ISSN 0965-5425, Computational Mathematics and Mathematical Physics, 2020, Vol. 60, No. 7, pp. 1108–1115. © Pleiades Publishing, Ltd., 2020.
Russian Text © The Author(s), 2020, published in Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki, 2020, Vol. 60, No. 7, pp. 1143–1150.

A Heuristic Adaptive Fast Gradient Method in Stochastic Optimization Problems

A. V. Ogal'tsov[a] and A. I. Tyurin[a,*]

[a] *State University—Higher School of Economics, Moscow, 101000 Russia*

**e-mail: atyurin@hse.ru*

Received October 7, 2019; revised November 12, 2019; accepted March 10, 2020

Abstract— A fast adaptive heuristic stochastic gradient descent method is proposed. It is shown that this algorithm has a higher convergence rate in practical problems than currently popular optimization methods. Furthermore, a justification of this method is given, and difficulties that prevent obtaining optimal estimates for the proposed algorithm are described.

Keywords: fast gradient descent, stochastic optimization, adaptive optimization

DOI: 10.1134/S0965542520070088

1. INTRODUCTION

In this paper, we propose a heuristic algorithm for solving stochastic optimization problems that is a generalization of the fast gradient descent method [1]. Stochastic optimization methods have recently become popular because they reduce the complexity of calculating the gradient, which is extremely important since there are functions for which it is impossible to calculate the gradient of the function to be optimized even at one point in a reasonable amount of time (see [2, 3]). Moreover, due to the problem formulation in stochastic optimization [4], sampling of vectors that have unbiased directions with respect to the true gradient is the only reasonable method for calculating the descent direction. This approach is also popular in deep learning [3] in problems in which the objective function have the form of a sum of functions [4]. Note that the complexity of computing the stochastic gradient can usually be controlled. We also note that a slower version of the proposed algorithm can be found in [5].

The problem of finding a fast adaptive gradient descent method with a stochastic gradient is still important. As far as we know, it has not yet been solved, and the work in this direction is going on. In [6], a method that is effective both for smooth and nonsmooth problems was proposed. However, it is not accelerated—the step is chosen in such a way that it does not increase, while in the method proposed in the current paper the step is chosen adaptively, and it can both increase or decrease. In [7], it was proposed to combine the unaccelerated stochastic gradient descent method with Armijo's rule [8]; as a result, it was proved that the proposed method has the same convergence rate as the unaccelerated gradient descent method; however, this was done only under certain special assumptions about the optimization problem. In [9], a proof of a version of the method Adagrad [10] for the case when the function is smooth and nonconvex was proposed; however, as in [6], the step of the method does not increase and the method is not fully adaptive and cannot adjust to local smoothness. A similar difficulty occurs in [11, 12], but in these papers accelerated versions of the gradient descent method and more flexible step size in the algorithms are proposed. In [12], the methods depend on an estimate of the distance between the initial and the optimal points (size of solution). A considerable amount of work on finding a fast adaptive stochastic gradient descent method was done in [13]. In that paper, an adaptive stochastic method for variational inequalities was proposed, and the proof of convergence of this method is based on the theory of empirical processes [14, 15]. Note that the unaccelerated gradient descent method is a special case of the method designed for solving variational inequalities, while the accelerated gradient descent method cannot be derived on the basis of [13]. Moreover, there is a constraint on the maximum size of the step, which somewhat simplifies the analysis of the algorithms proposed in [13].

Unfortunately, we failed to obtain theoretical estimates for the proposed method. In this paper, we describe experiments (see Section 4) that show that the proposed algorithm has a higher convergence rate than Adagrad [10] and Adam [16] in machine learning problems with logistic regression, neural network and convolutional neural network. In addition, in Section 3, we give a justification of the proposed algo-

rithm and discuss some details that prevented us from rigorously proving the method convergence in the optimal form.

2. FAST ADAPTIVE STOCHASTIC GRADIENT DESCENT METHOD

First, we give the general formulation of the convex optimization problem [1]. Let a function $f(x): Q \to \mathbb{R}$ and an arbitrary norm $\|\cdot\|$ in $\mathbb{R}^n$ be given. The adjoint norm is defined by

$$\|\lambda\|_* \overset{\text{def}}{=} \max_{\|v\|\le 1; v\in\mathbb{R}^n} \langle \lambda, v\rangle \quad \forall \lambda \in \mathbb{R}^n.$$

We assume that

1. $Q \subseteq \mathbb{R}^n$ is a convex closed set;
2. $f(x)$ is a continuous convex function on Q;
3. $f(x)$ is bounded from below on Q and attains its minimum at a certain point (not necessarily unique) $x_* \in Q$.

Consider the optimization problem

$$f(x) \to \min_{x\in Q}.$$

Let us introduce the concepts of prox-function and Bregman divergence [17].

Definition 1. $d(x)$: $Q \to \mathbb{R}$ is called a prox-function if $d(x)$ is continuously differentiable on $\operatorname{int} Q$ and $d(x)$ is 1-strongly convex with respect to the norm $\|\cdot\|$ on $\operatorname{int} Q$.

Definition 2. The Bregman divergence is

$$V(x,y) \overset{\text{def}}{=} d(x) - d(y) - \langle \nabla d(y), x - y\rangle,$$

where $d(x)$ is an arbitrary prox-function.

Definition 3. A stochastic (δ, L)-oracle is an oracle that, for the given point $y \in Q$, produces a pair $(f_\delta(y), \nabla f_\delta(y;\xi))$ such that

$$0 \le f(x) - f_\delta(y) - \langle \nabla f_\delta(y), x - y\rangle \le \frac{L}{2}\|x - y\|^2 + \delta \quad \forall x \in Q,$$

$$\mathbb{E}[\nabla f_\delta(y;\xi)] = \nabla f_\delta(y) \quad \forall y \in Q,$$

$$\mathbb{E}\left[\exp\left(\frac{\|\nabla f_\delta(y;\xi) - \nabla f_\delta(y)\|_*^2}{\sigma^2}\right)\right] \le \exp(1) \quad \forall y \in Q,$$

where $\sigma^2 > 0$ and ξ is an arbitrary random variable.

Note that the concept of a stochastic (δ, L)-oracle is based on the concept of the (δ, L)-oracle [18–20].

Remark 1. In Definition 3, we can additionally require that not only the gradient but also the function itself are random; i.e., we can require that the stochastic (δ, L)-oracle returns a random value $f_\delta(y;\xi)$ instead of the nonrandom $f_\delta(y)$.

Define the constant R_Q such that

$$R_Q \ge \max_{x,y\in Q}\|x - y\|.$$

We assume that $R_Q < \infty$.

In the method proposed below, we will estimate the true gradient at each step by the quantity $\nabla f_\delta(y;\xi_j)$, $j \in [1 \dots m_{k+1}]$ using the mini-batch technique (e.g., see [21]).

Define

$$\tilde{\nabla}^{m_{k+1}} f_\delta(y) \overset{\text{def}}{=} \frac{1}{m_{k+1}} \sum_{j=1}^{m_{k+1}} \nabla f_\delta(y;\xi_j). \tag{1}$$

Let κ be the regularity constant defined in [22] for $(\mathbb{R}^n, \|\cdot\|_*)$. For certain simple cases, it holds that if $\|\cdot\|_* = \|\cdot\|_2$, then $\kappa = 1$. Moreover, if $\|\cdot\|_* = \|\cdot\|_q$ for $q \in [2,\infty]$, then $\kappa = \min[q-1, 2\ln(n)]$. Introduce the notation $\widetilde{\Omega} \overset{\text{def}}{=} 2\kappa + 4\Omega\sqrt{\kappa} + 2\Omega^2$.

In the proposed algorithm, we will use the constant $L_0 \geq 0$, which has the sense of supposedly local Lipschitz constant of the gradient at the point x_0. Consider the adaptive mirror version of the method of similar triangles with the stochastic (a, L)-oracle.

Algorithm 1 (Theoretical version)

Input: an initial point x_0, the desired solution accuracy ϵ, the constants δ and L in the (δ, L)-oracle, the confidence level β, L_0 ($L_0 \leq L$), and the constant σ^2 in Definition 3.

Algorithm: Set

$$N := \left\lceil \frac{2\sqrt{3}\sqrt{L}R_Q}{\sqrt{\epsilon}} \right\rceil, \quad \Omega := \sqrt{6\ln\frac{N}{\beta}}.$$

Step 0:

$$y_0 := x_0, \quad u_0 := x_0, \quad \alpha_0 := 0, \quad A_0 := \alpha_0.$$

Step k + 1:

$$j_{k+1} := 0.$$

Until the inequality

$$f_\delta(x_{k+1}) \leq f_\delta(y_{k+1}) + \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}), x_{k+1} - y_{k+1} \right\rangle + \frac{L_{k+1}}{2}\|x_{k+1} - y_{k+1}\|^2 + \frac{3\sigma^2\widetilde{\Omega}}{L_{k+1}m_{k+1}} + \delta \tag{2}$$

becomes true, repeat

$$L_{k+1} := 2^{j_{k+1}-1}L_k, \quad \alpha_{k+1} := \frac{1+\sqrt{1+4A_kL_{k+1}}}{2L_{k+1}}, \quad A_{k+1} := A_k + \alpha_{k+1},$$
$$y_{k+1} := \frac{\alpha_{k+1}u_k + A_kx_k}{A_{k+1}}, \quad m_{k+1} := \left\lceil \frac{3\sigma^2\tilde{\Omega}\alpha_{k+1}}{\epsilon} \right\rceil.$$

Generate independent identically distributed (i.i.d.) random variables ξ_j ($j = 1,\ldots,m_{k+1}$) and calculate $\tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1})$.

$$\phi_{k+1}(x) \overset{\text{def}}{=} V(x,u_k) + \alpha_{k+1}\left(f_\delta(y_{k+1}) + \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}), x - y_{k+1}\right\rangle\right)$$
$$u_{k+1} := \operatorname*{argmin}_{x\in Q} \phi_{k+1}(x), \quad x_{k+1} := \frac{\alpha_{k+1}u_{k+1} + A_kx_k}{A_{k+1}}, \quad j_{k+1} := j_{k+1} + 1.$$

End of algorithm.

Hypothesis 1. *Let x_* be the closest minimizer to the point x_0 in the sense of the Bregman divergence and the accuracy be $\delta = \mathbb{O}(\epsilon^{3/2})$. Then, for the proposed algorithm, the following equality holds with the probability $1 - \mathbb{O}(\beta)$:*

$$f(x_N) - f(x_*) = \mathbb{O}(\epsilon).$$

3. JUSTIFICATION OF THE ALGORITHM

Here, we give a justification of this algorithm. The algorithm is based on the gradient descent methods described in [20, 23, 5]. For the fast **adaptive** gradient descent method with the (δ, L)-oracle, the following convergence estimate can be obtained (see [20, 23, 5]):

$$f(x_N) - f(x_*) \le \mathbb{O}\left(\frac{LR_Q^2}{N^2} + N\delta\right).$$

For the fast **stochastic** gradient descent method with the (δ, L)-oracle, the following estimate was obtained in [20]:

$$f(x_N) - f(x_*) \le \mathbb{O}\left(\frac{LR_Q^2}{N^2} + \frac{\sigma R_Q}{\sqrt{N}}\right).$$

Therefore, the expected estimate for the fast **adaptive stochastic** gradient descent method with the (δ, L)-oracle should be

$$f(x_N) - f(x_*) \le \mathbb{O}\left(\frac{LR_Q^2}{N^2} + N\delta + \frac{\sigma R_Q}{\sqrt{N}}\right).$$

Currently, the main difficulty in proving Hypothesis 1 is related to "rejection sampling" [24]. In the proofs of optimization methods with stochastic gradient (see (1)), the key point is the fact that the stochastic gradient has an unbiased expectation relative to the true gradient. However, this is not the case for the algorithm presented in Section 2. Note that the proposed method is adaptive and is adjusted to local smoothness by checking inequality (2). Thus, when we select the step in the inner loop, we first generate a mini-batch, and then check to see if this mini-batch is adequate by verifying inequality (2). Thus, we implicitly perform the rejection sampling [24]; therefore, the generated stochastic gradient is biased.

We made various attempts to revise the algorithm, but we failed to resolve the rejection sampling problem.

In Algorithm 1 in Section 2, we each time generate a mini-batch in the procedure of selecting the parameter L_{k+1}. A possible way of resolving the rejection sampling problem is to generate the mini-batch only once before selecting the parameter L_{k+1} (see Algorithm 2 in Section 4). Unfortunately, this trick causes another difficulty. As in [20], in the course of proof the expression

$$\sum_{k=0}^{N-1} \alpha_{k+1} \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}) - \nabla f_\delta(y_{k+1}), x - u_k \right\rangle$$

appears. The main problem is to find good upper bounds on this sum. In [20], such bounds were obtained, and it was proved that the following inequality holds with a "high probability":

$$\sum_{k=0}^{N-1} \alpha_{k+1} \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}) - \nabla f_\delta(y_{k+1}), x - u_k \right\rangle \le \mathbb{O}\left(\frac{\sigma R_Q}{\sqrt{N}}\right).$$

This inequality was proved using Azuma–Hoeffding type inequalities ([20, Lemma 7.11], [25]) and the fact that α_{k+1} are not random and the conditional expectation of the mini-batch equals the true gradient. For the case when we fix the mini-batch only one time, we have random α_{k+1} that correlate with the mini-batch; therefore, the standard Azuma–Hoeffding inequalities [20, Lemma 7.11] cannot be used.

In the next section, we perform numerical experiments by solving various optimization problems, and we show that the practical version of the proposed algorithm (see Algorithm 2 in Section 4) performs better than the popular algorithms Adagrad [10] and Adam [16].

4. NUMERICAL EXPERIMENTS

First, we describe the differences between the theoretical (Algorithm 1) and practical (Algorithm 2) versions.

1. In practice, we do not always know the constant L, and the constant R_Q can be unbounded; therefore, we are unable to exactly and a priori determine N from Algorithm 1. For this reason, we assume in the practical version that $\tilde{\Omega} = 1$.

2. In the practical version, the constant m_{k+1} is estimated using L_k rather than the constant L_{k+1}. This enables us to select the parameter L_{k+1} more efficiently because the number m_{k+1} of terms in the calculation of the mini-batch does not change.

3. For many classes of functions to be optimized, which include logistic regression, neural networks, and convolutional neural networks, the complexity of function evaluation has the same order of complexity as the evaluation of the gradient [26]. Hence, we also evaluate the function value using the mini-batch technique (see Remark 1):

$$f_\delta^{m_{k+1}}(y) \stackrel{\text{def}}{=} \frac{1}{m_{k+1}} \sum_{j=1}^{m_{k+1}} \nabla f_\delta(y; \xi_j).$$

The theoretical considerations in the preceding sections suggest Algorithm 2, which is the practical version of Algorithm 1. We conducted three series of experiments. All the functions were optimized in the Euclidean norm $\|\cdot\|_2$ with $V(x, y) = \frac{1}{2}\|x - y\|_2^2$, and $Q = \mathbb{R}^n$. The initial parameters of Algorithm 2 were identical for all experiments: the upper bound on the gradient variance $\sigma_0^2 = 0.1$, the accuracy $\epsilon = 0.002$, and the constant $L_0 = 1$. In the experiments, we used the following sets of data.

1. MNIST [27] consists of images of handwritten digits of size 28×28 pixels.

2. CIFAR [28] consists of color images of size $3 \times 32 \times 32$ pixels each which belongs to one of ten classes.

In the first two series of experiments, we used the set MNIST. In the experiments, we trained the logistic regression (a linear classifier with a convex loss function) and a two-layer neural network with the size of the hidden layer equal to 1000 (a nonlinear classifier with a nonconvex loss function). In the case of logistic regression, the number of optimized parameters was 7850, and in the case of the two-layer neural network it was 795010. The optimization was performed using the proposed algorithm and the most popular adaptive algorithms Adam [16] and AdaGrad [10]. For Adagrad and Adam, the standard parameters were used: the batch size 128 and $lr = 0.001$. The comparison was performed in terms of the rate of the loss function variation as a function of the training iteration index and in terms of the classification accuracy on the test sample. The comparison results for the logistic regression are shown in Figs. 1a and 1b, and for the neural network in Figs. 1c and 1d. In both cases, the proposed method outperforms the methods mentioned above both in the rate of convergence of the loss function and in accuracy.

In the third series of experiments, a small convolutional neural network learned to predict the class of image on the data set CIFAR. The following convolutional neural network architecture was used.

1. The image matrix of size $3 \times 32 \times 32$ is scanned by the convolution window of size $3 \times 6 \times 5$. The convolution weights are adjusted parameters.

2. After the convolution, the resulting image is scanned by the so-called MaxPooling—the window of size 2×2 that leaves the maximum element within it and sets the other elements to zero.

3. Next, the result of the preceding step is scanned by the convolution window of size $6 \times 16 \times 5$.

4. Next, there are three sequential completely connected layers with the decreasing size 400, 120, and 84.

The total number of parameters in this network is 62000. In this series of experiments, the results produced by the proposed algorithm were also compared with the results produced by Adagrad and Adam (Figs. 1e and 1f). Again, Algorithm2 demonstrated better results.

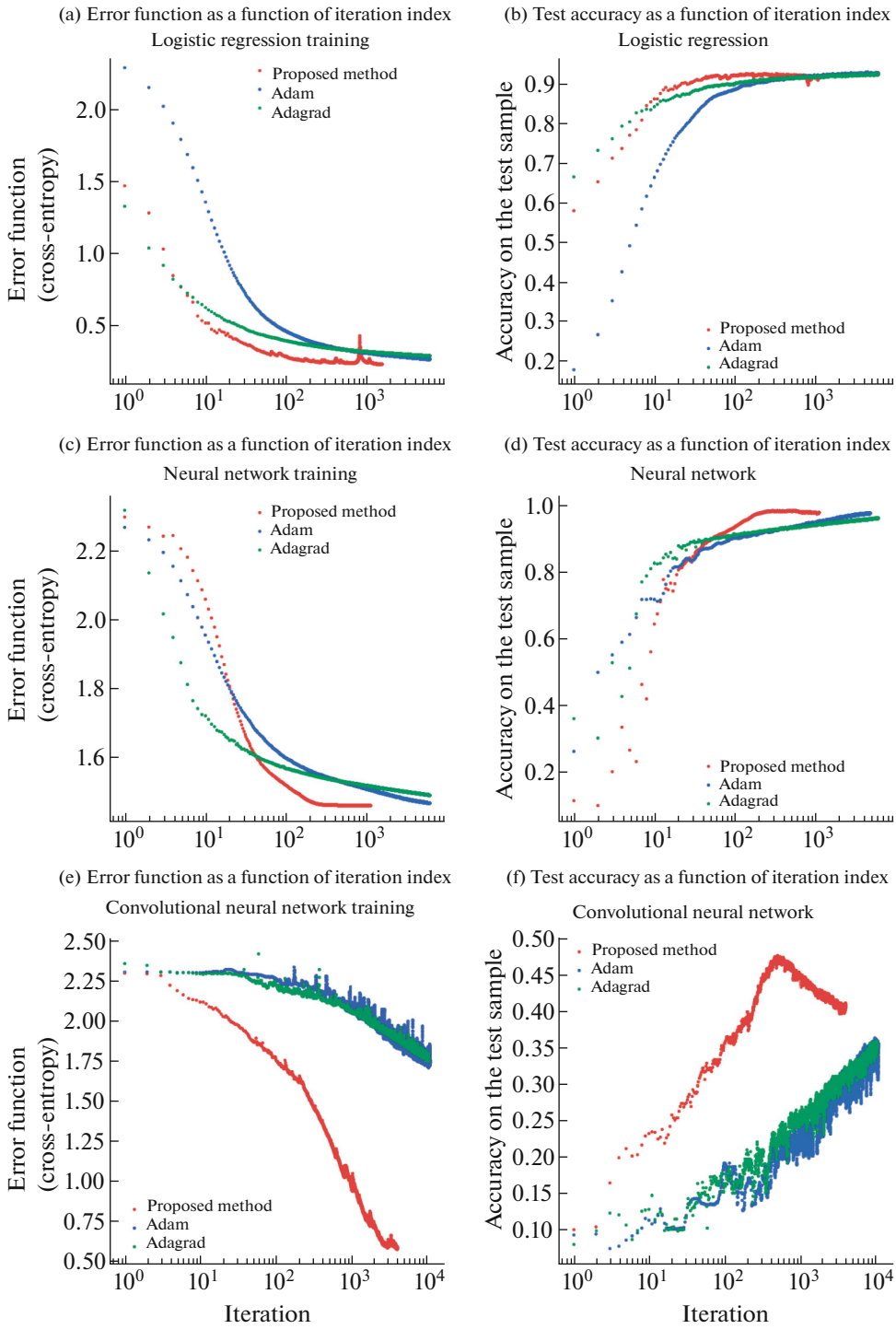

Fig. 1. Numerical experiments.

Algorithm 2 (Practical version)

Input: an initial point x_0, the constants $\epsilon > 0$, $L_0 > 0$, and $\sigma_0^2 > 0$.

Algorithm:

Step 0:

$$y_0 := x_0, \quad u_0 := x_0, \quad \alpha_0 := 0, \quad A_0 := \alpha_0.$$

Step k + 1:

$$\tilde{\alpha}_{k+1} = \frac{1+\sqrt{1+4A_kL_k}}{2L_k}, \quad m_{k+1} := \left\lceil \frac{3\sigma_0^2\tilde{\alpha}_{k+1}}{\epsilon} \right\rceil, \quad j_{k+1} := 0,$$

generate i.i.d. random variables ξ_j $(j = 1,\ldots,m_{k+1})$.

Until the inequality

$$f_\delta^{m_{k+1}}(x_{k+1}) \le f_\delta^{m_{k+1}}(y_{k+1}) + \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}), x_{k+1} - y_{k+1} \right\rangle + \frac{L_{k+1}}{2}\|x_{k+1} - y_{k+1}\|^2 + \frac{\epsilon}{L_{k+1}\alpha_{k+1}} \tag{3}$$

becomes true, repeat

$$L_{k+1} := 2^{j_{k+1}-1}L_k, \quad \alpha_{k+1} := \frac{1+\sqrt{1+4A_kL_{k+1}}}{2L_{k+1}}, \quad A_{k+1} := A_k + \alpha_{k+1},$$
$$y_{k+1} := \frac{\alpha_{k+1}u_k + A_kx_k}{A_{k+1}}.$$

Calculate $\tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1})$ and $f_\delta^{m_{k+1}}(y_{k+1})$.

$$\phi_{k+1}(x) \stackrel{\text{def}}{=} V(x,u_k) + \alpha_{k+1}\left(f_\delta^{m_{k+1}}(y_{k+1}) + \left\langle \tilde{\nabla}^{m_{k+1}} f_\delta(y_{k+1}), x - y_{k+1} \right\rangle\right)$$
$$u_{k+1} := \operatorname*{argmin}_{x\in Q} \phi_{k+1}(x), \quad x_{k+1} := \frac{\alpha_{k+1}u_{k+1} + A_kx_k}{A_{k+1}}, \quad j_{k+1} := j_{k+1} + 1.$$

Calculate $f_\delta^{m_{k+1}}(x_{k+1})$.

End of the algorithm.

FUNDING

The work by A. I. Tyurin was supported by the Russian Foundation for Basic Research, project no. 19-31-90062 Aspiranty.

REFERENCES

1. Yu. E. Nesterov, *Introduction to Convex Optimization* (Mosk. Tsentr Nepreryvnogo Matematicheskogo Obrazovaniya, Moscow, 2010) [in Russian].
2. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning (MIT Press, Cambridge, Mass., 2016).
3. A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems* (2012), pp. 1097–1105.
4. A. V. Gasnikov, P. E. Dvurechensky, and I. N. Usmanova, "On the nontriviality of fast randomized methods," Trudy Mosk. Fiz.-Teknh. Inst. **8** (2), 67–100 (2016).
5. A. V. Gasnikov, *Modern Numerical Optimization Methods: The Universal Gradient Descent Method* (Mosc. Fiziko-tekhnicheskii Institut, 2018). arXiv:1711.00394
6. F. Bach and K. Y. Levy, "A universal algorithm for variational inequalities adaptive to smoothness and noise," COLT, 2019.
7. S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, and S. Lacoste-Julien, "Painless stochastic gradient: Interpolation, line-search, and convergence rates," NIPS, 2019.
8. J. Nocedal and S. Wright, *Numerical Optimization* (Springer Science, 2006).
9. R. Ward, X. Wu, and L. Bottou, "AdaGrad stepsizes: Sharp convergence over nonconvex landscapes, from any initialization," ICML, 2019.
10. J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," J. Mach. Learning Res. **12**, 2121–2159 (2011).
11. Q. Deng, Y. Cheng, and G. Lan, "Optimal adaptive and accelerated stochastic gradient descent," 2018. arXiv:1810.00553.
12. K. Y. Levy, A. Yurtsever, and V. Cevher, "Online adaptive methods, universality and acceleration," NIPS, 2018.
13. A. N. Iusem, A. Jofre, R. I. Oliveira, and P. Thompson, "Variance-based extragradient methods with line search for stochastic variational inequalities," SIAM J. Optim. **29**, 175–206 (2019).

14. S. Boucheron, G. Lugosi, and P. Massart, *Concentration Inequalities: A Nonasymptotic Theory Of Independence* (Oxford Univ. Press, 2013).
15. D. Panchenko, "Symmetrization approach to concentration inequalities for empirical processes," Annals Probab. **31**, 2068–2081 (2003).
16. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," ICLR, 2015.
17. M. D. Gupta and T. Huang, "Bregman distance to l1 regularized logistic regression," in *19th International Conference on Pattern Recognition,* 2008, pp. 1–4.
18. O. Devolder, F. Glineur, and Yu. Nesterov, "First-order methods with inexact oracle: The strongly convex case," CORE Discussion Paper 2013/16. 2013.
https://www.uclouvain.be/cps/ucl/doc/core/documents/coredp2013_16web.pdf
19. O. Devolder, F. Glineur, and Yu. Nesterov, "First-order methods of smooth convex optimization with inexact oracle," Math. Program. **146**, 37–75 (2014).
20. O. Devolder, "Exactness, inexactness and stochasticity in first-order methods for largescale convex optimization," PhD thesis, CORE UCL, 2013.
21. M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," ACM, 2014.
22. A. Juditsky and A. Nemirovski, "Large deviations of vector-valued martingales in 2-smooth normed spaces," 2008. arXiv:0809.0813
23. A. V. Gasnikov and A. I. Tyurin, "Fast gradient descent for convex minimization problems with an oracle producing a $(\delta; L)$-model of function at the requested point," Comput. Math. Math. Phys. **59**, 1085–1097 (2019).
24. C. Robert and G. Casella, *Monte Carlo Statistical Methods* (Springer Science, 2013).
25. G. Lan, A. Nemirovski, and A. Shapiro, "Validation analysis of mirror descent stochastic approximation method," Math. Program. **134**, 425–458 (2012).
26. A. Griewank, "On automatic differentiation," Math. Program: Recent Developments Appl. **6** (6), 83–107 (1989).
27. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE **86**, 2278–2324 (1998).
28. A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images,* PhD thesis, University of Toronto, 2009.

Translated by A. Klimontovich