mathematics

MDPI

Article

Reinforcement Procedure for Randomized Machine Learning

Yuri S. Popkov [1,2,*], Yuri A. Dubnov [1,3] and Alexey Yu. Popkov [1]

[1] Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, 44/2 Vavilova, 119333 Moscow, Russia; yury.dubnov@phystech.edu (Y.A.D.); apopkov@isa.ru (A.Y.P.)
[2] Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, 65 Profsoyuznaya, 117997 Moscow, Russia
[3] Faculty of Computer Science, National Research University "Higher Schools of Economics", 20 Myasnitskaya, 109028 Moscow, Russia
* Correspondence: popkov@isa.ru

Abstract: This paper is devoted to problem-oriented reinforcement methods for the numerical implementation of Randomized Machine Learning. We have developed a scheme of the reinforcement procedure based on the agent approach and Bellman's optimality principle. This procedure ensures strictly monotonic properties of a sequence of local records in the iterative computational procedure of the learning process. The dependences of the dimensions of the neighborhood of the global minimum and the probability of its achievement on the parameters of the algorithm are determined. The convergence of the algorithm with the indicated probability to the neighborhood of the global minimum is proved.

Keywords: randomized machine learning; reinforcement learning; utility function; payoff function; Bellman's optimality principle

MSC: 68Q87; 68T05

1. Introduction

The beginning of this century has been marked by an increased interest in the problems of reinforcement learning. The essence of this branch of machine learning is to train an object (model, algorithm, etc.) by interacting not with a teacher but with an environment, using the trial-and-error method with reward or penalty depending on the results.

Let us look at this idea, abstracting from the specifics of the experiment, exclusively from the methodological point of view. Clearly, it represents a virtual game procedure where the game is simulated by two player-agents, their strategies, and quantitative assessments of their payoffs and losses. Reflecting on the peculiarities of learning processes, F. Rosenblatt, the author of the perceptron, introduced the concept of learning without a teacher and classified the types of structural tuning for playing automata [1].

The same concept can be traced in the paper [2] by I.M. Gelfand, I.I. Pyatetskij-Shapiro, and M.L. Tsetlin. The authors proposed a mathematical model of a game between two automata with a variable structure changing in the course of interaction with the environment. The interaction results were characterized by quantitative assessments.

Later, the response to the action of "environment" was given a particular term, the so-called "reinforcement." It became a whole branch in the theory and applications of machine learning. Admittedly, both focused on two problems, clustering (visualization) and pattern recognition. Such problems involve objects with their quantitative characteristics (feature), and, most importantly, the "distances" between them can be calculated. Some kinds of rewards or penalties in the algorithm parameters were arranged based on the distance matrix. Neural networks were used as algorithms [3]. In particular, the so-called "Kohonen maps" were one of the first research works in this area; for details, see [4]. In such maps, the weights of a neural network are adjusted using a game-theoretic model that implements

check for updates

Citation: Popkov, Y.S.; Dubnov, Y.A.; Popkov, A.Y. Reinforcement Procedure for Randomized Machine Learning. *Mathematics* **2023**, *11*, 3651. https://doi.org/10.3390/math11173651

Academic Editor: Ioannis G. Tsoulos

Received: 24 July 2023
Revised: 17 August 2023
Accepted: 22 August 2023
Published: 23 August 2023

the principle of competition between its nodes: an advantage is gained for the nodes with the minimum distance between the objects at each step of the algorithm.

Subsequently, reinforcement learning was actively developed based on the automata models of an object (agent) interacting with the environment in game-theoretic terms (strategies, utility functions, and payoffs). It was presented to the scientific community as a certain model of human management [5].

Numerous algorithms appeared with different models and volumes of a priori information about the environment, different methods for choosing strategies, and different procedures for forming utility functions. For example, we refer to [6–8]. A fairly comprehensive survey of reinforcement learning methods was prepared at the Department of Mathematical Methods of Forecasting (Faculty of Computational Mathematics and Cybernetics, Moscow State University) [9].

The general structure of reinforcement learning procedures is interpreted in terms of a Markov decision process, an extremely general construction of one-step iterations in continuous time t with feedback, accompanied by a specific terminology [10]. Its main components are an agent model with output (agent's action) and inputs in the form of environment states and rewards, current or averaged over a certain number of iterations, and an environment model with input (agent's action) and outputs in the form of specified rewards and responses (environment states). The fundamental feature of this procedure is the empirical estimation of the conditional probabilities of rewards for the agent's actions based on adjustable random Monte Carlo simulations. Such simulations (also called iterations or trials) are used to average a fixed number of current rewards or discount them. The resulting function depends on the environment state and the agent's strategy and is being taken as a utility function (an analog of the objective function in teacher-assisted learning procedures). During learning, this function is sequentially maximized [11,12] using Bellman's optimality principle [13] in its stochastic setting [14] (Many researchers of reinforcement learning interpret it as learning without a teacher. Indeed, this approach involves no goal-setting in the form of a teacher's error-and-response function to be minimized. However, the corresponding role is played by experimentally generated utility and reward functions, which represent a virtual "teacher." The structures of these functions and methods for calculating their mathematical expectations are based on experimental statistical material and expert opinions. Therefore, the results of using reinforcement learning often provoke discussions.).

Reinforcement learning is actively applied in its traditional field—robotics [15–17]—as well as in self-tuning procedures for trade forecasting [18], adaptive programming technologies [19,20], and dynamic decision support [21].

In the papers [22,23], and the book [24], a new machine learning procedure (Randomized Machine Learning, RML) was developed. The basic concept of RML is based on the use of a parameterized model with random parameters, its optimization using the conditional information entropy maximization method, and the subsequent generation of random parameters with optimized probability density functions. According to this concept, it consists of three stages: analytical (determining the entropy-optimal probability density function of randomized model parameters and measurement noises consistent with empirical balances with the data), computational (solving the empirical balance equations numerically), and experimental (performing Monte Carlo simulations to reproduce random sequences with the entropy-optimal probabilistic characteristics).

Because all machine learning problems incorporate intrinsic uncertainty in models and data, it was proposed to maximize the informational entropy of probability density functions (PDFs) of the model parameters and measurement noises as a measure of uncertainty subject to empirical balances with real data. This is a functional entropy-linear programming problem of the Lyapunov type [24]. It has an analytical solution, i.e., the optimal PDFs parameterized by Lagrange multipliers, which are determined from the empirical balance equations.

They are specific nonlinear equations containing the so-called integral components (multidimensional parametrized definite integrals). Therefore, it is impossible to establish any fruitful properties of the equations that would ensure the convergence of iterative procedures for their solution.

In this paper, we employ the GFS method based on Monte Carlo batch iterations [25,26]. The basic method GFS (Generation, Filtration, Selection) is an improved method for finding an approximate value of the global minimum on a unit cube, with an estimate of the size of the neighborhood and the probability of reaching it.

A problem-oriented version of the reinforcement concept is being developed to fundamentally improve the computational properties of the GFS method and the RML procedure as a whole. We prove the theorem on the strict monotonic decrease of the residual function for a system of nonlinear equations of empirical balances in which only measurements of the values of the functions are available. The latter is used to study the convergence with probability 1 of an iterative procedure with reinforcement and to estimate the size of the neighborhood of the global minimum and the probability of reaching it with a finite number of iterations.

Therefore, our contribution to the theory and practice of RMS is to develop a reinforcement scheme that allows us to increase the computational efficiency of the procedure and prove its convergence to the neighborhood of the global minimum with a certain probability.

2. The Mathematical Model of the RML Procedure

We study the problem of learning the model of dependence between one-dimensional input and output data. Consider a set of measurements of input data $x[0], \ldots, x[N]$ and output data $y[0], \ldots, y[N]$. The latter are measured with random and independent noises $\xi[0], \ldots, \xi[N]$ of the interval type:

$$\xi[k] \in \Xi_k = [\xi_k^{(-)}, \xi_k^{(+)}], \quad k \in \mathcal{N} = [0, N], \tag{1}$$

where $\xi_k^{(-)}, \xi_k^{(+)}$ are left and right boundaries of the interval. The probabilistic properties of the measurement noises are characterized by PDFs $Q_k(\xi[k])$, $k \in \mathcal{N}$. Suppose that they are continuously differentiable.

The mathematical model of the general dynamic dependence with finite memory ρ is described by a functional $\mathcal{B}$ [24]:

$$\hat{y}[k] = \mathcal{B}(x[i],\, k - \rho \le i \le k \,|\, \mathbf{a}), \quad k \in \mathcal{N}, \tag{2}$$

where parameters $\mathbf{a} = \{a_1, \ldots, a_m\}$.

If the functional $\mathcal{B}$ is linear and continuous, it can be represented by a segment of the Volterra functional power series [24].

In the equality above, the parameters $\mathbf{a}$ are random and interval-type:

$$\mathbf{a} \in \mathcal{A} \subset R^m, \quad \mathcal{A} = \left[\mathbf{a}^{(-)}, \mathbf{a}^{(+)}\right]. \tag{3}$$

The probabilistic properties of the parameters are characterized by a PDF $P(\mathbf{a})$, which is supposed to be continuously differentiable as well.

The output of the model is observed with additive noises:

$$\hat{v}[k] = \hat{y}[k] + \xi[k], k \in \mathcal{N}. \tag{4}$$

Because the model parameters are random and measurements of the output are distorted by random noises, according to (1) and (4) it is generated ensembles of random trajectories $\hat{y}[k] \in \mathcal{Y}$ and $\hat{v}[k] \in \mathcal{V}$, where $k \in \mathcal{N}$.

To form the morphological properties of the PDFs, we adopt the numerical characteristics of ensembles based on moments and called *normalized total moments:*

$$m^{(s)}[k] = (\mathcal{M}\{\hat{y}^s[k]\})^{1/s} + (\mathcal{M}\{\xi^s[k]\})^{1/s}, \quad s \in [1, S], \tag{5}$$

where s is a degree of the moment, and S is a number of moments.

$$\begin{aligned} \mathcal{M}\{\hat{y}^s[k]\} &= \int_{\mathcal{A}} P(\mathbf{a})\,(\mathcal{B}(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a}))^s d\mathbf{a}, \\ \mathcal{M}\{\xi^s[k]\} &= \int_{\Xi_k} Q_k(\xi[k])\,(\xi[k])^s d\xi[k], \quad k \in \mathcal{N}. \end{aligned} \tag{6}$$

The numerical characteristics (5) are the values of the normalized total moments along the trajectories of the observed model output. Output data $u^{(s)}[k]$, $s \in [1, S]$, $k \in \mathcal{N}$ are assumed to be similar indicators of some real process:

$$u^{(s)}[k] = (\mathcal{M}\{y^s[k]\})^{1/s}. \tag{7}$$

In particular, such properties are inherent in trading procedures for options [27,28]. In this case, the basic RML algorithm [24] has the following form:

$$\begin{aligned} &\mathcal{H}[P(\mathbf{a}); Q_0(\xi[0]), \ldots, Q_N(\xi[N])] = \\ &- \int_{\mathcal{A}} P(\mathbf{a})\, \ln P(\mathbf{a})\, d\mathbf{a} - \sum_{k=0}^{N} \int_{\Xi_k} Q_k(\xi[k])\, \ln Q_k(\xi[k])\, d\xi[k] \Rightarrow \max \end{aligned} \tag{8}$$

subject to

—the normalization conditions

$$\int_{\mathcal{A}} P(\mathbf{a}) d\mathbf{a} = 1, \;\; \int_{\Xi_k} Q_k(\xi[k]) d\xi[k], \;\; k \in \mathcal{N}, \tag{9}$$

and

—the empirical balance conditions

$$m^{(s)}[k] = u^{(s)}[k], \quad k \in \mathcal{N}. \tag{10}$$

Problem (8)–(10) has an analytical solution parameterized by Lagrange multipliers $\Lambda = [\lambda_{s,k} \,|\, s = \overline{1, S};\, k \in \mathcal{N}]$:

$$\begin{aligned} P^*(\mathbf{a}) &= \frac{\exp\left(-\sum_{s=1,k=0}^{S,N} \lambda_{s,k} \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a})\right)}{\mathcal{P}(\Lambda)}, \\ Q_k^*(\xi[k]) &= \frac{\exp(\lambda_{s,k} \xi^s[k])}{\mathcal{Q}_k(\lambda_{s,k})}, \end{aligned} \tag{11}$$

where

$$\begin{aligned} \mathcal{P}(\Lambda) &= \int_{\mathcal{A}} \exp\left(-\sum_{s=1,k=0}^{S,N} \lambda_{s,k} \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a})\right) d\mathbf{a}, \\ \mathcal{Q}_k(\lambda_{s,k}) &= \int_{\Xi_k} \exp(-\lambda_{s,k}\, \xi^s[k])\, d\xi[k]. \end{aligned} \tag{12}$$

The Lagrange multipliers figuring in these equations satisfy the empirical balance equations

$$\begin{aligned} & \mathcal{P}^{-1}(\Lambda) \int_A \exp\left(- \sum_{s=1,k=0}^{S,N} \lambda_{s,k} \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a}) \right) \times \\ \times \quad & \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a})\, d\mathbf{a} + \\ + \quad & \mathcal{Q}_k^{-1}(\lambda_{s,k}) \int_{\Xi_k} \exp(\lambda_{s,k}\, \xi^s[k])\, \xi^s[k]\, d\xi[k] = u^{(s)}[k], \\ & s \in [1,S];\, k \in \mathcal{N}. \end{aligned} \tag{13}$$

It can be seen from these equations that they contain the so-called integral components, namely, definite parametrized multidimensional integrals on m-dimensional parallelepipeds $\mathcal{A}$ (3). In general, it is possible to determine numerically only the values of the functions in which they are included. The latter excludes the possibility of a reasonable declaration of the properties of functions in the left parts of these equations.

3. The Adaptive Method of Monte Carlo Packet Iterations with Reinforcement (the GFS-RF Algorithm)

To solve these equations, in [26], the GFS algorithm was proposed, which is a modification of the random search method, in which the generation (G) of the number M_i of random and independent points specified at each iteration step i on the unit cube in R^m, filtering (F) "good" points, i.e., that fall into the admissible region, their selection (S) according to the values of the residual functional adopted for these equations. The convergence properties of GFS were based on the existence of certain functional properties of the functions involved in these equations. It is proposed to fundamentally modify this algorithm using the ideas of reinforcement.

The Canonical Form of the Problem

The system of formula (13) can be represented in the following form:

$$\Phi_{s,k}(\Lambda) = 0, \quad s \in [1,S],\ k \in [0,N], \tag{14}$$

where $\Lambda = [\lambda_{s,k} \,|\, s \in [1,S],\, k \in \mathcal{N}]$—Lagrange multipliers matrix, and functions

$$\begin{aligned} \Phi_{s,k}(\Lambda) = & \mathcal{P}^{-1}(\Lambda) \int_A \exp\left(- \sum_{s=1,k=0}^{S,N} \lambda_{s,k} \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a}) \right) \times \\ \times \quad & \mathcal{B}^s(x[\tau],\, k-\rho \le \tau \le k \,|\, \mathbf{a})\, d\mathbf{a} + \\ + \quad & \mathcal{Q}_k^{-1}(\lambda_{s,k}) \int_{\Xi_k} \exp(\lambda_{s,k}\, \xi^s[k])\, \xi^s[k]\, d\xi[k] - u^{(s)}[k], \\ & s \in [1,S];\, k \in \mathcal{N}. \end{aligned} \tag{15}$$

In the vectorization procedure [29], Equations (14) and (15) can be written as

$$\boldsymbol{\phi}(\boldsymbol{\lambda}) = \mathbf{0}, \tag{16}$$

where the vector function $\boldsymbol{\phi}$, the variable $\boldsymbol{\lambda}$, and the 0-vector on the right-hand side have the dimension $d = S \times (N+1)$. The vector $\boldsymbol{\lambda} \in R^d$, i.e., its components take values $-\infty < \lambda_n < \infty$.

We reduce problem (16) to the canonical form using the following change of variables:

$$\begin{aligned} z_n &= \frac{1}{1+\exp(-b_n\, \lambda_n)}, \\ \lambda_n &= \frac{1}{b_n} \ln \frac{z_n}{1-z_n}, \quad n \in [1,d], \end{aligned} \tag{17}$$

where b_n is a parameter. This mapping changes the infinity interval to the interval $[0,1]$.

As a result, Equation (16) takes the form

$$\boldsymbol{\psi}(\mathbf{z}) = \mathbf{0}, \quad \mathbf{z} \in Z_+^d = [\mathbf{0}, \mathbf{1}]. \tag{18}$$

We introduce the residual function (the Euclidean norm)

$$J(\mathbf{z}) = \| \boldsymbol{\Psi}(\mathbf{z}) \|_E . \tag{19}$$

Solving Equation (18) is equivalent to finding points $\mathbf{z}^*$, in which the global minimal of the residual function $J(\mathbf{z})$ is reached. Such an interpretation turns out to be fruitful since the global minimum is known:

$$J(\mathbf{z}) \geq J(\mathbf{z}^*) = 0. \tag{20}$$

Thus, solving Equation (18) is reduced to finding the global minimum of a continuous function that is bounded below and algorithmically computable function values on the unit cube:

$$J(\mathbf{z}) \Rightarrow \text{glob min}, \quad \mathbf{z} \in Z_+^d. \tag{21}$$

Because the function $J(\mathbf{z})$ is continuous and $\mathbf{z} \in Z_+^d$, there exist its modulus of continuity $\omega(h)$ and positive constants (H, h):

$$\omega(H, h) = \max_{(\mathbf{v},\mathbf{y}) \in Z_+^d; \|\mathbf{v}-\mathbf{y}\| \leq h} |J(\mathbf{v}) - J(\mathbf{y})| \leq H h^s, \tag{22}$$

where the constants H, s are unknown. In order to use these constants to study the properties of the iterative process, we have to estimate them using only the values of the residual function.

4. Structure of Reinforcement Procedure

Let us introduce a useful terminological framework. The function $J(\mathbf{z})$ is treated as an *environment* and its values $J_k(\mathbf{z}^{(k)})$ on iteration k are responses to the *agent*'s strategy (*action*) $\mathbf{z}^{(k)}$. The quality of the environment response is assessed by a *utility function* $Q(J)$, whose values on iteration k are $Q_k(J_k)$. The quality of the agent's actions (strategies) is characterized by a *payoff function* $\kappa(Q)$.

The self-learning algorithm minimizing the residual function (21) based on Monte Carlo packet iterations has the following reinforcement scheme. Note that this algorithm enumerates in a controlled way the values of the residual function on the unit cube. Therefore, the reinforcement scheme is focused on learning rational controllability to accelerate the iterative process.

Agent. The agent's strategy on iteration k is to generate a packet of uniformly distributed random vectors on the unit cube. The strategy is characterized by the grid step η_k and the number M_k of random values for each component of the vector $\mathbf{z}$ from the interval $[0, 1]$. They have the relation

$$\eta_k = M_k^{-q}, \quad 0 < q < 1, \tag{23}$$

where q is a fixed parameter.

Due to this relation, let *the agent's strategy be the value* M_k.

On a given grid step, it is possible to generate a different number N_k of independent random vectors (agent's strategies) with the uniform distribution on the cube Z_+^d :

$$\mathcal{Z}_k = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(N_k)}\}. \tag{24}$$

Assume that in this packet (As has been emphasized, we employ *simple Monte Carlo simulations*: the same number of independent random numbers with the uniform distribution on $[0,1]$ is generated for each coordinate of the original space),

$$N_k = M_k^d. \tag{25}$$

For each pair of the $(k-1)$th and kth packets, the corresponding $(k-1,k)$-records, and the decrements are calculated by the formulas

$$\begin{aligned} J_{k-1}^*(M_k) &= J(\mathbf{z}^{N_{k-1}}) = \min_{\mathbf{z}\in\mathcal{Z}_{k-1}} J(\mathbf{z}), \\ J_k^*(M_k) &= J(\mathbf{z}^{N_k}) = \min_{\mathbf{z}\in\mathcal{Z}_k} J(\mathbf{z}), \end{aligned} \tag{26}$$

and

$$u_k(M_k) = J_k^*(M_k) - J_{k-1}^*(M_k), \tag{27}$$

respectively.

Utility function. The performance of the iterative process is characterized by the values of the decrements. Because the iterative process involves Monte Carlo simulations, the values u_k appear to be random. To operate more reliable trend indicators of the iterative process, we organize m simple Monte Carlo simulations with M_k (23) trials on each iteration k and compute the mean values $\bar{u}_k(M_k)$:

$$\bar{u}_k(M_k) = \frac{1}{m}\sum_{h=1}^{m} u_h^{(k)}(M_k). \tag{28}$$

To describe the state of the iterative process, we adopt the concept of *exponential comparative utility* [30,31]. In this context, the utility function $\varphi(\bar{u}_k(M_k))$ is assumed continuously differentiable, positive, and monotonically decreasing in the variable $\bar{u}_k$:

$$\varphi(\bar{u}_k) > 0, \quad \varphi'(\bar{u}_k) < 0 \text{ for all } -\infty < \bar{u}_k < +\infty. \tag{29}$$

Following [30,31], we choose the exponential comparative utility function

$$\varphi(\bar{u}_k, \bar{u}_{k-1} \mid M_k) = \eta \, \exp(\gamma(\bar{u}_k(M_k) - \bar{u}_{k-1}(M_k))) = \varphi_{k,k-1}(M_k), \tag{30}$$

where $\eta > 0$ and $\gamma > 0$ are some parameters.

Payoff function. In the concept of reinforcement, the payoff function reflects the dependence of the payoff r_k on the utility $\varphi_{k,k-1}(M_k)$. By assumption, the payoff grows with increasing the exponential comparative utility. Therefore, the payoff function satisfies the condition $r_k(\varphi_{k,k-1} \mid M_k) > 0$ and is monotonically increasing in the variable $\varphi_{k,k-1}$, i.e.,

$$r_k'(\varphi_{k,k-1} \mid M_k) > 0. \tag{31}$$

The reinforcement decision is taken after accumulating a given number L of the payoffs Q_k by iteration k, i.e., the mean payoff $\bar{Q}_k(M_k)$ over L iterations:

$$\bar{Q}_k(M_k) = \frac{1}{L}\sum_{j=0}^{L} r_{k-L+j}(\varphi_{k-L+j,k-L+j-1} \mid M_k). \tag{32}$$

The value $\bar{Q}_k(M_k)$ is an important characteristic of the reinforcement procedure and is used to optimize the main parameter of MC trials—the number of required random points at the $(k+1)$-th iteration step.

Formation of the Monotonic Sequences of Records

Following the concept of reinforcement, we use a Markov iterative process for RML; the state of this process on iteration $(k+1)$ depends only on the state of the previous iteration $(k.)$

To search for the agent's optimal strategy (the value M_{k+1}), let us use Bellman's optimality principle [13]. In its extended interpretation, the agent's optimal strategy on iteration $(k+1)$ depends on the weighted optimal strategy on iteration k. This principle can be implemented within the additive

$$M_{k+1} = M_k + \alpha \max_{M_k} \bar{Q}_k(M_k) \tag{33}$$

or multiplicative

$$M_{k+1} = M_k \left(\max_{M_k} \bar{Q}_k(M_k) \right)^{\gamma} \tag{34}$$

form of the algorithm. Here, α and γ are some parameters.

Remark 1. *Generally speaking, Bellman's optimality principle is only a declaration here, which sometimes may fail. In particular, learning processes and their internal mechanisms are underinvestigated, and they do not necessarily satisfy the Markov property. As a result, the agent's strategy on iteration* $(k+1)$ *can be formed from the weighted optimal strategies on iterations* $(k-s)-,\ldots,k$. *For example,*

$$M_{k+1} = \sum_{j=0}^{s} \alpha_{k-s+j} \max_{M_{k-s+j}} \bar{Q}_{k-s+j}(M_{k-s+j}), \tag{35}$$

where $\alpha_{k-s},\ldots,\alpha_k$ *are some parameters.*

The reinforcement procedure generates an optimal number M_{k+1} of random values for each iteration. The local record $J^*_{k+1}(M_{k+1})$ and the decrement $u_{k+1}(M_{k+1})$ are then determined for the resulting value M_{k+1}. They are compared to their counterparts obtained on the previous iteration k. If the first record is smaller than the previous one, it becomes a member of the strictly monotonically decreasing sequence of local records. In this case, the sequence of decrements has strictly negative elements:

$$u^*_{k^*_i} < 0, \quad k^*_i \in \mathcal{K}^*. \tag{36}$$

Thus, the *Reinforcement* module has the logical diagram shown in Figure 1. *Agent* is the central block of this diagram. It generates the number M_{k+1} of random values on iteration $(k+1)$ as the sum of the number of random values on the previous iteration k and the optimized component ΔM_k with the parameter α. At each iterative step, the *Optimization* block outputs the maximum $\bar{Q}_k$ of the payoffs r_k accumulated over L iterations (a fixed number), which are calculated in the *Payoff function* block. The necessary values of the comparative utility function $\varphi_{k,k-1}$, records J^*_k and J^*_{k-1}, and their decrements u_k and u_{k-1} are calculated in the *Feedback* block.

Thus, the described reinforcement procedure proves the following assertion:

Let at each step of the iterative process of finding a solution to the Equations (14) and (16) a reinforcement procedure (30)–(32) is carried out, which implements the Belman optimality principle in the form (33) or (34).

Then, a strictly monotonically decreasing sequence of local records is generated:

$$J^*_{k^*_1} > J^*_{k^*_2} > \cdots > J^*_{k^*_i} > \ldots, \quad \mathcal{K}^* = \{k^*_1 < k^*_2 < \cdots < k^*_i < \ldots\} \tag{37}$$

Note that the sequence of local records consists of random elements but satisfies the chain of inequalities (36).

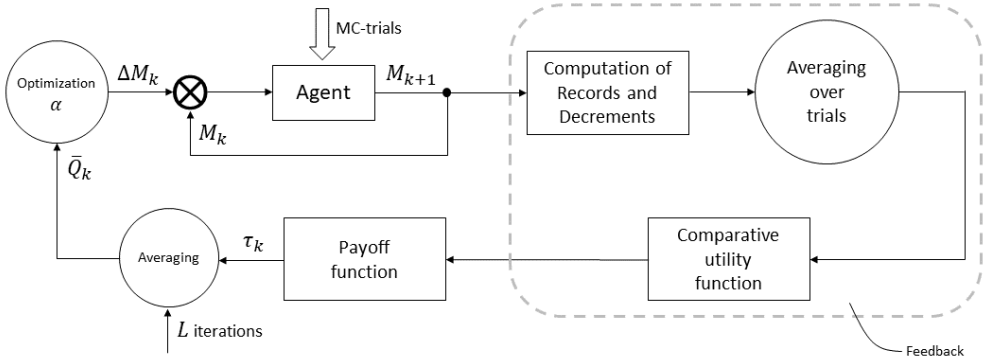

Figure 1. Logical diagram of the *Reinforcement* module.

5. The Probabilistic Properties of Random Sequences Generated by the GFS-RF Algorithm

5.1. The Probabilistic Characteristics of the Packet $\mathcal{Z}_k$

The iterative procedure is based on generating the packet $\mathcal{Z}_k^d$ of random and independent vectors with a uniform distribution on the unit d-dimensional cube. The source of this packet is a random generator that produces on each iteration k a $(d \times M_k)$-dimensional array of independent random variables with a uniform distribution on the interval $[0,1]$.

Consider the d-dimensional unit cube Z_+^d and the grid with step η_k (23). The cube Z_+^d is the union of the elementary cubes with side M_k^{-q}. We estimate the probability $P(M_k, d, q)$ that each elementary cube will contain at least one of the random vectors from the packet $\mathcal{Z}_k^d$ generated on iteration k.

Lemma 1. *The probability $P(M_k, d, q)$ satisfies the upper bound*

$$\begin{aligned} P(M_k, d, q) &\leq 1 - (M_k^{-q/2} + 1)^d \, (1 - M_k^{-qd})^{M_k^d} < \\ &< 1 - (1 - M_k^{-qd})^{M_k^d} = P_0(M_k, d, q). \end{aligned} \tag{38}$$

where

$$P(M_k, d, q) \leq \hat{P}(M_k, d, q) \sim M_k^{q/2} \exp\left(-M_k^{d(1-q/2)}\right) < 1. \tag{39}$$

as $M_k \to \infty$.

Proof. Consider the partition of the interval $[0,1]$ by a grid with step $\eta_k \ll 1$ (23). At least one random value from $M_k = (1/\eta)^{1/q}$ will fall into the elementary interval with the probability η_k. Let this grid be applied to all sides of the unit cube. Then the event A that at least one random vector from $N_k = M_k^d$ will fall into the elementary cube and has the probability η_k^d. Hence, the complementary event (not getting into the elementary cube) has the probability $(1 - \eta_k^d)^{N_k}$.

The upper bounds on the number of elementary intervals and the number of elementary cubes are $(1 + M_k^{-q/2})$ and $(1 + M_k^{-q/2})^d$, respectively. Therefore, the upper bound on the probability of the event A is given by

$$\hat{P}(M_k, d, q) = (1 + M_k^{-q/2})^d \, (1 - \eta_k^d)^{N_k}. \tag{40}$$

Due to the relation (25) between M_k and N_k, we finally arrive at the upper bound (38). For large values $M_k = x$,

$$\lim_{x \to \infty} \frac{(1 - x - q)^x}{\exp(-x^{1-q})} = 1,$$

which yields (39). □

5.2. The Probabilistic Properties of the Local Record Sequence (36)

The reinforcement procedure forms the strictly monotonically decreasing sequence J^* of local records and the sequence of their arguments $\mathbf{z}^*$. Because of their strict monotonic decrease, it is more convenient to renumber the elements by integers $1, 2, \ldots, i, \ldots$:

$$\begin{aligned} \mathcal{J}^* &= \{J_1^* > J_2^*, > \ldots, > J_i^*, > \ldots\} \\ \mathbf{z}^* &= \{\mathbf{z}_1^*, \mathbf{z}_2^*, \ldots, \mathbf{z}_i^*, \ldots\}. \end{aligned} \tag{41}$$

Let $\mathbb{Z}$ denote the set of points $\mathbf{z}^0$ corresponding to the zero value of the residual function: $J(\mathbf{z}^0) = J^* = 0$ (20). Due to the continuity of the function $J(\mathbf{z})$, this set is compact.

We introduce the distance between an arbitrary point in the cube and the set $\mathbb{Z}$:

$$\varrho(\mathbf{z}, \mathbb{Z}) = \min_{\mathbf{y} \in \mathbb{Z}} \|\mathbf{z} - \mathbf{y}\|. \tag{42}$$

The elements of the local record sequence are ordered but random values. Therefore, the deviation from the global record (the global minimum) takes a random value J_i^* on each iteration. Using the assumption that the residual function (19) has a modulus of continuity $\omega(H, h)$ (22), we can formulate the following Lemma 2.

Lemma 2. *For a finite number of iterative steps i with a probability not smaller than $P_0(M_i, d, q)$ (38) and (39), we have the bilateral estimate*

$$Pb\{0 \le J_i^* - J^* \le \omega(H, h_i)\} > P_0(M_i, d, q), \tag{43}$$

where $\omega(H, h_i)$ denotes the modulus of continuity of the function $J(\mathbf{z})$ (22), and $h_i = \frac{\sqrt{d}}{2} M_i^{-q}$.

Proof. Consider the random points generated on iteration i among them, let $\hat{\mathbf{z}}$ be the closest one to the set $\mathbb{Z}$ in terms of the distance (42).

At least one of these points will fall with a probability not smaller than $P_0(M_i, d, q)$ into each elementary cube with side M_i^{-q}; see Lemma 1. Hence,

$$h_i = \|\hat{\mathbf{z}} - \mathbf{z}^0\| \le M_i^{-q} \frac{\sqrt{d}}{2}. \tag{44}$$

This happens if the point $\mathbf{z}^0$ corresponding to the zero value of the residual function is in the center of the elementary cube with side M_i^{-q} and its nearest random points are in the cube vertices so that each cube contains at least one random point.

By the Hölder condition (22), we have

$$0 \le J(\hat{\mathbf{z}}) - J^* \le \omega(h_i). \tag{45}$$

On the other hand,

$$J^* = J(\mathbf{z}^0) = \min_{\mathbf{z} \in Z^d} J(\mathbf{z}) \le \min_{s, 1 \le s \le i} J(\mathbf{z}^s) = J_i^* \le J(\hat{\mathbf{z}}). \tag{46}$$

This chain of inequalities implies

$$0 \le J_i^* - J^* \le J(\hat{\mathbf{z}}) - J^*. \tag{47}$$

From (45) and (47) it follows that

$$0 \le J_i^* - J^* \le \omega(H, h_i) \tag{48}$$

with a probability not smaller than $P_0(M_i, d, q)$. □

Inequality (48) provides an upper bound on the deviation from the zero value of the residual function on each iteration obtained after the reinforcement procedure and a lower bound on the probability $P_0(M_i, d, q)$ (38) of its realization. The upper bound is the value of the modulus of continuity of the function J on these iterations. In other words, according to (22),

$$\omega(H, h_i) \le H h_i^s = H\left(\eta_i \frac{\sqrt{d}}{2}\right)^s = H\left(M_i^{-q} \frac{\sqrt{d}}{2}\right)^s. \tag{49}$$

With the notations

$$D = H\left(\frac{\sqrt{d}}{2}\right)^s, \; p = sq,$$
$$r_i(D, p) = D\, M_i^{-p}. \tag{50}$$

we arrive at a very useful probabilistic form of inequality (49):

$$P\{0 \le J_i^* - J^* \le r_i(D, p)\} \ge P_0(M_i, d, q). \tag{51}$$

It gives a lower bound on the probability that the current record will fall into the neighborhood of the global minimum as well as determines its size.

5.3. The Size of the Neighborhood of the Global Minimum

Consider a sequence of decrements on a finite number of iterations k:

$$u_k = J_k^* - J_{k-1}^* < 0, \; \forall\, k = 1, 2, \ldots \tag{52}$$

We represent the decrements as

$$|u_k| = |(J_k^* - J^*) - (J_{k-1}^* - J^*)|. \tag{53}$$

Due to (51)

$$|u_k| \le |(J_{k-1}^* - J^*)| - |(J_k^* - J^*)| \le D\left(M_{k-1}^{-p} - M_k^{-p}\right) = D\, M_k^{-p}\, \beta_k(p) \le D\, M_k^{-p}, \tag{54}$$

where, due to (33)

$$\beta_k(p) = 1 - \left(\frac{M_{k-1}}{M_k}\right)^{-p}, \quad \beta_k(p) \in [0, 1], \; \forall p, k. \tag{55}$$

The boundary value of the modulus of continuity of the decrement for k iterations is

$$u_k^* = D\, M_k^{-p}, \tag{56}$$

or, in the logarithmic scale,

$$\log u_k^* = \log D - p \, \log M_k. \tag{57}$$

Thus, we have a linear dependence with unknown parameters $\log D$ and p, which are related to the parameters of the modulus of continuity (22). Their values can be estimated using the available data on $\log u_k^*$ and $\log M_k$ by the least squares method. The parameters D and p determine the size of the neighborhood of the global minimum and the probability of reaching it (50) and (51).

Remark 2. *The upper bound (54) is very conservative: it focuses on estimating the elements of the local record sequence and neglects an essential feature of the decrement sequence. In the latter,*

the number of random values on iteration $(k+1)$ *changes compared to iteration* k *due to the reinforcement procedure (33) and (34).*

This feature is reflected in the expression for the decrement boundary value:

$$u_k^* = D\, M_k^{-p}\, \beta_k(p), \quad \beta_k(p) = 1 - \left(\frac{1}{\tilde{M}_k}\right)^p, \tag{58}$$

where the reinforcement procedure (34) generates the values

$$\tilde{M}_k = \arg\max_{M_k} \bar{Q}_k(M_k). \tag{59}$$

By analogy with (57), we obtain

$$\log u_k^* = \log D - p\, \log M_k + \log \beta_k(\tilde{M}_k, p). \tag{60}$$

This dependence still has two parameters, D and p, but the data include $\log u_k^*$, M_k, and $\tilde{M}_k$ additionally generated by the reinforcement procedure. The dependence (60) is nonlinear. Its parameters can be restored using the least squares method as well. As in the previous case, however, there is no guarantee of obtaining the optimal result.

6. The Convergence of the GFS-RF Algorithm to the Global Minimum

The reinforcement procedure (30)–(33), combined with the selection of local records, makes their sequence the property of a strictly monotonic decrease (37), accompanied by a sequence of decrements with negative elements (36). Based on them, we can formulate the following Theorem 1.

Theorem 1. *Let the following conditions be satisfied for a finite number of iterations equal to k:*

(а)*. inequalities (37) and (36) are true;*

(б)*. function* $J(\mathbf{z})$ *is of Hoelder type with parameters of mudulus of continuity* (H^*, h_k^*) *which are estimated by (57);*

(в)*. area* $\mathcal{R}_k^*$ *of the existence of global extrema is as follows:*

$$\mathcal{R}_k^* = \{J : |J_k^* - J^*| \le r_k(D,p)\}, \tag{61}$$

where

$$r_k(D,p) = DM_k^{-p}. \tag{62}$$

Then the sequence of local records $\mathcal{J}_k^* = \{J_1^* > J_2^*, > \cdots > J_k^*\}$ *at k iterations achieve the area* $\mathcal{R}_k^*$ *with probability not less than*

$$P_0(M_k, d, q) = 1 - \left(1 - M_k^{dq}\right)^{M_k^d}, \tag{63}$$

and at high values of M_k

$$P_0(M_k, d, q) \sim M_k^{q/2} \exp\left(-M_k^{d(1-q/2)}\right). \tag{64}$$

Proof. The proof follows from Lemmas 1 and 2 and the estimate (51). □

7. Discussion and Conclusions

The concept and computational procedure of Randomized Machine Learning proposed in [22] turned out to be very useful in terms of inaccurate data estimation *probability distributions*, and also an effective computer technique for solving many applied problems [24]. The modules of this procedure have been applied to practical problems of the randomized forecasting of World population [32], electrical load in the power systems [33],

the evolution of the thermokarst lakes in the Arctic zone [34], randomized classification of the objects [35,36]. In these works, we used public datasets of the UN [37], and [38]. However, its practical application is associated with solving a very difficult problem of finding solutions to a specific system of nonlinear equations in which only the values of the functions included in it are available.

In this paper, we propose to use the idea of reinforcement to give adaptive properties to computational algorithms. A problem-oriented reinforcement procedure based on the agent-based approach is proposed, in which the agent generates a strategy in terms of the optimal number of random numbers generated at each step of the iterative process. As a utility function, the exponential comparative utility function is used, which depends on the average decrements of local records achieved at each main iteration. An important role in the reinforcement procedure is played by the payoff function, which generates "penalties" on the values of the utility function. Optimization of the agent's strategy is carried out using R. Belman's principle of optimality. As a result of applying the reinforcement procedure, the dimensions of the neighborhood of the global minimum of the quadratic residual function and the probability of its achievement with a finite number of iterations are determined.

Author Contributions: Conceptualization, Y.S.P.; Data curation, A.Y.P.; Methodology, Y.S.P., A.Y.P. and Y.A.D.; Software, A.Y.P. and Y.A.D.; Supervision, Y.S.P.; Writing–original draft, Y.S.P., A.Y.P. and Y.A.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Higher Education of the Russian Federation, project no. 075-15-2020-799.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rosenblatt, F. *Principles of Neirodynamic: Perceptrons and the Theory of Brain Mechanisms*; Spartan Books: Washington, DC, USA, 1962.
2. Gelfand, I.M.; Pyatetskij-Shapiro, I.I.; Tsetlin, M.L. Certain Classes of Games and Automata Games. *Sov. Phys. Dokl.* **1964**, *8*, 964–966.
3. Wasserman, P.D. *Neural Computing: Theory and Practice*; Van Nostrand Reinhold Co.: New York, NY, USA, 1992.
4. Kohonen, T. *Self-Organizing Maps*; Springer: Berlin/Heidelberg, Germany, 1995.
5. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A. Human-level Control through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
6. Sutton, R.S.; Barto, A.G. *Introduction to Reinforcement Learning*; MIT Press: Cambridge, UK, 1998.
7. Russel, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2010.
8. van Hasselt, H. Reinforcement Learning in Continuous State and Action Spaces. In *Reinforcement Learning: State-of-the-Art*; Wiering, M., van Otterio, M., Eds.; Springer Sciences & Business Media: Berlin/Heidelberg, Germany, 2012; pp. 207–257.
9. Kropotov, D.; Bobrov, E.; Ivanov, S.; Temirchev, P. Reinforcement Learning Textbook. *arXiv* **2022**, arXiv:2201.09746v1. (In Russian)
10. Bozinovski, S. Crossbar Adaptive Array: The First Connectionist Network That Solved the Delayed Reinforcement Learning Problem. In *Artificial Neural Nets and Genetic Algorithms*; Dobnikar, A., Steele, N.C., Pearson, D.W., Albrecht, R.F., Eds.; Springer Science & Business Media: Berlin/Heidelberg, Germany, 1999; pp. 320–325.
11. Watkins, C.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
12. van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-learning. In Proceedings of the 13th AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 2094–2100.
13. Bellman, R. *Dynamic Programming*; Princeton University Press: Princeton, NJ, USA, 1957.
14. Robbins, H.; Monro, S. A Stochastic Approximation Method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
15. Spong, M.W.; Hutchinson, S.; Vidyasagar, M. *Robot Modeling and Control*; Wiley: Hoboken, NJ, USA, 2005.
16. Koshmanova, N.P.; Pavlovsky, V.E.; Trifonov, D.S. Reinforcement Learning for Manipulator Control. *Rus. J. Nonlin. Dyn.* **2012**, *8*, 689–704. (In Russian) [CrossRef]
17. Fu, Y.; Jha, D.K.; Zhang, Z.; Yuan, Z.; Ray, A. Neural Network-Based Learning from Demonstration of an Autonomous Ground Robot. *Machines* **2019**, *7*, 24. [CrossRef]

18. Nikitin, P.V.; Gorokhova, R.I.; Korchagin, S.A.; Krasnikov, V.S. Applying Deep Reinforcement Learning to Algorithmic Trading. *Mod. Inf. Technol. IT-Educ.* **2020**, *16*, 510–517. (In Russian)
19. Esfahani, N.; Malek, S. Uncertainty in Self-Adaptive Software Systems. In *Software Engineering for Self-Adaptive Systems II*; de Lemos, R., Giese, H., Müller, H.A., Shaw, M., Eds.; Lecture Notes in Computer Science Book Series; Springer: Berlin/Heidelberg, Germany, 2013; pp. 214–238. [CrossRef]
20. Ghezzi, C.; Salvaneschi, G.; Pradella, M. ContextErlang. *Sci. Comput. Program.* **2015**, *102*, 20–43. [CrossRef]
21. Bencvoma, N.; Belaggoun, A. Supporting Decision-Making for Soft-Adaptive Systems: From Goal Models to Dynamic Decision Network. In *Requirements Engineering: Foundation for Software Quality, proceedings of the 19th International Working Conference, REFSQ 2013, Essen, Germany, 8–11 April 2013*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 221–236.
22. Popkov, Y.S.; Popkov, A.Y. New Nethod of Entropy-Robust Estimation for Randomized Models under Limited Data. *Entropy* **2014**, *16*, 675–698. [CrossRef]
23. Popkov, Y.S.; Dubnov, Y.A.; Popkov, A.Y. Randomized Machine Learning: Statement, Solution, Applications. In Proceedings of the IEEE 8th International Conference on Intelligent Systems, Sofia, Bulgaria, 4–6 September 2016; pp. 27–39.
24. Popkov, Y.S.; Popkov, A.Y.; Dubnov, Y.A. *Entropy Randomization in Machine Learning*; CRC Press: Boca Raton, FL, USA, 2023.
25. Darkhovskii, B.S.; Popkov, A.Y.; Popkov, Y.S. Monte Carlo Method of Batch Iterations: Probabilistic Characteristics. *Autom. Remote Control* **2015**, *76*, 775–784. [CrossRef]
26. Popkov, A.Y.; Darkhovskii, B.S.; Popkov, Y.S. Iterative MC-Algorithm to Solve the Global Optimization Problems. *Autom. Remote Control* **2017**, *78*, 261–275. [CrossRef]
27. Avellaneda, M. Minimum-Relative-Entropy Calibration of Asset-Pricing Models. *Int. J. Theor. Appl. Financ.* **1998**, *1*, 447–472. [CrossRef]
28. Vine, S. *Options: Trading Strategy and Risk Management*, 1st ed.; Wiley: Hoboken, NJ, USA, 2005.
29. Magnus, J.R.; Neudecker, H. *Matrix Differential Calculus (with Applications in Statistics and Econometrics)*; John Wiley and Sons: New York, NY, USA, 1999.
30. von Neumann, J.; Morgenstern, O. *Theory of Games and Economic Behavior*; Princeton Univiversity Press: Princeton, NJ, USA, 1944.
31. Fishburn, P.C. *Utility Theory for Decision Making*; Wiley: New York, NY, USA, 1970.
32. Popkov, Y.S.; Dubnov, Y.A.; Popkov, A.Y. New Method of Randomized Forecasting Using Entropy-Robust Estimation: Application to the World Population Prediction. *Mathematics* **2016**, *4*, 16. [CrossRef]
33. Popkov, Y.S.; Popkov, A.Y.; Dubnov, Y.A.; Solomatine, D. Entropy-Randomized Forecasting of Stochastic Dynamic Regression Models. *Mathematics* **2020**, *8*, 1119. [CrossRef]
34. Dubnov, Y.A.; Popkov, A.Y.; Polyschuk, V.Y.; Sokol, E.A.; Melnikov, A.V.; Polyschuk, Y.M.; Popkov, Y.S. Randomized Machine Learning to Forecast the Evolution of Thermokarst Lakes in Permafrost Zones. *Autom. Remote Control* **2023**, *84*, 56–70. [CrossRef]
35. Popkov, Y.S.; Volkovich, Z.; Dubnov, Y.A. Entropy "2"-Soft Classification of Objects. *Entropy* **2017**, *19*, 178. [CrossRef]
36. Dubnov, Y.A. Entropy-Based Estimation in Classification Problems. *Autom. Remote Control* **2019**, *80*, 502–512. [CrossRef]
37. UNdata—A World of Information. Available online: https://data.un.org (accessed on 17 August 2023).
38. Hong, T.; Prinson, P.; Fan, S.; Zareipour, H.; Triccoli, A.; Hyndman, R.J. Probabilistic Energy Forecasting: Global Energy Forecasting Competition 2014 and Beyond. *Int. J. Forecast.* **2016**, *32*, 896–913. [CrossRef]