ISSN 0005-1179 (print), ISSN 1608-3032 (online), Automation and Remote Control, 2024, Vol. 85, No. 5, pp. 544–548.


**TOPICAL ISSUE**

# Iterative Methods with Self-Learning for Solving Nonlinear Equations

**Yu. S. Popkov**[*,**]

[*] *Federal Research Center "Computer Science and Control," Russian Academy of Sciences, Moscow, Russia*
[**] *Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*
*e-mail: popkov@isa.ru*

Received January 25, 2024
Revised March 12, 2024
Accepted March 20, 2024

**Abstract**—This paper is devoted to the problem of solving a system of nonlinear equations with an arbitrary but continuous vector function on the left-hand side. By assumption, the values of its components are the only a priori information available about this function. An approximate solution of the system is determined using some iterative method with parameters, and the qualitative properties of the method are assessed in terms of a quadratic residual functional. We propose a self-learning (reinforcement) procedure based on auxiliary Monte Carlo (MC) experiments, an exponential utility function, and a payoff function that implements Bellman's optimality principle. A theorem on the strict monotonic decrease of the residual functional is proven.

*Keywords*: nonlinear equation, iterative methods, reinforcement, Monte Carlo experiment

**DOI:** 10.31857/S0005117924050076

## 1. INTRODUCTION

The vast majority of applied problems lead to the need to solve nonlinear equations. Parameterized iterative methods are a classical tool yielding approximate solutions of nonlinear equations under definite conditions [1–4]. These conditions are certain properties of the functions (convexity, concavity, differentiability, etc.) included in the equations and interval *sufficient* parametric conditions ensuring the convergence of the corresponding iterative method.

The increasing complexity of functions narrows the set of their classes where such properties can be verified and the verification results can be used in suitable iterative methods. On the other hand, interval conditions on the parameters of iterative methods significantly depend on the properties of functions that are generally unverifiable.

To find a way out of this situation, the ideas of reinforcement learning can be applied to the iterative computational process to determine the values of its parameters using statistical Monte Carlo (MC) experiments and a game-theoretic mathematical model [5, 6]. The essence of this branch of machine learning is to train an object (model, algorithm, etc.) by interacting not with a "teacher" (supervised learning) but with an "environment," using the trial-and-error method, followed by rewards or penalties for its results.

This approach was employed in clustering and recognition problems, apparently because of the ability to calculate the so-called feature characteristics in the form of "distances" between objects. The distance matrix was adopted to arrange some "rewards" or "penalties" when tuning the algorithm parameters. The algorithms considered were neural networks [7] and game-theoretic

models implementing the principle of competition between neural network nodes [8]. In particular, the advantage was given to the nodes with the minimum distance between objects at each step of the algorithm.

Subsequently, reinforcement learning based on the automata models of interaction between an object (agent) and an environment was simulated in game-theoretic terms (strategies, utility functions, and payoffs) and was actively developed [9]. Many algorithms appeared, differing in the models and amounts of a priori information about the environment, algorithms for choosing strategies, and procedures for designing utility functions [10–13].

An important component of reinforcement learning procedures is MC experiments intended to simulate agent's strategies [14]. They are used to average a fixed number of current rewards with their discounting. The resulting function depends on the state of the environment and the agent's strategy; it is taken as a utility function (an analog of the objective function in supervised learning procedures) and is sequentially maximized during the learning process [15, 16] using Bellman's optimality principle [17] in combination with stochastic approximation [18].

This paper considers the problem of numerically solving a system of nonlinear equations using a parameterized iterative procedure whose convergence depends on the parameter values. To determine these values, we develop a reinforcement learning procedure based on a game-theoretic model.

The problem addressed below was actively discussed and shaped under the influence of Boris Polyak. The author was fortunate to work and be friends with him for many years. The blessed memory of Boris will always be the author's compass in life.

## 2. PROBLEM STATEMENT

Consider the nonlinear equation

$$\mathbf{f}(\mathbf{x}) = \mathbf{1}, \quad (\mathbf{f}, \mathbf{x}, \mathbf{1}) \in R^n. \tag{1}$$

By assumption, the only a priori information available about the function $\mathbf{f}$ is the values of its components $f_i(\mathbf{x}^{(k)})$, $i = \overline{1,n}$; $k = 1, \ldots$.

We introduce the residual functional

$$J(\mathbf{x}) = \|\mathbf{f}(\mathbf{x}) - \mathbf{1}\|^2 \geqslant 0. \tag{2}$$

The absolute minimum of this functional is zero. In the general case, it is not unique, i.e., there exists a finite set $\mathbb{X} = \{\mathbf{x}_*^{(1)}, \ldots, \mathbf{x}_*^{(r)}\}$ of points at which the residual functional vanishes. In this situation, any solution from the set $\mathbb{X}$ will be considered suitable.

An approximate solution of this equation is determined using an iterative Markov-type procedure. In this procedure, the approximate solution $\mathbf{x}^{(p+1)}$ at step $(p+1)$ is set equal to the value of the operator $\mathcal{B}[\mathbf{x}^{(p)}, \mathbf{a}^{(p)}]$ of the iterative procedure at step $p$, depending on the values of the components of the function $\mathbf{f}(\mathbf{x}^{(p)})$ and the parameter vector $\mathbf{a}^{(p)} \in \mathcal{A} \subset R^r$ that adjusts the qualitative properties of the iterative process:

$$\mathbf{x}^{(p+1)} = \mathcal{B}[\mathbf{f}(\mathbf{x}^{(p)}), \mathbf{a}^{(p)}]. \tag{3}$$

Qualitative properties often include *convergence*, *the rate of convergence*, and *accuracy*. The conditions for ensuring these properties are formulated in terms of the vector $\mathbf{a}$ and interval inequalities depending on the properties of the operator $\mathcal{B}$ and those of the function $\mathbf{f}(\mathbf{x})$.

However, the function $\mathbf{f}(\mathbf{x})$ may have an arbitrary structure, making it impossible to postulate or reveal its properties. As a result, these inequalities become analytically unverifiable.

To find a way out of this situation, we utilize the ideas of *reinforcement*, which are actively used in various implementations in modern machine learning procedures. In this case, reinforcement is intended to determine, at each step of the iterative procedure, a suitable parameter vector $\mathbf{a}$ via an appropriate self-learning procedure.

We propose a game-theoretic model to calculate the suitable parameters $\mathbf{a}$ of the iterative procedure (3). This model operates in the intervals between steps $p$ and $(p+1)$ and simulates the behavior of an *agent*, i.e., a strategy for varying the parameters $\mathbf{a}$ depending on the response quality of an *environment*. The quality is characterized by a conditional *payoff*, a function that depends on the values of the residual functional and its decrement.

## 3. STRUCTURE OF THE REINFORCEMENT PROCEDURE

Consider the original problem (1) and find its solution by minimizing the residual functional

$$J(\mathbf{x}) \Rightarrow \min, \quad \mathbf{x} \in R^n. \tag{4}$$

In some applications, it may be useful to transform problem (1). We introduce the new variables

$$z_i = \frac{1}{1+\exp(-b_i x_i)}, \quad x_i = \frac{1}{b_i} \ln \frac{z_i}{1-z_i}, \quad i = \overline{1,n}.$$

Then problem (1) takes the form

$$J(\mathbf{z}) = \|\mathbf{\Psi}(\mathbf{z})\| \Rightarrow \min, \ \mathbf{z} \in Z_+^n = [\mathbf{0},\mathbf{1}], \quad \mathbf{\Psi}(\mathbf{z}) = \mathbf{f}(\mathbf{z}) - \mathbf{1}.$$

Problem (1) will be solved using the iterative procedure (3) under the assumption that the parameters $\mathbf{a}$ are of interval type: $\mathbf{a} \in [\mathbf{a}^-, \mathbf{a}^+]$.

To determine the values of the components of the vector $\mathbf{a}$ in the procedure (3), we employ the *reinforcement* technology, implementing it in the interval between steps $p$ and $(p+1)$ of the iterative procedure (3).

This technology is based on a game-theoretic model simulating the game of an *agent* with an *environment*. The agent generates *strategies* (actions) that cause changes in the environment. The magnitude of these changes is characterized by a *utility function*. The value of a *payoff function* depends on the success of the agent's strategy and its utility for the environment.

In the interval between the successive steps of the iterative procedure, a statistical simulation is carried out via a given number $M$ of MC experiments that simulate the agent's strategies, i.e., the values of the components of the vector $\mathbf{a}^{(p,k)}$, where $k = \overline{1,M}$.

As the agent's actions, we will consider the vector

$$\mathbf{x}^{(p,k+1)} = \mathcal{B}[\mathbf{f}(\mathbf{x}^{(p,k)}), \mathbf{a}^{(p,k)}], \quad p = \text{fix}, \ k = \overline{1,M}. \tag{5}$$

In this problem, the environment is the residual functional $J(\mathbf{x} \,|\, \mathbf{a})$. The MC-simulated actions of the agent yield a sequence of $M$ residuals,

$$J(\mathbf{x}^{(p,1)} \,|\, \mathbf{a}^{(p,1)}), \ldots, J(\mathbf{x}^{(p,M)} \,|\, \mathbf{a}^{(p,M)}), \tag{6}$$

and their decrements,

$$u^{(p,k)}(\mathbf{a}^{(p,k)}) = J(\mathbf{x}^{(p,k+1)} \,|\, \mathbf{a}^{(p,k)}) - J(\mathbf{x}^{(p,k)} \,|\, \mathbf{a}^{(p,k-1)}), \quad k = \overline{1,M}. \tag{7}$$

We introduce a *utility function* to characterize the response quality of the environment measured by the decrement:

$$\varphi(u^{(p,k)}(\mathbf{a}^{(p,k)})) = \alpha \, \exp[u^{(p,k)}(\mathbf{a}^{(p,k)})]. \tag{8}$$

The quality of the agent's strategies is assessed in terms of a *payoff function* that characterizes the dependence of the agent's payoff on its strategy. Choosing an appropriate payoff function seems to be a creative task [2] involving some enumeration. Several general properties of this function can be declared. It is a continuous and bounded function of the following form:

$$Q(\mathbf{a}^{(p,k)}) = \begin{cases} l(u^{(p,k)}(\mathbf{a}^{(p,k)})) & \varphi(u^{(p,k)}(\mathbf{a}^{(p,k)})) \leqslant 1 \\ 0 & \varphi(u^{(p,k)}(\mathbf{a}^{(p,k)})) > 1, \end{cases} \tag{9}$$

with the function

$$l(u^{(p,k)}(\mathbf{a}^{(p,k)})) = \begin{cases} \alpha\,\varphi(u^{(p,k)}(\mathbf{a}^{(p,k)})), & 0 \geqslant u^{(p,k)}(\mathbf{a}^{(p,k)}) \geqslant -U, \\ 0, & u^{(p,k)}(\mathbf{a}^{(p,k)}) \geqslant 0, \end{cases} \tag{10}$$

where $U$ is the limit of the decrement's magnitude.

MC experiments yield a value set of the payoff functions. Following the concept of reinforcement as applied to the iterative procedure (3), we determine the optimal value of the parameter $\mathbf{a}^{(p+1)}$ by the rule

$$\mathbf{a}^{(p+1)} = \mathbf{a}^{(p)} + \beta \arg \max_{1 \leqslant j \leqslant M} Q(\mathbf{a}^{(p,k_j)}). \tag{11}$$

If the agent chooses its strategy by the rule (11), in view of (9), we have

$$J(\mathbf{a}^{(p+1)}) < J(\mathbf{a}^{(p)}). \tag{12}$$

Thus, the following result has been established for the properties of the residual sequence in the iterative reinforcement procedure (8)–(11):

**Theorem 1.** *Assume that:*

*a) The only a priori information available about the function* $\mathbf{f}(\mathbf{x})$ *in* (1) *is the values of its components* $f_i(\mathbf{x}^{(k)})$, $i = \overline{1,n}$.

*b) The parameters of the iterative procedure* $\mathbf{a}$ *are chosen by the rule* (12), (11), (9).

*Then the iterative procedure* (5) *with reinforcement* (8)–(11) *generates a strictly monotonically decreasing sequence of the residual functionals* $J(\mathbf{x})$ (4).

This theorem is not a convergence theorem for the iterative procedure in the mathematical sense (convergence to one of the solutions). However, it is known that this solution corresponds to a zero residual value. The theorem states that the sequence of residuals is strictly monotonically decreasing. Since the calculation error is finite and can be specified, the final value of the parameters $\mathbf{a}$ obtained when reaching this error can be taken as a solution.

## 4. CONCLUSIONS

This paper has been devoted to the problem of solving a system of nonlinear equations with continuous functions on the left-hand sides. By assumption, the only a priori information available about these functions is their values. To find solutions under such conditions, an iterative procedure with parameters has been used: by tuning their values, it is possible to ensure the convergence of the procedure in some sense.

It has been proposed to employ the ideas of reinforcement, which are being rather actively developed in the theory and practice of machine learning. A self-learning procedure has been designed in which a given number of MC experiments are carried out at each iteration step to simulate the agent's strategy (the values of the parameters of the iterative procedure). In this procedure, the environment is the residual functional (5), and its response to the agent's actions is

the decrement (6). For an acceptable evolution of the iterative process, the decrement magnitude must decrease. The decrement has been characterized by an exponential utility function so that smaller decrement magnitudes correspond to larger values of the utility function. The agent's actions, i.e., the implemented parameters of the iterative procedure, have been assessed in terms of a payoff function whose morphology considers both the state of the environment and the degree of success of the agent's actions.

It has been proven that, due to this self-learning procedure, the iterative reinforcement algorithm generates a strictly monotonically decreasing sequence of residual functionals.

## REFERENCES

1. Krasnosel'skii, M.A., Vainikko, G.M., Zabreiko, P.P., Rutitski, Ja.B., and Stecenko, V.Ja., *Approximated Solutions of Operator Equations*, Groningen: Walters-Noordhoff, 1972.
2. Bakhvalov, N.S., Zhidkov, N.P., and Kobel'kov, G.M., *Chislennye metody* (Numerical Methods), Moscow: Binom, 2003.
3. Polyak, B.T., *Introduction to Optimization*, Optimization Software, 1987.
4. Strekalovsky, A.S., *Elementy nevypukloi optimizatsii* (Elements of Nonconvex Optimization), Novosibirsk: Nauka, 2003.
5. Lyle, C., Rowland, M., Dabney, W., Kwiatkowska, M., and Gal, Y., Learning Dynamics and Generalization in Deep Reinforcement Learning, *Proceedings of the 39th International Conference on Machine Learning (PMLR)*, 2022, vol. 162, pp. 14560–14581.
6. Wang, C., Yaun, S., and Ross, K.W., On the Convergence of the Monte Carlo Exploring Starts Algorithm for Reinforcement Learning, *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.
7. Wasserman, Ph.D., *Neural Computing: Theory and Practice*, Coriolis Group, 1989.
8. Kohonen, T., *Self-organizing Maps*, Berlin–Heidelberg: Springer, 1995.
9. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., and Fidjeland, A., Human-Level Control through Deep Reinforcement Learning, *Nature*, 2015, vol. 518, no. 7540, pp. 529–533.
10. Sutton, R.S. and Barto, A.G., *Introduction to Reinforcement Learning*, Cambridge: MIT Press, 1998.
11. Russel, S.J. and Norvig, P., *Artificial Intelligence: A Modern Approach*, 3rd ed., Upper Saddle River: Prentice Hall, 2010.
12. van Hasselt, H., Reinforcement Learning in Continuous State and Action Spaces, in *Reinforcement Learning: State-of-the-Art*, Wiering, M. and van Otterio, M., Eds., 2012, Springer, pp. 207–257.
13. Ivanov, S., Reinforcement Learning Textbook, *ArXiv*, 2022. `https://doi.org/10.48550/arXiv.2201.09746`.
14. Bozinovski, S., Crossbar Adaptive Array: The First Connectionist Network That Solved the Delayed Reinforcement Learning Problem, in *Artificial Neural Nets and Genetic Algorithms*, Proc. Int. Conf., Portoroz, Slovenia, Dobnikar, A., Steele, N.C., Pearson, D.W., and Albrecht, R.F., Eds., Springer, 1999, pp. 320–325.
15. Watkins, C. and Dayan, P., Q-learning, *Machine Learning*, 1992, vol. 8, no. 3–4, pp. 279–292.
16. van Hasselt, H., Guez, A., and Silver, D., Deep Reinforcement Learning with Double Q-learning, *Proc. AAAI Conf. Artificial Intelligence*, 2016, vol. 30, no. 1, pp. 2094–2100.
17. Bellman, R., *Dynamic Programming*, Princeton: Princeton University Press, 1957.
18. Robbins, H. and Monro, S., A Stochastic Approximation Method, *The Annals of Mathematical Statistics*, 1951, vol. 22, no. 3, pp. 400–407.

*This paper was recommended for publication by P.S. Shcherbakov, a member of the Editorial Board*