OPT2021: 13th Annual Workshop on Optimization for Machine Learning

High Probability Step Size Lower Bound for Adaptive Stochastic Optimization

Billy Jin BZJ3@CORNELL.EDU
Katya Scheinberg KATYAS@CORNELL.EDU
Miaolan Xie MX229@CORNELL.EDU
Cornell University, USA

Abstract

Several classical adaptive optimization algorithms, such as line search and trust region methods, have been recently extended to stochastic settings. Unlike the stochastic gradient method and its many variants, these algorithms do not use a pre-specified sequence of step sizes, but increase or decrease the step size adaptively according to the estimated progress of the algorithm. These algorithms rely on stochastic oracles that estimate function values, gradients, and Hessians in some cases. The accuracy requirement of these oracles is also adaptive and depends on the step size. In the deterministic setting, a lower bound on the step size is easily derived, however, in the stochastic setting, due to possible oracle failures, bounds on the step size have not been previously derived. In this paper we give a lower bound on the step size that holds with high probability. This bound is dependent on the probability of the oracle failures, recovering the deterministic result as an extreme case when this probability is zero.

1. Introduction

Widespread use of stochastic optimization algorithms for problems arising in machine learning and signal processing have made stochastic gradient method and its variants become overwhelmingly popular despite their theoretical and practical shortcomings. *Adaptive stochastic optimization* algorithms, on the other hand, borrow from decades of advances in deterministic optimization research, and offer new paths forward for stochastic optimization to be more effective and even more applicable. Adaptive algorithms can avoid many of the practical deficiencies of contemporary methods (such as the tremendous costs of tuning the step sizes of an algorithm for each individual application) while possessing strong convergence and worst-case complexity guarantees in incredibly diverse settings. In this work, we resolve one of the open problems underlying a class of adaptive stochastic optimization algorithms: Provide a high probability lower bound on the step size parameter.

We consider a class of adaptive stochastic algorithms for continuous optimization problems of the form

$$\min_{x\in\mathbb{R}^d} \phi(x), \tag{1}$$

where ϕ is possibly non-convex. We make the standard assumption of that ϕ is (twice-)continuously differentiable with Lipschitz continuous derivatives, but knowledge of Lipschitz constants is not

assumed by the algorithms. The goal of these algorithms is to find an ε-optimal point x_ε for ϕ, for which $\phi(x_\varepsilon) - \inf_x \phi(x) \leq \varepsilon$ if ϕ is convex and $\|\nabla\phi(x_\varepsilon)\| \leq \varepsilon$ if ϕ is nonconvex. We consider a setting where neither the function value $\phi(x)$, nor the gradient $\nabla\phi(x)$ are directly computable. Instead, given any $x \in \mathbb{R}^d$, it is assumed that stochastic approximations of $\phi(x)$, $\nabla\phi(x)$, and possibly $\nabla^2\phi(x)$ can be computed, although these approximations may possess different levels of *accuracy* and *reliability* depending on the particular setting of interest.

We now introduce the algorithmic framework for adaptive stochastic optimization in Algorithm 1. Several recent algorithms fall under this framework: for example, the adaptive line search algorithms in [3], [6], [1] and [5]; the adaptive trust region methods in [2] and [4], and the adaptive cubic regularization method in [3].

The framework is iterative, where in each iteration a stochastic model $m_k : \mathbb{R}^d \to \mathbb{R}$ of the objective function is constructed based on function, gradient and (possibly) Hessian approximations obtained via stochastic oracles. We refer to the zeroth-order approximation as $f(x, \xi_0)$, the first-order approximation as $g(x, \xi_1)$, and the second-order approximation as $H(x, \xi_2)$. A jth-order oracle is defined over a set of valid inputs $\mathcal{S}_j$. In the definition below, $\varphi(x, \xi_j)$ is used to denote the estimate computed by a general jth-order oracle.

Stochastic jth-order oracle over $\mathcal{S}_j$: Given a pair of real numbers $(M_j, \delta_j) \in \mathcal{S}_j$ and $x \in \mathbb{R}^d$, the oracle computes $\varphi(x, \xi_j)$ satisfying

$$\mathbb{P}_{\xi_j}[\|\varphi(x, \xi_j) - \nabla^j\phi(x)\| \leq M_j] \geq 1 - \delta_j,$$

Here M_j is the *accuracy*, and $1 - \delta_j$ is the *reliability*. ξ_j is a random variable whose distribution depends on (M_j, δ_j) and x, and $\mathbb{P}_{\xi_j}$ denotes the probability w.r.t. that distribution.

Algorithm 1: Algorithmic Framework for Adaptive Stochastic Optimization

0. Initialization

Choose $\eta \in (0, 1)$, $\gamma \in (0, 1)$, $\alpha_{\max} \in (0, \infty)$, $x_0 \in \mathbb{R}^d$, and $\alpha_0 \in (0, \alpha_{\max}]$. Set $k \leftarrow 0$.

1. Determine model and compute step

Construct a stochastic model m_k of ϕ at x_k using $f(x_k, \xi_{0,k})$, $g(x_k, \xi_{1,k})$, and (optionally) $H(x_k, \xi_{2,k})$ from *probabilistic zeroth-, first-, and (optionally) second-order oracles*. Compute $s_k(\alpha_k)$ such that the model reduction $m_k(x_k) - m_k(x_k + s_k(\alpha_k)) \geq 0$ is sufficiently large.

2. Check for sufficient reduction

Set $x_k^+ \leftarrow x_k + s_k(\alpha_k)$ and compute $f(x_k^+, \xi_{0,k}^+)$ as a stochastic approximation of $\phi(x_k^+)$ using a *probabilistic zeroth-order oracle*. Check if $f(x_k, \xi_{0,k}) - f(x_k^+, \xi_{0,k}^+)$ is sufficiently large (e.g., relative to the model reduction $m_k(x_k) - m_k(x_k^+)$) using a condition parameterized by η.

3. Successful iteration

If sufficient reduction has been attained (along with other potential requirements), then set $x_{k+1} \leftarrow x_k^+$ and $\alpha_{k+1} \leftarrow \min\{\gamma^{-1}\alpha_k, \alpha_{\max}\}$.

4. Unsuccessful iteration

Otherwise, set $x_{k+1} \leftarrow x_k$ and $\alpha_{k+1} \leftarrow \gamma\alpha_k$.

5. Next iteration

Set $k \leftarrow k + 1$ and go to Step 1.

Using the model m_k, a step $s_k(\alpha_k)$ that improves the value of the model m_k is computed, where α_k represents a step size parameter that influences the length of the step. The trial point $x_k^+ \leftarrow x_k + s_k(\alpha_k)$ is evaluated using the estimate $f(x_k^+, \xi_{0,k}^+)$. If this estimate suggests that sufficient improvement is attained, then the step is deemed *successful*, x_k^+ is accepted as the next iterate, and the step size parameter is increased up to a multiplicative factor; otherwise, the step is deemed *unsuccessful*, the iterate does not change, and α_k is decreased by a multiplicative factor. Unlike in the deterministic case, new calls to all oracles are made in the next iteration *even when the iterate does not change*.

For all prior adaptive algorithms in the literature [1–6], convergence and expected complexity are derived based on specific requirements of the stochastic oracles. In particular, δ is assumed to be sufficiently small, but fixed (often $\delta_j < \frac{1}{2}$ is sufficient), while M_j is chosen differently at each iteration, dependent on α_k. For example, in the first order trust region framework, $M_0 = \kappa\alpha_k^2$ and $M_1 = \kappa\alpha_k$, for some fixed constant κ. In many settings (e.g. expected risk minimization), the oracle works by computing a sample average over a batch of data points, and it can satisfy different accuracy requirements by using an appropriately sized batch of samples; in this case, ξ_j represents the randomness in the selection of the batch.

Under appropriate conditions on the oracles, previous works derive the expected iteration complexities of the adaptive algorithms, which are comparable to the iteration complexities of the respective deterministic algorithms. In the deterministic case, one of the key steps in the analysis is deriving a lower bound on α_k, which in turn guarantees consistent progress of the algorithm. In the stochastic case, a lower bound on α_k that holds with probability 1 does not exist, but it can be shown that there exists a *threshold* value $\bar{\alpha}$ for α_k such that once the step size falls below $\bar{\alpha}$, it tends to increase. This is sufficient to derive the expected iteration complexity. However, in the stochastic setting, there remains the question of providing a lower bound on α_k that holds *with high probability*. This question is important because the number of samples needed to compute an oracle value is dictated by the accuracy M_j required for the oracle, which in most settings is proportional to α_k, the step size parameter of the algorithm. The smaller the step size parameter, the more samples are needed for one oracle computation. Providing a lower bound on the step size parameter in turn gives an upper bound on the number of samples needed for each oracle evaluation. The goal of this paper is to derive such a lower bound under conditions which are satisfied by the existing algorithms cited above.

2. Conditions

We consider Algorithm 1 based on stochastic oracles, applied to problem (1). This algorithm generates a stochastic process (with respect to the randomness underlying the stochastic oracles), for which we define a stopping time.

Definition 1 (Stopping time) *For $\varepsilon > 0$, let T_ε be the first time such that a specified condition is satisfied. For example: $T_\varepsilon = \min\{k : \|\nabla\phi(x_k)\| \leq \varepsilon\}$ if ϕ is nonconvex, or $T_\varepsilon = \min\{k : \phi(x_k) - \inf_x \phi(x) \leq \varepsilon\}$ if ϕ is convex. We will refer to T_ε as the* stopping time *of the algorithm.*

We also say that iteration k is a *true* iteration if all oracles used on that iteration deliver the required accuracy. The following properties are assumed to hold for Algorithm 1.

Assumption 1 (Properties of the stochastic process generated by the adaptive stochastic algorithm) *The following hold for all iterations $k < T_\varepsilon$:*

(i) The probability that iteration k is true, conditioned on the past, is at least p, for some $p > \frac{1}{2}$.

(ii) There exists a constant $\bar{\alpha} > 0$, such that if $\alpha_k \leq \bar{\alpha}$ and iteration k is true, then it is successful.

The algorithms in [1–6] satisfy Assumption 1 with various stochastic oracles.

In the next section, under Assumption 1, we derive a high probability lower bound on α_k as a function of the number of iterations n ($< T_\varepsilon$), $\bar{\alpha}$, p, and γ.

3. High probability lower bound on the minimum step size

Theorem 1 *Let Assumption 1 hold for Algorithm 1, then for any $n < T_\varepsilon$, $\nu > 0$, with probability at least $1 - n^{-\nu} - cn^{-\frac{1}{2}(1+\nu)}$, we have*

$$\min_{1\leq k\leq n} \alpha_k \geq \bar{\alpha}\gamma\gamma^{(1+\nu)\log_{\frac{p}{1-p}} n} = \bar{\alpha}\gamma n^{-(1+\nu)\log_{\frac{p}{1-p}} 1/\gamma},$$

where $c = \frac{2p\sqrt{1-p}}{\sqrt{p}(1-2\sqrt{p(1-p)})^2}$.

The proof of this theorem involves two steps. First, in Subsection 3.1, we show that for $n < T_\varepsilon$, the step sizes of the algorithm can be coupled with a random walk on the non-negative integers. This reduces the problem to that of bounding the maximum value of a one-sided random walk in the first n steps. We then derive a high probability upper bound on this maximum value.

The following are some implications of the theorem.

Remark 2 *If we choose $\nu = 1$, then the minimum step size is lower bounded by $\bar{\alpha}\gamma n^{-2\log_{\frac{p}{1-p}} \frac{1}{\gamma}}$ with probability at least $1 - \frac{1+c}{n}$. Here are some observations:*

1. *For fixed n, γ, and $\bar{\alpha}$, the lower bound is a function of p. It increases as p increases. Specifically, the exponent of n changes with p, and the exponent goes to 0 as p goes to 1. Hence as p goes to 1, this lower bound simplifies to $\bar{\alpha}\gamma$, which matches the lower bound in the deterministic case.*

2. *When p is close to 1 (i.e. when the stochastic oracles are highly reliable), this lower bound decreases slowly as a function of n, since the exponent of n is close to 0. Alternatively, when the stochastic oracles are not highly reliable, increasing the value of γ allows the algorithm to maintain slow decrease of the step size.*

3. *Enlarging γ as p decreases makes intuitive sense for the algorithm. When p is large, an unsuccessful step is more likely to be caused by the step size being too large rather than the failure of the oracles to deliver desired accuracy. On the other hand, when p is small, unsuccessful iterations are likely to occur even when the step size parameter is already small. Thus in the latter case, larger γ values help avoid erroneous rapid decrease of the step size parameter.*

4. *If we choose* $\gamma = \left(\frac{p}{1-p}\right)^{-\frac{1}{4}}$ *then the minimum step size is lower bounded by* $\bar{\alpha}\gamma n^{-\frac{1}{2}}$ *with high probability. This coincides with the typical choice of the step size decay schemes for the stochastic gradient method applied to non-convex functions.*

3.1. Reduction to Random Walk

Let $\{\mathcal{A}_k\}_{k=0}^{\infty}$ denote the random sequence of step size parameter values for Algorithm 1. Let us assume, w.l.o.g, that $\mathcal{A}_0 = \gamma^j \bar{\alpha}$, for some $j \leq 0$. (Recall here that $0 < \gamma < 1$.) Then we observe that $\mathcal{A}_k = \gamma^{Y_k}\bar{\alpha}$, where $\{Y_k\}_{k=0}^{\infty}$ is a random sequence of integers, with $Y_0 = j \leq 0$, which increases by one on every unsuccessful step, and decreases by one on every successful step. Moreover, by Assumption (1), whenever $Y_k \geq 0$, the probability that it decreases by one is at least p. We now couple $\{Y_k\}_{k=0}^{\infty}$ with a random walk $\{Z_k\}_{k=0}^{\infty}$ which stochastically dominates Y_k.

Consider the following one-sided random walk $\{Z_k\}_{k=0}^{\infty}$, defined on the non-negative integers, with a self-loop at 0.

$$Z_0 = 0 \qquad Z_{k+1} = \begin{cases} Z_k + 1, & \text{w.p. } 1-p, \\ Z_k - 1, & \text{w.p. } p, \text{ if } Z_k \geq 1, \\ 0, & \text{w.p. } p, \text{ if } Z_k = 0. \end{cases} \tag{2}$$

Lemma 3 *For all* $k \leq T_\epsilon$, Z_k *stochastically dominates* Y_k.

This lemma is proved by exhibiting a coupling between the steps of the algorithm and this one-sided random walk. The full proof is in Appendix A. The lemma shows that under a coupling of Z_k and Y_k, $\gamma^{Y_k}\bar{\alpha}$ is always lower bounded by $\gamma^{Z_k}\bar{\alpha}$ for any $k \leq T_\epsilon$. Hence, to lower-bound the smallest step size of the algorithm, it suffices to upper-bound the maximum value of the random walk.

3.2. Upper-bounding the maximum value of the random walk

Definition 4 *Let* $\mathcal{N}(\ell, n)$ *be the random variable that denotes the number of times* $Z_k = \ell$ *in the first* n *steps of the random walk.*

By definition of $\mathcal{N}(\ell, n)$, we have $\mathcal{N}(\ell, n) > 0$ if and only if state ℓ is visited in the first n steps of the random walk. The next proposition upper bounds the probability that $\mathcal{N}(\ell, n) > 0$.

Proposition 1 *Let* $q = 1 - p$. *We have*

$$\mathbb{P}(\mathcal{N}(\ell, n) > 0) \leq (n - \ell + 1)\frac{1 - (q/p)}{1 - (q/p)^{\ell+1}}\left(\frac{q}{p}\right)^{\ell} + \frac{2p}{(1 - 2\sqrt{pq})^2}\left(\frac{q}{p}\right)^{(\ell+1)/2}.$$

The proof of Proposition 1 is in Appendix B.

Remark 5 *The bound for Proposition 1 is essentially tight, as the decay of* $\mathbb{P}(\mathcal{N}(\ell, n) > 0)$ *is not faster than geometric;* q^ℓ *is a lower bound. Hence, the bound in Theorem 1 is essentially tight.*

With the above proposition at hand, Theorem 1 is proved by choosing an appropriate level ℓ, for which $\mathbb{P}(\mathcal{N}(\ell, n) = 0)$ is high. The full proof of Theorem 1 is in Appendix C.

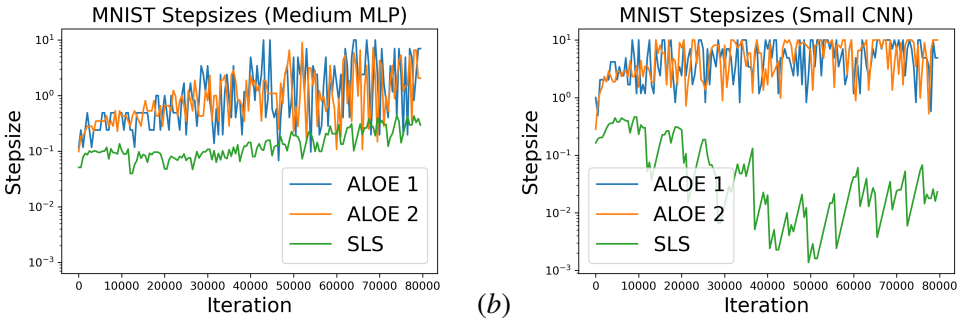
4. Numerical Illustration

In this section we conduct numerical experiments to illustrate how the step size parameter behaves in practice for an adaptive stochastic algorithm. One example algorithm that falls under the framework of Algorithm 1 is the "ALOE" algorithm in [5] (described in Appendix D). ALOE stands for Adaptive Line-search with Oracle Estimations. It is proposed by [1], and is an extension of the line search algorithm in [3] to the setting of inexact function estimates using a relaxed Armijo condition. It is shown in [1] that ALOE satisfies Assumption 1. In contrast, the SLS algorithm, introduced in [7], is a stochastic line search algorithm which does not satisfy Assumption 1.

We present experimental results for the step sizes of ALOE and SLS on two different neural network architectures, using softmax loss function, trained on MNIST Handwritten Digit Classification Dataset. The experiment settings are the same as those in [5]. The first architecture is a multi-layer perceptron (MLP) neural network that has four layers: an input layer with 784 nodes, two hidden layers with 512 and 256 nodes, and an output layer with 10 nodes. All activation functions are ReLU. This is the same architecture as in [7]. The second network is a small convolutional neural network (CNN) that in addition to the input and output layers, has two convolutional layers and one fully connected layer. Each convolutional layer uses a 3×3 kernel with a stride length of 1, and is followed by a 2×2 max pooling. The CNN architecture follows the tutorial at this link.

Figure 1 plots the step size parameters of the following algorithms, on the above two architectures. 1) ALOE 1 is the ALOE algorithm with $\gamma = 0.7$, 2) ALOE 2 is the ALOE algorithm with $\gamma = 0.9$. 3) SLS algorithm uses the suggested parameters as in the paper [7].

We see that the step sizes for the ALOE algorithms behave similarly to a random walk that is bounded below. On the other hand, the step sizes for SLS are smaller than those of ALOE. This is especially apparent on the CNN, where the step sizes for SLS get very small. This resulted in the SLS algorithm making less progress than ALOE on the CNN architecture with the same number of iterations. On the CNN, both ALOE algorithms had a final training loss of around 0.3, whereas SLS had a training loss of 1.4. Also, the final classification accuracies of the ALOE algorithms were above 99%, whereas that of SLS was around 49%. On the other hand, on the MLP network, both the ALOE algorithms and SLS had relatively large step sizes, and resulted in similar classification accuracies (around 98%).



(*a*) (*b*)

Figure 1: Step sizes of the ALOE and SLS algorithms, on the MNIST dataset for two NN architectures.

Acknowledgments

The authors wish to thank James Allen Fill for pointing out useful references and the proof of Proposition 1. This work was partially supported by NSF Grants TRIPODS 17-40796, NSF Grant CCF 2008434 and DARPA Lagrange award HR-001117S0039. Miaolan Xie was partially supported by a PhD Fellowship provided by MunchRe. Billy Jin was partially supported by NSERC fellowship PGSD3-532673-2019.

References

[1] Albert S Berahas, Liyuan Cao, and Katya Scheinberg. Global convergence rate analysis of a generic line search algorithm with noise. *arXiv preprint arXiv:1910.04055*, 2019.

[2] Jose Blanchet, Coralia Cartis, Matt Menickelly, and Katya Scheinberg. Convergence rate analysis of a stochastic trust-region method via supermartingales. *INFORMS journal on optimization*, 1(2):92–119, 2019.

[3] C. Cartis and K. Scheinberg. Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Mathematical Programming*, 169(2):337–375, 2017. doi: 10.1007/s10107-017-1137-4.

[4] Serge Gratton, Clément W Royer, Luís N Vicente, and Zaikun Zhang. Complexity and global rates of trust-region methods based on probabilistic models. *IMA Journal of Numerical Analysis*, 38(3):1579–1597, 2018.

[5] Billy Jin, Katya Scheinberg, and Miaolan Xie. High probability complexity bounds for line search based on stochastic oracles. *arXiv preprint arXiv:2106.06454*, 2021.

[6] Courtney Paquette and Katya Scheinberg. A stochastic line search method with expected complexity analysis. *SIAM Journal on Optimization*, 30(1):349–376, 2020.

[7] Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Appendix A. Proof of Lemma 3

Proof [of Lemma 3] Initially, $Z_0 = 0$ and $Y_0 \leq 0$. If $Y_k \leq -1$, we update Z_{k+1} from Z_k according to Equation (2), independently of how Y_k changes to Y_{k+1}. If $Y_k \geq 0$, then we first check if Y_k increased or decreased. Let p' be the probability that $Y_{k+1} = Y_k - 1$ on this sample path. Since $Y_k \geq 0$, we know that $p' \geq p$. Now, if $Y_{k+1} = Y_k + 1$, then we set $Z_{k+1} = Z_k + 1$. On the other hand, if $Y_{k+1} = Y_k - 1$, then we set $Z_{k+1} = Z_k + 1$ with probability $1 - \frac{p}{p'}$, and $Z_{k+1} = \max\{Z_k - 1, 0\}$ with probability $\frac{p}{p'}$. Note that these probabilities are well-defined because $p' \geq p$.

Observe that under this coupling, $Z_k \geq Y_k$ on every sample path. Moreover, $\{Z_k\}$ and $\{Y_k\}$ have the correct marginal distributions. For Y_k, this is easy to see, since it evolves according to its true distribution and we are constructing Z_k from it. For Z_k, on any step with $Y_k \leq -1$, Z_{k+1} evolves from Z_k correctly according to Equation (2) by construction. On a step with $Y_k \geq 0$, the update to Z_{k+1} also follows Equation (2), since the probability that Z_k increases is $(1-p') + p'(1-\frac{p}{p'}) = 1-p$.

To summarize, we have exhibited a coupling between $\{Z_k\}$ and $\{Y_k\}$, under which $Z_k \geq Y_k$ on any sample path. Thus, $\mathcal{A}_k = \gamma^{Y_k}\bar{\alpha}$ the step size of the algorithm at iteration k, stochastically dominates $\gamma^{Z_k}\bar{\alpha}$. Thus, to obtain a lower bound on the step sizes of the algorithm, it suffices to obtain an upper bound on the random walk $\{Z_k\}$. ■

Appendix B. Proof of Proposition 1

Proof [of Proposition 1] Let $q = 1 - p$. Denote by $p_{\ell,N}$ the probability that state ℓ is reached in the first N steps. Note that $p_{\ell,N} = \mathbb{P}(\mathcal{N}(\ell, N) > 0)$ by definition. First, observe that $p_{\ell,N}$ remains unchanged if we change the state space from $\{0, 1, 2, \ldots\}$ to $\{0, 1, 2, \ldots, \ell\}$ and modify the chain to hold in state ℓ with probability q (instead of moving from ℓ to $\ell + 1$ with that probability). Call the modified transition kernel P. Noting that $P^m_{0,\ell}$ is the probability that $X_m = \ell$, we see that

$$p_{\ell,N} \leq \sum_{m=\ell}^{N} P^m_{0,\ell}. \tag{3}$$

The matrix P is explicitly diagonalized in [1, Section XVI.3]. In (3.16) there, take $n = m$ and $j = 0$ and $k = \ell$. The result is

$$P^m_{0,\ell} = \frac{1 - (q/p)}{1 - (q/p)^{\ell+1}} \left(\frac{q}{p}\right)^{\ell} - \frac{2q}{\ell+1} \left(\frac{q}{p}\right)^{\frac{\ell-1}{2}} \sum_{r=1}^{\ell} \frac{\left[\sin \frac{\pi r}{\ell+1}\right] \left[\sin \frac{\pi r \ell}{\ell+1}\right] \left[2\sqrt{pq} \cos \frac{\pi r}{\ell+1}\right]^m}{1 - 2\sqrt{pq} \cos \frac{\pi r}{\ell+1}} \tag{4}$$

The absolute value of the sum appearing in (4) can of course be bounded above by

$$\sum_{r=1}^{\ell} \frac{(2\sqrt{pq})^m}{1 - 2\sqrt{pq}} = \ell \frac{(2\sqrt{pq})^m}{1 - 2\sqrt{pq}}$$

and this readily yields

$$P^m_{0,\ell} \leq \frac{1 - (q/p)}{1 - (q/p)^{\ell+1}} \left(\frac{q}{p}\right)^{\ell} + \frac{2p}{1 - 2\sqrt{pq}} \left(\frac{q}{p}\right)^{\frac{\ell+1}{2}} (2\sqrt{pq})^m \tag{5}$$

Summing (5) over $m = \ell, \ldots, N$ and using (3) we obtain the bound on $p_{\ell,N}$ claimed in the proposition. ■

Appendix C. Proof of Theorem 1

Proof [of Theorem 1] Let $q = 1 - p$. By Proposition 1, we have:

$$\mathbb{P}(\mathcal{N}(k,n) > 0) \leq (n-k+1)\frac{1-(q/p)}{1-(q/p)^{k+1}}\left(\frac{q}{p}\right)^k + \frac{2p}{(1-2\sqrt{pq})^2}\left(\frac{q}{p}\right)^{(k+1)/2}.$$

Or in other words:

$$\begin{aligned}\mathbb{P}(\mathcal{N}(k,n) = 0) &\geq 1 - (n-k+1)\frac{1-(q/p)}{1-(q/p)^{k+1}}\left(\frac{q}{p}\right)^k - \frac{2p}{(1-2\sqrt{pq})^2}\left(\frac{q}{p}\right)^{(k+1)/2} \\ &\geq 1 - n\left(\frac{q}{p}\right)^k - \frac{2p}{(1-2\sqrt{pq})^2}\left(\frac{q}{p}\right)^{(k+1)/2}\end{aligned}$$

Let $a > 1$ be a parameter to be set later, and take $k = \lceil \log_a(n) \rceil$. Then the above inequality implies:

$$\mathbb{P}(\mathcal{N}(k,n) = 0) \geq 1 - n\left(\frac{q}{p}\right)^{\log_a(n)} - \frac{2p}{(1-2\sqrt{pq})^2}\left(\frac{q}{p}\right)^{(\log_a(n)+1)/2}.$$

Note that $\left(\frac{q}{p}\right)^{\log_a(n)} = n^{\log_a(\frac{q}{p})}$, hence

$$\mathbb{P}(\mathcal{N}(k,n) = 0) \geq 1 - n^{1-\log_a(\frac{p}{q})} - \frac{2p\sqrt{q}}{\sqrt{p}(1-2\sqrt{pq})^2}n^{-\frac{1}{2}\log_a(\frac{p}{q})}.$$

Note that when $a < \frac{p}{q}$, both $1 - \log_a(\frac{p}{q})$ and $-\frac{1}{2}\log_a(\frac{p}{q})$ are negative, so the event $\mathcal{N}(k,n) = 0$ holds "with high probability". Suppose $a = \left(\frac{p}{q}\right)^{1/(1+\nu)}$ for some $\nu > 0$. Then $\log_a \frac{p}{q} = 1 + \nu$, so that

$$\mathbb{P}(\mathcal{N}(k,n) = 0) \geq 1 - n^{-\nu} - \frac{2p\sqrt{q}}{\sqrt{p}(1-2\sqrt{pq})^2}n^{-\frac{1}{2}(1+\nu)}.$$

In other words, with probability at least $1 - n^{-\nu} - cn^{-\frac{1}{2}(1+\nu)}$ with $c = \frac{2p\sqrt{q}}{\sqrt{p}(1-2\sqrt{pq})^2}$, the algorithm will not reach any step size smaller or equal to $\gamma^{\lceil \log_a n \rceil}\bar{\alpha} = \gamma^{\lceil (1+\nu)\log_{p/q} n \rceil}\bar{\alpha}$ in the first n steps.

■

Appendix D. ALOE algorithm

Algorithm 2: ALOE (Adaptive Line-search with Oracle Estimations)

Input: Parameter ϵ_f of the zeroth order oracle, starting point x_0, max step size $\alpha_{\max} > 0$, initial step size $\alpha_0 < \alpha_{\max}$, constants $\theta, \gamma \in (0, 1)$.

for $k = 0, 1, 2, \ldots$ **do**

Compute gradient approximation g_k:

Generate the direction $g_k = g(x_k, \xi_k')$ using the probabilistic first order oracle, with $\alpha = \alpha_k$.

Check sufficient decrease:

Let $x_k^+ = x_k - \alpha_k g_k$. Generate $f(x_k, \xi_k)$ and $f(x_k^+, \xi_k^+)$ using the probabilistic zeroth order oracle. Check the modified *Armijo* condition:

$$f(x_k^+, \xi_k^+) \leq f(x_k, \xi_k) - \alpha_k \theta \|g_k\|^2 + 2\epsilon_f. \tag{6}$$

Successful step:

If (6) holds, then set $x_{k+1} \leftarrow x_k^+$ and $\alpha_{k+1} \leftarrow \min\{\alpha_{\max}, \gamma^{-1}\alpha_k\}$.

Unsuccessful step:

Otherwise, set $x_{k+1} \leftarrow x_k$ and $\alpha_{k+1} \leftarrow \gamma\alpha_k$.

end