf{U}| + d_2}$, (18)

The dispersion of a random step $\Delta\mathbf{X}$ must be inversely proportional to the absolute value of the vector $\mathbf{U}$. The coefficients in equation (18) take the following values:

$$d_1 = \text{const}, \quad d_2 = \text{const}, \quad d_1 > 0, \quad d_2 > 0 \backslash$$

When these conditions are the function $Q(\Xi, \mathbf{U})$ implements the spatial distribution of the random step, which may purposefully vary over time as a result of accumulation of experience by system of search.

As examples for the function can cite such expressions:

$$Q_1 = \frac{R\Xi + \mathbf{U}}{|R\Xi + \mathbf{U}|}, \qquad (19)$$

$$Q_2 = \frac{B(\Xi + 0{,}5\mathbf{U})}{|B(\Xi + 0{,}5\mathbf{U})|}, \qquad (20)$$

where the matrix

$$B = \begin{Vmatrix} 1 & 0 & 0 & . & . & . & 0 & 0 \\ 0 & \dfrac{b}{b_1} & 0 & . & . & . & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & . & 0 & \dfrac{b}{b_1} \end{Vmatrix} \qquad (21)$$

$$b = \left[\frac{1}{n}\left(1 - \frac{|\mathbf{U}|^2}{4}\right)\right]^{\frac{1}{2}}; \qquad b_1 = \left[1 - \frac{n-1}{n}\left(1 - \frac{|\mathbf{U}|^2}{4}\right)\right]^{\frac{1}{2}} \qquad (22)$$

$$Q_3 = \Xi \quad if \left|\frac{U}{U + d_2} - \Xi\right| \le f|U|, \qquad (23)$$

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

where $f(|\mathbf{U}|)$ – a non-decreasing function

$$0 \le f(|\mathbf{U}|) \le 2, \qquad (24)$$

at this $f(0) = 2$.

We note that the function describes getting into dimensional hyper-cone with an angle $4\arcsin\frac{f(|\mathbf{U}|)}{2}$ at the top.

Adaptation algorithm (13) which is carrying out the restructuring of the vector memory for functions (18), (20) and (23) can be represented as a vector of the recurrence relation that links two consecutive values of the vector

$$\mathbf{U}_{N+1} = k\mathbf{U}_N - \delta(\Delta F + d)\Delta\mathbf{X}_N, \qquad (25)$$

where k – the coefficient of forgetting. At $k = 0$ is remembered only the last step;

δ – parameter that determines the speed of adaptation $(\delta > 0)$;

ΔF_N – increment the optimized function on N –th step;

d – the coefficient of "skepticism";

$\Delta\mathbf{X}_N$ – the vector of N –th step of search in space $\mathbf{X}$. In order to avoid unwanted determinization of search system, the modulus of vector $\mathbf{U}$ is limited in such a way: $|\mathbf{U}| < d \quad (d > 0)$.

It is easy to see that when working with the algorithm (25) vector memory $\mathbf{U}$ aspires to reconstruct itself in the direction opposite to the gradient of optimized function. This implies that in accordance with the condition (17) the steps of search $\Delta\mathbf{X}$ will directed towards the average in side of the rapid reducing of function $F(\mathbf{X})$.

D. Algorithms of discrete adaptation

1. The exponential dependence:

$$p(u_i) = \begin{cases} \frac{1}{2}e^{-\alpha u_i} & \text{if} \quad u_i < 0 \\ 1 - \frac{1}{2}e^{-\alpha u_i} & \text{if} \quad u_i \ge 0 \end{cases}. \qquad (29)$$

If the random vector step $\Delta\mathbf{X}$ takes the value out of a finite set of vectors, the adaptation algorithms we conventionally will call "discrete" [2,5].

The vector memory $\mathbf{U}$ in this case may take the values as out of finite and out of infinite sets of vectors. In this case, the probability density $p_N(\Delta\mathbf{X})$ is replaced by a set of probability densities $p_1^{(N)}, p_2^{(N)}, ..., p_m^{(N)}$, where:

$$p_i^{(N)} = \text{Prob}\left(\Delta\mathbf{X} = \frac{\Delta\mathbf{X}^{(i)}}{\mathbf{U}} = \mathbf{U}_N\right) \qquad (26)$$

$$p_1^{(N)} \ge 0; \quad \sum_{i=1}^{m} p_i^{(N)} = 1 \quad (i = 1,2,....,m) \qquad (27)$$

One of the approaches to the construction of a discrete adaptation algorithms is the use of the method of coordinate-wise adaptation [2]. At the coordinatewise adaptation is rebuilt the probability of choosing a positive sign of increments Δx_i; for each $i-$th coordinate $(i = 1,2,....,n)$, depending on the sign of the increment Δx_i and on the sign of $F(\mathbf{X})$. Let the probability p_i of choosing a positive step along $i-$ой the variable x_i is the certain function of parameter of memory at this coordinate u_i:

$$p_i^{(N)} = p\left(u_i^{(N)}\right). \qquad (28)$$

Here N – step number of search. The function type $p(u_i)$ may be different, but it should satisfy the condition $0 \le p(u_i) \le 1$ and to be non-decreasing, i.e., with an increase of parameter u_i the value of probability of step in the positive direction on $i-$th coordinate should also increase. As an example of function $p(u_i)$, we will adduce a few dependencies [5, 7].

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

2. Linear dependence:

$$p(u_i)=\begin{cases}0 & \text{if} \quad u_i<-1\\ \frac{1}{2}(1+u_i), & \text{if} \quad -1\le u_i<1\\ 1 & \text{if} \quad u_i\ge 1\end{cases}. \qquad (30)$$

3. Gaussian dependence:

$$p(u_i)=\frac{1}{2}\left[1+\Phi(u_i)\right], \qquad (31)$$

where $\Phi(z)$ – the probability integral.

4. The sinusoidal dependence:

$$p(u_i)=\begin{cases}0 & \text{if} \quad u_i<-1\\ \frac{1}{2}+\frac{1}{\pi}\arcsin u_i & \text{if} \quad -1\le u_i<1\\ 1 & \text{if} \quad u_i\ge 1\end{cases}. \qquad (32)$$

5. The stair dependence

$$p(u_i)=\begin{cases}0 & \text{if} \quad u_i<-1\\ \frac{1}{2} & \text{if} \quad -1\le u_i<1\\ 1 & \text{if} \quad u_i\ge 1\end{cases}. \qquad (33)$$

6. Relay's dependence:

$$p(u_i)=\begin{cases}0 & \text{if} \quad u_i\le 0\\ 1 & \text{if} \quad u_i>0\end{cases}. \qquad (34)$$

The adaptation algorithm for dependencies (29)-(34) is implemented by a corresponding change of the parameters of memory, for example by means of a recurrence formula [5]:

$$u_i^{(N+1)}=u_i^{(N)}-\delta\cdot\operatorname{sign}\left(\Delta x_i^{(N)}\cdot\Delta F_N\right), \qquad (35)$$

where δ – the value that determines the speed of adaptation $(\delta>0)$; ΔF_N – the increment of minimized function on $N-$th search step; Δx_i – increment of coordinate x_i on $N-$th search step; $u_i^{(N)}$ – the value of memory parameter on $N-$th search step. The more δ, the faster the search engine is adapted. At $\delta=0$ the adaptation is absent. The content of formula (35) is as follows: if the step taken has led to an increase in minimized function, i.e., the step was taken in the wrong direction, then the probability of choosing this direction along this coordinate on the next step is decreasing.

Conversely, in the case of decreasing of the minimized function the probability of choosing this direction along such coordinate increases. Thus, this formula provides reconstruction of the characteristics probability of selecting the increment sign along the coordinate, which is needed to adapt the search system in the process of optimization

However, when using this algorithm turns out that the search system overly determined by in a certain direction. For the restructuring on the other direction it is necessary to return the memory in a equiprobable state, i.e., to provide the possibility to make step in the desired direction, which is associated with a lengthy process of return parameter $u_i(i=1,2,...,n)$ to the zero state. This makes the algorithm not mobile.

$$d_i'\le u_i\le d_i,(i=1,2,....,n) \qquad (36)$$

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

In view of these limitations the adapting algorithm acquires the necessary mobility at the rebuilding. The meaning of boundaries d_i' and d_i $(i=1,2,....,n)$ is set depending on the character of the objective function.

In the above algorithm of adaptation the parameter of memory u_i along any of the coordinates $x_i (i=1,2,...,n)$ and at any result ΔF on one step of search is changed to a constant value, which is equal to the value step δ on memory u_i. However, the need to adaptation on certain coordinate is dependent primarily on the result ΔF and the degree of participation of the increment Δx_i in this result. So sometimes it is advisable to use other adaptation algorithms. Here are a few examples of such adaptation algorithms [5].

E. The proportional algorithm of adaptation

This algorithm can be represented as a recurrence formula:

$$u_i^{(N+1)} = u_i^{(N)} - \delta \cdot \Delta x_i^{(N)} \cdot \Delta F_N \quad (37)$$

where δ – the proportionality coefficient ($\delta > 0$).

The proportional algorithm [2,5] already reacts on the result of search step and reacts on the degree of the participation of a particular parameter in this result. The introduction of such proportionality does this algorithm is sensitive enough to determine the best direction of the search. Indeed, the direction along which the memory function changes slightly, affect on the parameters of the memory are also slightly. If the system is accidentally stumble on a more effective direction, the memory parameters will rebuild themselves in this direction faster.

The considered algorithm of adaptation remembers and stores in averaged form the entire previous experience of search. Obviously, this is not necessary. Moreover, changes of conditions of the work of the operating system requires a fairly rapid forgetting of previously acquired experience, as it was received in a different environment and is "obsolete."

That is why, in some conditions it is advisable to work with the adaptation algorithm with the forgetting.

F. The algorithm of adaptation with the forgetting [2,5]

We describe this algorithm in more detail. In conditions of equiprobable search out of point $\mathbf{X}_N$ belonging to the set R, in the space of optimized parameters is performed a step in a random direction. If the new state $\mathbf{X}_{N+1}$ value of the objective function is smaller than the previous, the next step is performed out of a found random state $\mathbf{X}_{N+1}$. Otherwise, the next step the search system performs out of the initial state.

The search system is described above reminds the algorithm with the return after a failed step, but it is not so. The system of search would perform the step out of state $\mathbf{X}_{N+1}$ not in the random direction but in direction selected at the previous step.

The coordinates of vector **X** are changed in such way:

$$x_i^{(N+1)} = x_j + \Delta x_i^{(N+1)}, \quad (38)$$

$$\Delta x_i^{(N+1)} = a\Xi_i, \quad (39)$$

$$x_j = x_i^{(N)}, \text{ if } \quad F(\mathbf{X}_{N+1}) < F_j^*, \quad (40)$$

where a – the working length of the step in the space of parameters to be optimized; Ξ – next realization of a random vector; x_j – optimal parameter values on the j previous steps; F_j^* – the minimum value of the objective function on the j previous steps.

The self-learning of system of search consists in the reformation of probabilistic search characteristics in such a way, that the search has become non-random. This is accomplished through a targeted influence on the unit random vector characterizing the direction of the search.

The component of the vector Ξ is determined as follows:

$$\xi_i = \alpha_i + (\beta_i - \alpha_i) x_{\text{psr}}, \quad (41)$$

where ξ_i – a random number uniformly distributed on the segment $[\alpha_i, \beta_i]$; x_{psr} – pseudo-uniformly distributed on the segment $[0,1]$.

Assume that the probability of choosing a positive step p_i along the $i-$ th variable (coordinate) is a piecewise linear function of some parameter, which we call memory parameter $i-$ th for the coordinates on $N-$ th step of the search:

$$p_i^{(N)} = 0{,}5\left(u_i^{(N)} + 1\right) \quad (42)$$

In general, a function $p_i = f(u_i)$ can be any, but always monotonous and non-decreasing. Then the border for random value ξ_i will be calculated as follows:

$$\alpha_i = \begin{cases} -1, & \text{if} \quad p_i < 0{,}5 \\ -1 + (p_i - 0{,}5)\cdot 2, & \text{if} \quad p_i \geq 0{,}5 \end{cases} \quad (43)$$

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

$$\beta_i = \begin{cases} +1, & \text{if} \quad p_i \geq 0{,}5 \\ +1+(p_i - 0{,}5)\cdot 2, & \text{if} \quad p_i < 0{,}5 \end{cases} \quad (44)$$

By changing the vector memory $\mathbf{U}$ using the recurrence formula

$$u_i^{(N+1)} = k \cdot u_i^{(N)} - \delta \cdot \Delta x_i^{(N)} \cdot \Delta F_N, \quad (45)$$

Appropriately we change the probability of selecting the next step (2.77), which in turn will change the values of α_i and β_i and hence to change ξ_i.

Here: $k > 0$ – the coefficient of forgetting; $\delta > 0$ – the value that determines the speed of learning; $\Delta x_i^{(N)}$ – an increase of the $i-$th parameter on $N-$th step.

The value ξ_i participates in the formation of increment of $i-$th coordinate (39) and therefore, of the coordinate itself (38). The change of coordinate value causes a change in the value of the objective function ΔF_N, which in turn leads to a change in the vector memory (45). That logical inverse connection of operations in the algorithm of coordinate-wise self-learning with forgetting allows to determine the most likely search direction. To avoid "redetermination" of search system, on the area of the change of memory parameters is appropriate to impose restrictions in the form of:

$$C_1 \leq \mathbf{U} \leq C_2, \quad (46)$$

i.e.

$$U = \begin{cases} C_1 & \text{at} \quad u_i \leq C_1 \\ u_i & \text{at} \quad C_1 \leq u_i \leq C_2 \\ C_2 & \text{at} \quad u_i > C_2 \end{cases}.$$

The introduction of such restrictions on the memory option allows you to avoid long-term adjustment of the search backwards, if the search system has selected not right direction.

In the district of extremum the system of search is rebuilt by changing the size of the step. This operation avoids oscillations of the quality function in the region of extremum and thus reduces the loss on search.

For this algorithm is essential the numerical value of probability of step in the positive direction at $u_i = 0$. It should be equal to 0.5, i.e., the search should be equiprobable.

Like the previous one, this algorithm is proportional. However, unlike previous algorithms, this learning algorithm has more flexible strategy. During the search of extremum he does not keep all the previously accumulated data, as the whole experience, accumulated by search system can adversely affect the search and lead to redetermination of system. The coefficient of forgetting of old experience in this algorithm manages by the process of forgetting. This coefficient is chosen within $(0 < k < 1)$. If $k = 1$, the forgetting is absent. If $k = 0$, the system remembers only the result of the last step.

We show that in the absence of experience ($\Delta F = 0$) the search system with forgetting degenerates into equiprobable search. Indeed, let in the initial state $u_i^{(0)} = 1$, $0 < k < 1$ and $\Delta F = 0$ ($N = 1,2,....$). Then, using the consecutively the formula (2.80), we get on $i-$th step:

$$u_i^{(N)} = k_i^{(N)} \cdot u^{(0)} = k^N.$$

Obviously, if $N \to \infty$, the $u_i^{(N)} \to 0$, that is equivalent to equiprobable search.

Adaptation algorithms, discussed above, are not able to constantly look for the best direction of change of the objective function; once the right direction is found, they will try to fix the movement search system in this direction without worrying about the possibility of re-deploy of search system in other direction. The reason of this is mostly the positive feedback, which is characteristic of a learning system and consisting in the fact that the probability of a favorable step always increases, regardless of whether was found the best direction, or not was found such direction.

This drawback can be overcome in two ways. On the one hand, can only use a negative experience, that is, changing the memory u only at increasing of objective function, when $\Delta F > 0$ (adaptation only at error). This measure leads to the fact that the system will not have the positive inverse connection, however, easy to see, it does not make it selective to different directions and does not force the search for the best direction of optimization.

But there is another way: to build adaptation algorithm in such a way, that the system always would not have of positive experience, and the result of any search step would be perceived as a negative, but in varying degrees. This is easily accomplished by introducing one additional parameter.

G. The algorithm of "skeptical" adaptation

The recurrence formula of this algorithm can be written as [2,5]:

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 3, March 2017)

$$u_i^{(N+1)} = u_i^{(N)} - \delta \cdot (\Delta F_N + d)\Delta x_i^{(N)}, \quad (47)$$

where d – some positive constant – "parameter of skepticism", which always provides $\Delta F_N + d > 0$.

It is easy to notice, such an algorithm "punishes" by decrease in probability the any variant of changing the parameters, but in varying degrees. Favorable variants "are punished" in less degree than unfavorable variants; due to this the adaptation takes place. This algorithm fixes no one direction, but on average provides an advantage only for the best direction in given circumstances . However, this property the algorithm has acquired at the expence of the deliberate refusal of enhance the probability of selecting favorable directions, i.e., at the cost of superfluous "suspiciousness", which, of course, increases the losses on search, and reduces the effectiveness of algorithm in simpler situations.

We note that the adaptation algorithms (35), (37)-(47) represent the linear algorithms adaptation of memory parameter u_i. In general view, they may be represented as follows:

$$u_i^{(N+1)} = k \cdot u_i^{(N)} - \varphi \cdot \left(\Delta x_i^{(N)}, \Delta F_N\right), \quad (48)$$

where the functions $\Delta x_i^{(N)}, \Delta F_N$ control the change of parameter of memory u_i, depending on the increment along the variable x_i, which was obtained on $N-$ th step and depending on the experience which was gained on this step. Another class of efficient algorithms we can be obtained, if in function $\varphi\left(\Delta x_i^{(N)}, \Delta F_N\right)$ the argument $\Delta x_i^{(N)}$ replace on the argument $u_i^{(N)}$, i.e., if the formula (48) is replaced by the formula [8]:

$$u_i^{(N+1)} = k \cdot u_i^{(N)} - \varphi \cdot \left(u_i^{(N)}, \Delta F_N\right). \quad (49)$$

The algorithms of adaptation that are described by the formula (49) are the automats with the deterministic transitions and random outputs [8], and their study is reduced to the research of the behavior of automat in a random environment [8, 9]. If the change of memory parameter describes by formula.

$$u_i^{(N+1)} = \begin{cases} u_i^{(N)} - \varphi\left(u_i^{(N)}, \Delta F_N\right) & \text{with the probability } q_i \\ u_i^{(N)} + \varphi\left(u_i^{(N)}, \Delta F_N\right) & \text{with the probability } 1-q_i \end{cases}, \quad (50)$$

Then we have a class of adaptation algorithms, which are the automats with the random transitions [10, 11, 12, 13].

REFERENCES

[1] Filatov G.V. The Adaptation of Random Search algorithms. International Journal of Emerging Technology & Advanced Engineering, Volume 6, Issue, August, 201, p.p..

[2] Rastrigin L.A., Ripa K.K., Tarasenko G.S. Adaptation of random search. – Riga, Zinatne, 1978. – 243 p..

[3] Rastrigin L.A. Statistical methods for evaluating the gradient // Automation and Computer Science. – 1970. – №4. – P.25-31.

[4] Rastrigin L.A. The system of exrteme management. M .: Nauka, 1974. – 630 p.

[5] Rastrigin L.A. Statistical methods of search.– M .: Science, 1968.– 376 p.

[6] Rastrigin L.A. Random search with linear tactics. – Riga Zinatne, 1971. – 310 p.

[7] Rastrigin LA A random search in problems of optimization of multivariable systems. – Riga Zinatne, 1965. – 287p.

[8] Rastrigin LA, Ripa K.K. Automaton theory of random search. Riga: Zinatne, 1973. – 343p.

[9] Tsetlin ML Research on the theory of automata and modeling of biological systems. – M .: Nauka, 1969.– 316 p.

[10] Grigorenko V.P., Rapoport A.P. To the theory of search by collective of independent automatic machins // Math. Executive. Proc. institutions. Radio Physics. – 1970. – №11. – P.1726-1735.

[11] Неймарк Ю.И. Neumark Y.I. Automatic optimization // Math. Executive. Proc. institutions. Radio Physics. – 1972. – №7. – P.967-971.

[12] Ripa KK Synthesis of optimal algorithms in the class of random search algorithms with automatic self-learning // Problems of random search. - Riga. –1974. – Vol.3. – P.42-66.

[13] Ripa K.K. On the equivalence of self-learning algorithms at random search // Problems of random search. – Riga. – 1975. – Vol 4, – P.66-84.