Stochastic Newton with Arbitrary Sampling

Denis Shveykin **Rustem Islamov**

Abstract

The problem of minimizing the average of a large number of sufficiently smooth and strongly convex functions is ubiquitous in machine learning. Stochastic first-order methods for this problem of Stochastic Gradient Descent type are well studied. In turn, second-order methods, such as Newton, have certain advances since they can adapt to the curvature of the problem. They are also known for their fast convergence rates. But stochastic variants of Newton-type methods are not studied as good as SGD-type ones and have limitations on the batch size. Previously, a method was proposed which requires no limitations on batch sizes. Our work explores this method with different sampling strategies that lead to practical improvements.

Keywords Stochastic Newton, sampling strategy

1 Introduction

The problem is to minimize the empirical risk which has finite-sum structure [10]:

$$\min_{x\in\mathbb{R}^d}\left[f(x)\overset{def}{=}\frac{1}{n}\sum_{i=1}^{n}f_i(x)\right], \tag{1}$$

where each f_i is assumed to have Lipschitz Hessian.

Typically, n - the total number of functions - is very large for modern real-world problems. Thus, the stochastic approach is used because it is computationally difficult to evaluate the gradient of all f_i at each step. The Stochastic Gradient Descent (SGD) method [16] calculates the gradients of some randomly chosen f_i which leads to cheaper computation per-iteration cost compared to the vanilla Gradient Descent (GD). The analysis of SGD and its modifications is rich and well explored. Firstly, the theory of SGD-type methods does not restrict the batch size.

Therefore, these algorithms can be applied even with small batches. It is known that simple SGD converges only to a neighbourhood of the solution only [14, 4] whose size is proportional to the variance of the stochastic gradient. However, there are techniques (also known as variance reduction) to solve this issue. These techniques [17, 7, 2, 6] modify the update rule of vanilla SGD which allows to mitigate the aforementioned effect without changing per-iteration cost. However, the main disadvantage of all gradient-type methods is that a computation complexity depends on the curvature of the problem, which is called the condition number and is defined as the ratio of Lipschitzness and strong convexity parameters.

This is a place where second-order methods such as the Newton method [13, 8, 3] come to play. Taking into account second-order derivatives it possible to adjust the algorithm's step sizes to the curvature of the problem [13]. Unfortunately, much less work has been done in the direction of stochastic Newton-type methods. Many methods [9, 1, 22, 19, 18, 21] require large batch sizes. In particular, the required batch size is commonly quadratically proportional to the inverse of the desired accuracy. That means that one need to evaluate a large number of f_i Hessians which sometimes can be much larger than n. [10] proposes a simple Stochastic Newton algorithm, which can work with batches of any size. This algorithm achieves local linear and in come cases super-linear convergence rates.

In practice, various sampling strategies are used for SGD-type algorithms to improve further the performance. One of the most famous sampling mechanisms is so-called Importance Sampling [5, 20, 11]. The idea is to compute gradients of the functions that have more impact on the problem. Another method of random optimization is Random Reshuffling, which does iterative gradient steps passing reshuffled data [12]. [15] studies many other sampling mechanisms. We analyse such strategies, but for the Algorithms 1 of [10], to improve the theoretical and practical applications of the algorithms. We explore various sampling strategies supporting them with rigorously constructed experiments.

2 Problem statement

We consider classical empirical risk minimization (ERM) which typically arises in many machine learning problems to train a model. The objective function f (1) is an average of a large number of functions f_i, where f_i represents a loss on the i-th train data point. For example, this notation can be attributed to linear regression. In this case the goal is to find optimal model parameters x that minimize the mean squared error (MSE) on the train data.

2.1 Assumptions

We make standard assumptions on functions f_i; the same that were originally taken into account in [10].

Assumption 1 (Strong convexity). *A differentiable function $\phi : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex, where $\mu > 0$, if $\forall\ x, y \in \mathbb{R}^d$*

$$\phi(x) \geqslant \phi(y) + \langle \nabla\phi, x - y\rangle + \frac{\mu}{2}\|x - y\|^2, \tag{2}$$

where the norm $\|\cdot\|$ is Euclidean. For twice differentiable functions this assumption is equivalent to the Hessian having each eigenvalue $\geqslant \mu$.

Assumption 2 (Lipschitz Hessian). *A function $\phi : \mathbb{R}^d \to R$ has H-Lipschitz Hessian if $\forall\ x, y \in \mathbb{R}^d$*

$$\|\nabla^2\phi(x) - \nabla^2\phi(y)\| \leqslant H\|x - y\| \tag{3}$$

2.2 Sampling

Definition 1 (Sampling). A random set-valued mapping $\hat{S} : [n] \to 2^{[n]}$ is called sampling.

That means that each $S_k \subseteq [n]$ is a realization of $\hat{S}$. In this case we can call any probability distribution on $2^{[n]}$ a sampling strategy.

2.3 Algorithm

We apply different sampling strategies on top of the Algorithm 1 from [10]:

Algorithm 1 Stochastic Newton (SN)

Initialize: Choose starting iterates $w_1^0, w_2^0, ...w_n^0 \in \mathbb{R}^d$

for $k = 0, 1, 2, ...$ **do**

$$x^{k+1} = \left[\frac{1}{n}\sum_{i=1}^{n}\nabla^2 f_i(w_i^k)\right]^{-1}\left[\frac{1}{n}\sum_{i=1}^{n}\nabla^2 f_i(w_i^k)w_i^k - \nabla f_i(w_i^k)\right]$$

Choose a subset $S^k \subseteq \{1, 2, ..., n\}$ with one of the sampling strategies

$$w_i^{k+1} = \begin{cases} x^{k+1} & i \in S^k \\ w_i^k & i \notin S^k \end{cases}$$

end for

2.4 Goal of the project

We compare different sampling strategies for the Algorithm 1 proving convergence guarantees and showing practical improvements over the baseline strategy.

3 Theory

3.1 Proofs for the Algorithm

The convergence of the Algorithm 1 was clearly described and proven in [10]. Firstly, we look at the statements of three lemmas from the aforementioned article.

Lemma 1. *Let f_i be μ-strongly convex and have H-Lipschitz Hessian for all $i = 1, ..., n$ and consider the following Lyapunov function*

$$\mathcal{W}^k \overset{def}{=} \frac{1}{n} \sum_{i=1}^{n} \|w_i^k - x^*\|^2, \tag{4}$$

where x^* is the minimizer of the objective function f (1). Then the iterates of the Algorithm 1 satisfy

$$\|x^{k+1} - x^*\| \leqslant \frac{H}{2\mu} \mathcal{W}^k. \tag{5}$$

Lemma 2. *Assume that every f_i is μ-strongly convex and has H-Lipschitz Hessian. If $\|w_i^0 - x^*\| \leqslant \frac{\mu}{H}$ for $i = 1, ..., n$, then for all k*

$$\mathcal{W}^k \leqslant \frac{\mu^2}{H^2}. \tag{6}$$

The last lemma is the one that incorporates the nature of the sampling strategy. The authors of the Algorithm 1 used so called τ-nice sampling, which chooses a subset $S^k \subseteq \{1, 2, ..., n\}$ precisely of size τ uniformly at random. Therefore it is possible to determine the expectation $\mathbb{E}_k[\mathcal{W}^{k+1}] \overset{\text{def}}{=} \mathbb{E}[\mathcal{W}^{k+1} | x^k, w_1^k, ..., w_n^k]$.

Lemma 3. *The random iterates of the Algorithm 1 with τ-nice sampling satisfy the identity*

$$\mathbb{E}_k[\mathcal{W}^{k+1}] = \frac{\tau}{n} \|x^{k+1} - x^*\|^2 + \left(1 - \frac{\tau}{n}\right) \mathcal{W}^k \tag{7}$$

Then follows the theorem, which estimates the convergence rate of the Algorithm 1 with τ-nice sampling.

Theorem 1. *Assume that every f_i is μ-strongly convex and has H-Lipschitz Hessians. Then for the random iterates of the Algorithm 1 we have the recursion*

$$\mathbb{E}_k[\mathcal{W}^{k+1}] \leqslant \left(1 - \frac{\tau}{n} + \frac{\tau}{n} \left(\frac{H}{2\mu}\right)^2 \mathcal{W}^k\right) \mathcal{W}^k. \tag{8}$$

Moreover, if $\forall i\ \|w_i^0 - x^*\| \leqslant \frac{\mu}{H}$ *then*

$$\mathbb{E}_k[\mathcal{W}^{k+1}] \leqslant \left(1 - \frac{3\tau}{4n}\right)\mathcal{W}^k. \tag{9}$$

3.2 Sampling Strategies

3.2.1 Doubly Uniform Strategies

To begin with, we discuss the sampling strategies that were presented in [15] in relation to the Parallel Coordinate Descend method. These strategies are *Doubly Uniform.*

Definition 2 (Doubly Uniform (DU) sampling)**.** This is a sampling strategy that generates all sets of equal cardinality with equal probability. That is, if $|S_1| = |S_2|$ then $\mathbb{P}(S_1) = \mathbb{P}(S_2)$.

The name comes from the fact this property is stronger than *uniformity*, which means that all $i \in [n]$ have equal probability to be included in the chosen set $\hat{S}$. That can be seen if we denote the probability $q_j = \mathbb{P}(|\hat{S}| = j)$ for $j = 0, 1, ...n$. Then for every $S \subseteq [n]$ we have $\mathbb{P}(S) = q_j / \binom{n}{j}$. Hence

$$p_i = \sum_{j=1}^{n} \sum_{|S|=j, i\in S} \mathbb{P}(S) = \sum_{j=1}^{n} \sum_{|S|=j, i\in S} \frac{q_j}{\binom{n}{j}} = \sum_{j=1}^{n} \frac{\binom{n-1}{j-1} q_j}{\binom{n}{j}} = \frac{1}{n}\sum_{j=1}^{n} j q_j = \frac{\mathbb{E}[|\hat{S}|]}{n} \tag{10}$$

It is clear that a Doubly Uniform sampling strategy is determined by the vector q of probability distribution on the possible sizes $j = 0, 1, ..., n$ of the set $\hat{S}$. There are some interesting DU strategies.

1. τ-**Nice sampling.** Fix $1 \leqslant \tau \leqslant n$. A sampling is called τ-nice if it is DU with $q_\tau = 1$. This is the strategy used by the authors of the Algorithm 1. This strategy can be interpreted in terms of multi-process computation. So, there are τ processors availible. On the moment of sampling we choose a batch of size τ and assign each train object $i \in \hat{S}$ to a dedicated processor. Therefore, each processor will update the values of the objective function, gradients and Hessians with respect to new w_i^k.
2. τ-**Independent sampling.** Fix $1 \leqslant \tau \leqslant n$. A τ-independent sampling is a DU sampling with

$$q_k = \begin{cases} \binom{n}{k} c_k, & k = 1, 2, ..., \tau \\ 0, & k = \tau + 1, ..., n \end{cases} \tag{11}$$

where $c_1 = (1/n)^\tau$ and $c_k = (k/n)^\tau - \sum_{i=1}^{k-1} \binom{k}{i} c_i$ for $k \geqslant 2$.

The interpretation is that each of τ processors chooses on of the train objects $i \in [n]$. If several processors have chosen the same train object, then only one of them gets access to it. This strategy is very easy to implement in terms of parallel computation. Besides, when batches are small, we have $\tau \ll n$ and then τ-independent sampling is a good approximation of τ-nice variant.

3. (τ, p_b)-**Binomial sampling.** Fix $1 \leqslant \tau \leqslant n$ and $0 \leqslant p_b \leqslant 1$. The strategy is a DU sampling with

$$q_k = \binom{\tau}{k} p_b^k (1 - p_b)^k, \quad k \leqslant \tau \tag{12}$$

That is a model of the situation when each of τ processors is available at the moment of sampling with probability p_b, hence q_k is the probability that k processors are available.

3.2.2 Independent strategies

Apart from Doubly Uniform strategies we consider strategies, which make a decision about each train object $i \in [n]$ independently with its own probability p_i. Notably, in case $p_1 = p_2 = ... = p_n$ the strategy remains DU. But in general case some changes appear in the convergence rate estimation. There is the a strategy called Important Sampling, which assigns probabilities proportionally to the Lipschitz parameters (3) of the Hessians $\nabla^2 f_i(x)$ (or gradients $\nabla f_i(x)$).

Importance sampling. The strategy is to assign

$$p_i = \frac{H_i}{\sum_{i=1}^{n} H_i} \tag{13}$$

3.2.3 Consecutive strategy

Lastly, we explore a strategy which consists in consecutively passing the shuffled data:

Consecutive strategy. At the first iteration of the Algorithm 1 we fix a random permutation π of all data points indexes $[n]$, and batch size τ. Then on the iteration number k we take train objects with the following numbers:

$$\pi(k+1 \bmod n), \pi(k+2 \bmod n), ..., \pi(k+\tau \bmod n). \tag{14}$$

And if $(k \bmod n) + \tau > n$ then we simply take all the objects that have numbers are between $k \bmod n$ and n, so on this step the batch size can be smaller than τ.

3.3 Convergence rate

3.3.1 Estimations for Doubly Uniform sampling

To determine the convergence rate of the Algorithm 1 with Doubly Uniform sampling (2) we turn to the Lemma 3, since it is the first place in the Proofs for the Algorithm 1 which depends on the sampling strategy. It is generalized to an arbitrary DU sampling.

Lemma 4. *The random iterates of the Algorithm 1 with Doubly Uniform sampling satisfy*

$$\mathbb{E}_k[\mathcal{W}^{k+1}] = p\mathbb{E}_k[\|x^{k+1} - x^*\|^2] + (1-p)\,\mathcal{W}^k, \tag{15}$$

where $p = \frac{\mathbb{E}[|\hat{S}|]}{n}$.

Proof. The sampling strategy being Doubly Uniform implies that all train objects have equal probability $p = \frac{\mathbb{E}[|\hat{S}|]}{n}$ to be included in the chosen set $\hat{S}$. Therefore, by the linearity of expectation, we obtain the result. □

Thereby the final estimation has the same form as the one presented in [10]:

$$\mathbb{E}_k[\mathcal{W}^{k+1}] \leqslant \left(1 - p + p\left(\frac{H}{2\mu}\right)^2 \mathcal{W}^k\right)\mathcal{W}^k \tag{16}$$

in general, and in case $\forall i\ \|w_i^0 - x^*\| \leqslant \frac{\mu}{H}$:

$$\mathbb{E}_k[\mathcal{W}^{k+1}] \leqslant \left(1 - \frac{3}{4}p\right)\mathcal{W}^k. \tag{17}$$

That means, if we fix the expected batch size $\mathbb{E}[|\hat{S}|] = \tau$ then the convergence rate is the same as in τ-nice sampling.

3.3.2 Independent and Consecutive strategies

Now we consider the general case of choosing each train object independently with probability p_i.

Proposition 1. *In a particular iteration of Algorithm 1 given a fixed expectation of the batch size* $\mathbb{E}_k[|\hat{S}^k|] = \tau$ *the minimum value in the estimation of* $\mathbb{E}_k[\mathcal{W}^{k+1}]$ *is obtained by setting* $p_i = 1$ *for several* $i \in [n]$*, possibly* $p_i \in (0,1)$ *for a single object* i*, and* $p_i = 0$ *for others.*

Proof. We can generalize the estimation from Lemma 4 to our case.

$$n \cdot \mathbb{E}_k[\mathcal{W}^{k+1}] = \sum_{i=1}^{n} p_i \|x^{k+1} - x^*\|^2 + \sum_{i=1}^{n} (1-p_i)\,\|w_i^k - x^*\|^2. \tag{18}$$

Considering $\tau = \mathbb{E}_k[|\hat{S}^k|] = \sum_{i=1}^{n} p_i$, we have

$$n \cdot \mathbb{E}_k[\mathcal{W}^{k+1}] = \left(\tau\|x^{k+1} - x^*\|^2 + n \cdot \mathcal{W}^k\right) - \sum_{i=1}^{n} p_i \|w_i^k - x^*\|^2. \tag{19}$$

So the goal is to maximize $\sum_{i=1}^{n} p_i \|w_i^k - x^*\|^2$ satisfying the following:

$$\begin{cases} \sum_{i=1}^{n} p_i = \tau \\ 0 \leqslant p_i \leqslant 1 \ \forall i \end{cases} \tag{20}$$

The solution to this Linear Programming problem is the vertex of the polyhedron defined by the constraints (20). The optimal vertex has the maximum projection of its coordinate vector $(p_1, ..., p_n)^\top$ on the vector $(\|w_i - x^*\|^2, ..., \|w_n - x^*\|^2)$. That is the vertex, which coefficients corresponding to the biggest $\|w_i - x^*\|^2$ are set to 1, corresponding to the smallest $\|w_i - x^*\|^2$ are set to 0, and one coordinate is intermediate if τ is not an integer. □

In this way, we can see that it is sensible to update those parameter vectors w_i^k, which have not been updated recently. It means that an optimal independent strategy can be "approximated" by the consecutive strategy. This proposition is illustrated in the experiments section (4.3).

3.3.3 Notes about Importance Sampling

In some methods, particularly in the traditional SGD, the benefit of applying Importance Sampling is to replace the estimations determined by H_{max} - the maximum of the Lipschitzness parameters H_i of each f_i - with the ones determined by $\overline{H}$ - the mean of all H_i. This improvement is achieved by considering a problem, similar to the original (1), but with each f_i scaled inversely proportionally to H_i, in order to preserve the expectation of the random estimation of $f(x)$ and $\nabla f(x)$ equal to the actual values f and $\nabla f(x)$.

However, the Algorithm 1 has slightly different nature. That implies the fact that the direction of a step is not estimated by information from only those objects that have been included in the batch on the current iteration. This direction is calculated taking into account all the information from the previous step with some adjustments from the current step. In this case, the aforementioned technique cannot be easily involved. Therefore it is very hard to obtain any particular estimations for the Importance Sampling applied to the Algorithm 1.

4 Experiments

We run the experiments on a dataset obtained by `make_classification` function from `sklearn` with 500 objects and 5 features. The code can be found in the repository of the project (link). The objective is l_2-regularized logistic regression: $f_i(x) = \log(1 + \exp(-b_i \cdot a_i^T x)) + \frac{\lambda}{2}\|x\|_2^2$, where $x, a_1, ..., a_n \in \mathbb{R}^d$, $b_i = \pm 1$ and the regularizer λ is set to 0.01. On each figure we show two graphs. The first represents the trajectory of $\mathcal{W}^k = \frac{1}{n}\sum_{i=1}^{n} \|w_i^k - x^*\|^2$ and the second represents the history of the functional suboptimality. The x-axis is the number of the passes through the dataset.

4.1 Doubly Uniform strategies

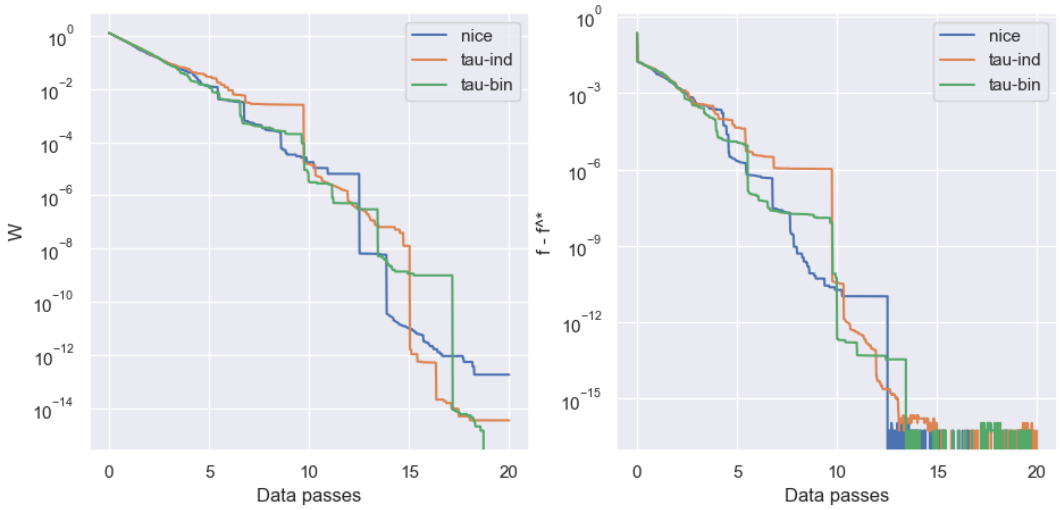

Figure 1: Comparison of the τ-nice, τ-independent and τ-binomial strategies

We see that all the doubly uniform strategies, having the same expected batch size, perform the same convergence rate.

4.2 Importance Sampling

We apply both Important Sampling strategies based on Hessians and gradients Lipschitzness parameters. Those are $L_i = \frac{1}{4}\|a_i\|^2$ and $H_i = \frac{1}{10}\|a_i\|^3$ for the logistic regression problem. It is clear that for the considered problem the Importance Sampling used with the Algorithm 1 does not bring practical improvements.

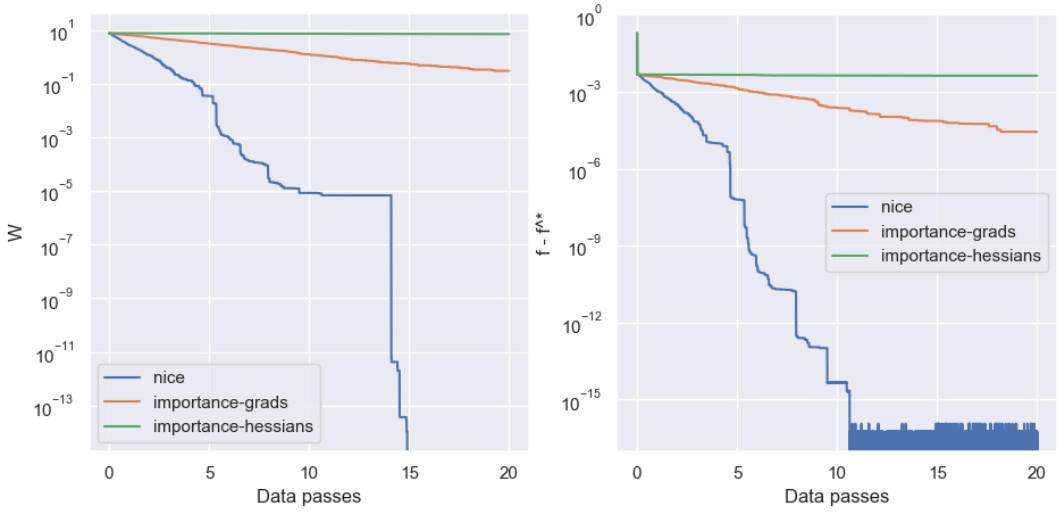

Figure 2: Comparison of the τ-nice and the Importance Sampling

4.3 Consecutive strategy

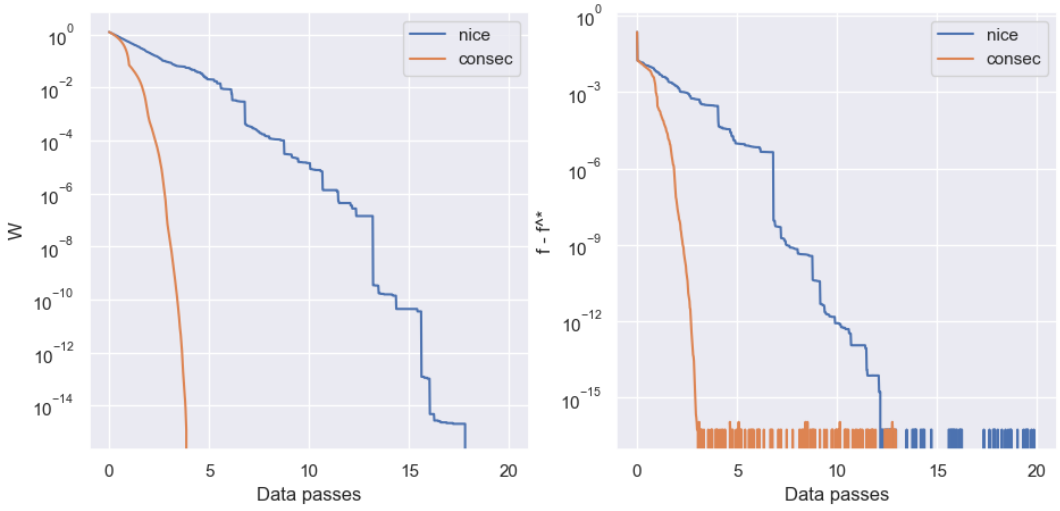

Figure 3: Comparison of the τ-nice and the consecutive strategy

In this case we have a practical improvement. The consecutive strategy does outperform the baseline τ-nice one. Nevertheless, theoretical explainations of this fact are still unclear.

5 Conclusion

So, we do a comparison of different sampling strategies applied to the Stochastic Newton algorithm (1). Firstly, we generalize the convergence rate estimations to all Doubly Uniform strategies and determine the practical equivalence of the most common representatives of this type of sampling (Lemma 4, Equation 16, Figure 4.1). Secondly, we obtain that the Importance Sampling is not likely to outperform the usual τ-nice strategy, and has been noticed to be inferior in practice (Figure 4.2). Finally, the improvements come from applying the consecutive strategy (Figure 4.3). This effect is encouraging, and therefore needs to be studied more precisely.

References

[1] Raghu Bollapragada, Richard H Byrd, and Jorge Nocedal. Exact and inexact subsampled Newton methods for optimization. *IMA Journal of Numerical Analysis*, 39(2):545–578, 04 2018.

[2] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent, 2019.

[3] Robert Gower, Dmitry Kovalev, Felix Lieder, and Peter Richtarik. Rsn: Randomized subspace newton. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[4] Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtarik. Sgd: General analysis and improved rates. 2019.

[5] Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. Sgd: General analysis and improved rates. In *International conference on machine learning*, pages 5200–5209. PMLR, 2019.

[6] Filip Hanzely and Peter Richtárik. One method to rule them all: Variance reduction for data, parameters and many new methods, 2019.

[7] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[8] Sai Praneeth Karimireddy, Sebastian U. Stich, and Martin Jaggi. Global linear convergence of newton's method without strong-convexity or lipschitz gradients, 2018.

[9] Jonas Moritz Kohler and Aurelien Lucchi. Sub-sampled cubic regularization for non-convex optimization. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70

of *Proceedings of Machine Learning Research*, pages 1895–1904. PMLR, 06–11 Aug 2017.

[10] Dmitry Kovalev, Konstantin Mishchenko, and Peter Richtárik. Stochastic newton and cubic newton methods with simple local linear-quadratic rates. *arXiv preprint arXiv:1912.01597*, 2019.

[11] Huikang Liu, Xiaolu Wang, Jiajin Li, and Anthony Man-Cho So. Low-cost lipschitz-independent adaptive importance sampling of stochastic gradients. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2150–2157, 2021.

[12] Konstantin Mishchenko, Ahmed Khaled, and Peter Richtárik. Random reshuffling: Simple analysis with vast improvements. *Advances in Neural Information Processing Systems*, 33:17309–17320, 2020.

[13] Yurii Nesterov. *Introductory lectures on convex optimization: a basic course/ by Yurii Nesterov*. Applied optimization, 87. Kluwer Academic Publishers, Boston, 2004.

[14] Lam Nguyen, PHUONG HA NGUYEN, Marten van Dijk, Peter Richtarik, Katya Scheinberg, and Martin Takac. SGD and hogwild! Convergence without the bounded gradients assumption. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3750–3758. PMLR, 10–15 Jul 2018.

[15] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156:433–484, 2016.

[16] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400 – 407, 1951.

[17] Nicolas Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[18] Nilesh Tripuraneni, Mitchell Stern, Chi Jin, Jeffrey Regier, and Michael I Jordan. Stochastic cubic regularization for fast nonconvex optimization. *Advances in neural information processing systems*, 31, 2018.

[19] Junyu Zhang, Lin Xiao, and Shuzhong Zhang. Adaptive stochastic variance reduction for subsampled newton method with cubic regularization. *INFORMS Journal on Optimization*, 4(1):45–64, 2022.

[20] Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling, 2014.

[21] Dongruo Zhou and Quanquan Gu. Stochastic recursive variance-reduced cubic regularization methods. In *International Conference on Artificial Intelligence and Statistics*, pages 3980–3990. PMLR, 2020.

[22] Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic variance-reduced cubic regularized newton methods. In *International Conference on Machine Learning*, pages 5990–5999. PMLR, 2018.