Towards closing the gap between the theory and practice of SVRG

Othmane Sebbouh, Nidham Gazagnadou, Samy Jelassi, Francis Bach, Robert M Gower

Towards closing the gap between the theory and practice of SVRG

Othmane Sebbouh
Télécom Paris, IPP, Paris
othmane.sebbouh@gmail.com

Nidham Gazagnadou
Télécom Paris, IPP, Paris
nidham.gazagnadou@gmail.com

Samy Jelassi
ORFE Department
Princeton University
sjelassi@princeton.edu

Francis Bach
INRIA - Ecole Normale Suprieure
PSL Research University
francis.bach@inria.fr

Robert Gower
Télécom Paris, IPP, Paris
robert.gower@telecom-paristech.fr

Abstract

Amongst the very first variance reduced stochastic methods for solving the empirical risk minimization problem was the SVRG method [11]. SVRG is an inner-outer loop based method, where in the outer loop a reference full gradient is evaluated, after which $m \in \mathbb{N}$ steps of an inner loop are executed where the reference gradient is used to build a variance reduced estimate of the current gradient. The simplicity of the SVRG method and its analysis has lead to multiple extensions and variants for even non-convex optimization. Yet there is a significant gap between the parameter settings that the analysis suggests and what is known to work well in practice. Our first contribution is that we take several steps here towards closing this gap. In particular, the current analysis shows that m should be of the order of the condition number so that the resulting method has a favourable complexity. Yet in practice $m = n$ works well irregardless of the condition number, where n is the number of data points. Furthermore, the current analysis shows that the inner iterates have to be reset using averaging after every outer loop. Yet in practice SVRG works best when the inner iterates are updated continuously and not reset. We provide an analysis of these aforementioned practical settings and show that they achieve the same favourable complexity as the original analysis (with slightly better constants). Our second contribution is to provide a more general analysis than had been previously done by using arbitrary sampling, which allows us to analyse virtually all forms of mini-batching through a single theorem. Since our setup and analysis reflects what is done in practice, we are able to set the parameters such as the mini-batch size and step size using our theory in such a way that produces a more efficient algorithm in practice, as we show in extensive numerical experiments.

Preprint. Under review.

1 Introduction

Consider the problem of minimizing a μ–strongly convex and L–smooth function f where

$$x^* = \arg\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(x) =: f(x), \tag{1}$$

and each f_i is convex and L_i–smooth. Several training problems in machine learning fit this format, e.g. least-squares, logistic regressions and conditional random fields. Typically each f_i represents a regularized loss of an ith data point. When n is large, algorithms that rely on full passes over the data, such as gradient descent, are no longer competitive. Instead, the stochastic version of gradient descent SGD [22] is often used since it requires only a mini-batch of data to make progress towards the solution. However, SGD suffers from high variance, which keeps the algorithm from converging unless a carefully often hand-tuned decreasing sequence of stepsizes is chosen. This often results in a cumbersome parameter tuning and a slow convergence.

To address this issue, many variance reduced methods have been designed in recent years including SAG [23], SAGA [5] and SDCA [24] that require only a constant stepsize to achieve linear convergence. In this paper, we are interested in variance reduced methods with an inner-out loop structure, such as S2GD [12], SARAH [18], L-SVRG [14] and the orignal SVRG [11] algorithm. Here we present not only a more general analysis that allows for any mini-batching strategy, but also a more *practical* analysis, by analysing methods that are based on what works in practice, and thus providing an analysis that can inform practice.

2 Background and Contributions

Convergence under arbitrary samplings. We give the first arbitrary sampling convergence results for SVRG type methods in the convex setting[1]. That is our analysis includes all forms of sampling including mini-batching and importance sampling as a special case. To better understand the significance of this result, we use mini-batching b elements *without* replacement as a running example throughout the paper. With this sampling the update step of SVRG, starting from $x^0 = w_0 \in \mathbb{R}^d$, takes the form of

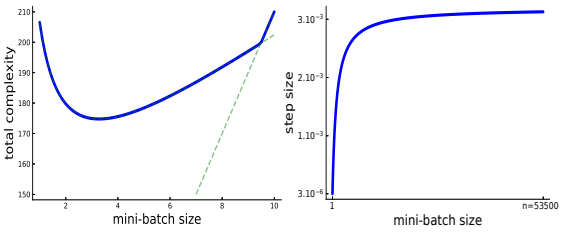


Figure 1: Left: the total complexity (3) for random Gaussian data, right: the step size (4) as b increases.

$$x^{t+1} = x^t - \alpha\left(\frac{1}{b}\sum_{i\in B}\nabla f_i(x^t) - \frac{1}{b}\sum_{i\in B}\nabla f_i(w_{s-1}) + \nabla f(w_{s-1})\right), \tag{2}$$

where $\alpha > 0$ is the stepsize, $B \subseteq [n] \overset{\text{def}}{=} \{1,\dots,n\}$ and $b = |B|$. Here w_{s-1} is the *reference point* which is updated after $m \in \mathbb{N}$ steps, the x^t's are the *inner iterates* and m is the loop length. As a special case of our forthcoming analysis in Corollary 4.1, we show that the *total complexity* of the SVRG method based on (2) to reach an $\epsilon > 0$ accurate solution has a simple expression which depends on n, m, b, μ, L and $L_{\max} \overset{\text{def}}{=} \max_{i\in[n]} L_i$:

$$C_m(b) \overset{\text{def}}{=} 2\left(\frac{n}{m} + 2b\right)\max\left\{\frac{3}{b}\frac{n-b}{n-1}\frac{L_{\max}}{\mu} + \frac{n}{b}\frac{b-1}{n-1}\frac{L}{\mu}, m\right\}\log\left(\frac{1}{\epsilon}\right), \tag{3}$$

so long as the stepsize is

$$\alpha = \frac{1}{2}\frac{b(n-1)}{3(n-b)L_{\max} + n(b-1)L}. \tag{4}$$

By total complexity we mean the total number of individual ∇f_i gradients evaluated. This shows that the total complexity is a simple function of the loop length m and the mini-batch size b. See Figure 1 for an example for how total complexity evolves as we increase the mini-batch size.

[1] SVRG has very recently been analyzed under arbitrary samplings in the non-convex setting [10].

Optimal mini-batch and step sizes for SVRG. The size of the mini-batch b is often left as a parameter for the user to choose or set using a rule of thumb. The current analysis in the literature for mini-batching shows that when increasing the mini-batch size b, while the iteration complexity can decrease[2], the total complexity increases or is invariate. See for instance results in the nonconvex case [19, 21], and for the convex case [9], [13], [1] and finally [15] where one can find the iteration complexity of several variants of SVRG with mini-batching. However, in practice, mini-batching can often lead to faster algorithms. In contrast our total complexity (3) clearly highlights that when increasing the mini batch size, the total complexity can decrease and the stepsize increases, as can be seen in our plot of (3) and (4) in Figure 1. Furthermore $C_m(b)$ is a convex function in b which allows us to determine the optimal mini-batch a priori. For $m = n$, a widely used loop length in practice, the optimal mini-batch size, depending on the problem setting, is given in Table 1. Moreover, we can also determine the optimal loop length The reason we were able to achieve these

$n \leq \frac{L}{\mu}$		$\frac{L}{\mu} < n < \frac{3L_{\max}}{\mu}$		$n \geq \frac{3L_{\max}}{\mu}$
$L_{\max} \geq \frac{nL}{3}$	$L_{\max} < \frac{nL}{3}$	$L_{\max} \geq \frac{nL}{3}$	$L_{\max} < \frac{nL}{3}$	
n	$\left\lfloor \hat{b} \right\rfloor$	$\left\lfloor \tilde{b} \right\rfloor$	$\left\lfloor \min\{\hat{b}, \tilde{b}\} \right\rfloor$	1

Table 1: Optimal mini-batch sizes for Algortithm 1 with $m = n$. The last line presents the optimal mini-batch sizes depending on all the possible problem settings, which are presented in the first two lines. Notations: $\hat{b} = \sqrt{\frac{n}{2}\frac{3L_{\max}-L}{nL-3L_{\max}}}$, $\tilde{b} = \frac{(3L_{\max}-L)n}{n(n-1)\mu-nL+3L_{\max}}$.

new tighter mini-batch complexity bounds was by using the recently introduced concept of expected smoothness [8] alongside a new constant we introduce in this paper called the expected residual constant. The expected smoothness and residual constants, which we present later in Lemmas 4.1 and 4.2, show how mini-batching (and arbitrary sampling in general) combined with the smoothness of our data can determine how smooth in expectation our resulting mini-batched functions are. The expected smoothness constant has been instrumental in providing a tight mini-batch analysis for SGD [7], SAGA [6] and now SVRG.

New practical variants. We took special care so that our analysis allows for practical parameter settings. In particular, often the loop length is set to $m = n$ or $m = n/b$ in the case of mini-batching[3]. And yet, the classical SVRG analysis given in [11] requires $m \geq 20L_{\max}/\mu$ in order to ensure a resulting iteration complexity of $O((n + L_{\max}/\mu)\log(1/\epsilon))$. Furthermore, the standard SVRG analysis relies on averaging the x^t inner iterates after every m iterations of (2), yet this too is not what works well in practice[4]. To remedy this, we propose *Free-SVRG*, a variant of SVRG where the inner iterates are not averaged at any point. Furthermore, by developing a new Lyapunov style convergence for *Free-SVRG*, our analysis holds for any choice of m, and in particular, for $m = n$ we show that the resulting complexity is also given by $O((n + L_{\max}/\mu)\log(1/\epsilon))$.

The only downside of *Free-SVRG* is that the reference point is set using a weighted averaging based on the strong convexity parameter. To further fix this issue, we introduce *L-SVRG-D*, an improved variant of *Loopless-SVRG* [14] that has no explicit inner-loop structure and instead updates the reference point based on a coin toss. *L-SVRG-D* requires no knowledge of the strong convexity

[2]Note that the total complexity is equal to the iteration complexity times the mini-batch size b.

[3]See for example the lightning package from scikit-learn [20]: http://contrib.scikit-learn.org/lightning/ and [18] for examples where $m = n$. See [2] for an example where $m = 5n/b$.

[4]Perhaps an exception to the above issues in the literature is the Katyusha method and its analysis [1], which is an accelerated variant of SVRG. In [1] the author shows that using a loop length $m = 2n$ and by not averaging the inner iterates, the Katyusha method achieves the accelerated complexity of $O((n+\sqrt{(nL_{\max})/\mu})\log(1/\epsilon))$. Though a remarkable advance in the theory of accelerated methods, the analysis in [1] does not hold for the unaccelerated case. This is important since, contrary to the name, the accelerated variants of stochastic methods are not always faster than their non-accelerated counterparts. Indeed, acceleration only helps in the stochastic setting when $L_{\max}/\mu \geq n$, in other words for problems that are sufficiently ill-conditioned.

parameter and no averaging whatsoever, and achieves the same complexity as *Free-SVRG*, albeit at the cost of introducing more variance into the procedure due to the coin toss.

3 Assumptions and Sampling

We collect all of the assumptions we use in the following.

Assumption 3.1. *There exist $L \geq 0$ and $\mu \geq 0$ such that for all $x, y \in \mathbb{R}^d$,*

$$f(x) \quad \leq \quad f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2} \|x - y\|_2^2, \tag{5}$$

$$f(x) \quad \leq \quad f(y) + \langle \nabla f(x), x - y \rangle - \frac{\mu}{2} \|x - y\|_2^2. \tag{6}$$

We say that f is L–smooth (5) *and μ–strongly convex* (6)*. Moreover, for all $i \in [n]$, f_i is convex and there exists $L_i \geq 0$ such that f_i is L_i–smooth.*

So that we can analyse all forms of mini-batching simulanously through arbitrary sampling we make use of a *sampling vector*.

Definition 3.1 (The sampling vector)**.** *We say that the random vector $v = [v_1, \ldots, v_n] \in \mathbb{R}^n$ with distribution $\mathcal{D}$ is a sampling vector if $\mathbb{E}_{\mathcal{D}}[v_i] = 1$ for all $i \in [n]$.*

With a sampling vector we can compute an unbiased estimate of $f(x)$ and $\nabla f(x)$ via

$$f_v(x) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^n v_i f_i(x) \quad \text{and} \quad \nabla f_v(w) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^n v_i \nabla f_i(x). \tag{7}$$

Indeed these are unbiased estimators since

$$\mathbb{E}_{\mathcal{D}}[f_v(x)] \;=\; \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\mathcal{D}}[v_i] f_i(x) \;=\; \frac{1}{n} \sum_{i=1}^n f_i(x) \;=\; f(x). \tag{8}$$

Likewise we can show that $\mathbb{E}_{\mathcal{D}}[\nabla f_v(x)] = \nabla f(x)$. Computing ∇f_v is cheaper than computing the full gradient ∇f whenever v is a sparse vector. In particular, this is the case when the support of v is based on a mini-batch sampling.

Definition 3.2 (Sampling)**.** *A sampling $S \subseteq [n]$ is any random set-valued map which is uniquely defined by the probabilities $\sum_{B \subseteq [n]} p_B = 1$ where $p_B \stackrel{def}{=} \mathbb{P}(S = B), \quad \forall B \subseteq [n]$. A sampling S is called proper if for every $i \in [n]$, we have that $p_i \stackrel{def}{=} \mathbb{P}[i \in S] = \sum_{C : i \in C} p_C > 0$.*

We can build a sampling vector using sampling as follows.

Lemma 3.1 (Sampling vector)**.** *Let S be a proper sampling. Let $p_i \stackrel{def}{=} \mathbb{P}[i \in S]$ and $\mathbf{P} \stackrel{def}{=} \operatorname{Diag}(p_1, \ldots, p_n)$. Let $v = v(S)$ be a random vector defined by*

$$v(S) \;=\; \mathbf{P}^{-1} \sum_{i \in S} e_i \;\stackrel{def}{=}\; \mathbf{P}^{-1} e_S. \tag{9}$$

It follows that v is a sampling vector.

Proof. The i-th coordinate of $v(S)$ is $v_i(S) = \mathbb{1}(i \in S)/p_i$ and thus

$$\mathbb{E}[v_i(S)] \;=\; \frac{\mathbb{E}[\mathbb{1}(i \in S)]}{p_i} \;=\; \frac{\mathbb{P}[i \in S]}{p_i} \;=\; 1. \qquad \square$$

Our forthcoming analysis holds for all samplings. However, we will pay particular attention to b-nice sampling, otherwise known as mini-batching without replacement, since it is often used in practice.

Definition 3.3 (b-nice sampling)**.** *S is a b-nice sampling if it is sampling such that*

$$\mathbb{P}[S = B] = \frac{1}{\binom{n}{b}}, \quad \forall B \subseteq [n], \;\; \textit{with} \;\; |B| = b.$$

To construct such a sampling vector based on the b–nice sampling, note that $p_i = \frac{b}{n}$ for all $i \in [n]$ and thus we have that $v(S) = \frac{n}{b}\sum_{i\in S} e_i$ according to Lemma 3.1. The resulting subsampled function is then $f_v(x) = \frac{1}{|S|}\sum_{i\in S} f_i(x)$, which is simply the mini-batch average over S.

Using arbitrary sampling also allows us to consider non-uniform samplings, and for completeness, we present this sampling and several others in Appendix D.

4 *Free-SVRG*: freeing up the inner loop size

Similarly to SVRG, *Free-SVRG* is an inner-outer loop variance reduced algorithm. It differs from the original SVRG [11] on two major ponts: how the reference point is reset and how the first iterate of the inner loop is defined, see Algorithm 1.

First, in SVRG, the reference point is the average of the iterates of the inner loop. Thus, old iterates and recent iterates have equal weights in the average. This is counterintuitive as one would expect that to reduce the variance of the gradient estimate used in (2), one needs a reference point which is closer to the more recent iterates. This is why, inspired by [17], we use the weighted averaging in *Free-SVRG* given in (10), which gives more importance to recent iterates compared to old ones.

Second, in SVRG, the first iterate of the inner loop is reset to the reference point. Thus, the inner iterates of the algorithm are not updated using a one step recurrence. In contrast, *Free-SVRG* defines the first iterate of the inner loop as the last iterate of the previous inner loop, as is also done in practice. These changes and a new Lyapunov function analysis is what allows us to freely choose the size of the inner loop[5]. To declutter the notation, we define for a given step size $\alpha > 0$:

$$S_m \overset{\text{def}}{=} \sum_{i=0}^{m-1} (1-\alpha\mu)^{m-1-i} \quad \text{and} \quad p_t \overset{\text{def}}{=} \frac{(1-\alpha\mu)^{m-1-t}}{S_m}, \quad \text{for } t = 0, \ldots, m-1. \tag{10}$$

Algorithm 1 *Free-SVRG*

Parameters inner-loop length m, step size α, a sampling vector $v \sim \mathcal{D}$, and p_t defined in (10)
Initialization $w_0 = x_0^m \in \mathbb{R}^d$
for $s = 1, 2, \ldots$ **do**
  $x_s^0 = x_{s-1}^m$
  for $t = 0, 1, \ldots, m-1$ **do**
    Sample $v_t \sim \mathcal{D}$
    $g_s^t = \nabla f_{v_t}(x_s^t) - \nabla f_{v_t}(w_{s-1}) + \nabla f(w_{s-1})$
    $x_s^{t+1} = x_s^t - \alpha g_s^t$
  $w_s = \sum_{t=0}^{m-1} p_t x_s^t$

4.1 Convergence analysis

Our analysis relies on two important constants called the *expected smoothness* constant and the *expected residual* constant. Their existence is a result of the smoothness of the function f and that of the individual function $f_i, i \in [n]$.

Lemma 4.1 (Expected smoothness. Theorem 3.6 in [7]). *Let $v \sim \mathcal{D}$ be a sampling vector and assume that Assumption 3.1 holds. There exists $\mathcal{L} \geq 0$ such that for all $x \in \mathbb{R}^d$,*

$$\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(x) - \nabla f_v(x^*)\|_2^2\right] \leq 2\mathcal{L}\left(f(x) - f(x^*)\right). \tag{11}$$

Lemma 4.2 (Expected residual. Lemma F.1). *Let $v \sim \mathcal{D}$ be a sampling vector and assume that Assumption 3.1 holds. There exists $\rho \geq 0$ such that for all $x \in \mathbb{R}^d$,*

$$\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(x) - \nabla f_v(x^*) - \nabla f(x)\|_2^2\right] \leq 2\rho\left(f(x) - f(x^*)\right). \tag{12}$$

[5]Hence the name of our method *free*-SVRG.

For completeness, the proof of Lemma 4.1 is given in Lemma E.1 in the supplementary material. The proof of Lemma F.1 is also given in the supplementary material, in Lemma F.1. Indeed, all proofs are deferred to the supplementary material.

Though Lemma 4.1 establishes the existence of the expected smoothness $\mathcal{L}$, it was only very recently that a tight estimate of $\mathcal{L}$ was conjectured in [6] and proven in [7]. In particular, for our working example of b–nice sampling, we have that the constants $\mathcal{L}$ and ρ have simple closed formulae that depend on b.

Lemma 4.3 ($\mathcal{L}$ and ρ for b-nice sampling)**.** *Let v be a sampling vector based on the b–nice sampling. It follows that.*

$$\mathcal{L} = \mathcal{L}(b) \quad \overset{def}{=} \quad \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L, \tag{13}$$

$$\rho = \rho(b) \quad \overset{def}{=} \quad \frac{1}{b}\frac{n-b}{n-1}L_{\max}. \tag{14}$$

The reason that the expected smoothness and expected residual constants are so useful in obtaining a tight mini-batch analysis is because, as the mini-batch size b goes from n to 1, $\mathcal{L}(b)$ (resp. $\rho(b)$) gracefully interpolates between the smoothness of the full function $\mathcal{L}(n) = L$ (resp. $\rho(n) = 0$), and the smoothness of the individual f_i functions $\mathcal{L}(1) = L_{\max}$ (resp $\rho(1) = L_{\max}$). Also, we can bound the second moment of a variance reduced gradient estimate using $\mathcal{L}$ and ρ as follows.

Lemma 4.4. *Let Assumption 3.1 hold. Let $x, w \in \mathbb{R}^d$ and $v \sim \mathcal{D}$ be sampling vector. Consider $g(x,w) \overset{def}{=} \nabla f_v(x) - \nabla f_v(w) + \nabla f(w)$. As a consequence of* (11) *and* (12) *we have that*

$$\mathbb{E}_{\mathcal{D}}\left[\|g(x,w)\|_2^2\right] \leq 4\mathcal{L}(f(x) - f(x^*)) + 4\rho(f(w) - f(x^*)). \tag{15}$$

Next we present a new Lyaponuv style convergence analysis through which we will establish the convergence of the iterates and the function values simultaneously.

Theorem 4.1. *Consider the setting of Algorithm 1 and the following Lyapunov function*

$$\phi_s \overset{def}{=} \|x_s^m - x^*\|_2^2 + \psi_s \quad \textit{where} \quad \psi_s \overset{def}{=} 8\alpha^2\rho S_m(f(w_s) - f(x^*)). \tag{16}$$

If Assumption 3.1 holds and if $\alpha \leq \frac{1}{2(\mathcal{L}+2\rho)}$, then

$$\mathbb{E}\left[\phi_s\right] \;\leq\; \beta^s\phi_0, \quad \textit{where} \quad \beta = \max\left\{(1-\alpha\mu)^m, \tfrac{1}{2}\right\}. \tag{17}$$

4.2 Total complexity for b–nice sampling

To gain better insight into the convergence rate stated in Theorem 4.1, we present the total complexity of Algorithm 1 when v is defined via the b–nice sampling introduced in Definition 3.3.

Corollary 4.1. *Consider the setting of Algorithm 1 and suppose that we use b–nice sampling. Let $\alpha = \frac{1}{2(\mathcal{L}(b)+2\rho(b))}$, where $\mathcal{L}(b)$ and $\rho(b)$ are given in* (13) *and* (14)*. We have that the total complexity of finding an $\epsilon > 0$ approximate solution that satisfies $\mathbb{E}\left[\|x_s^m - x^*\|_2^2\right] \leq \epsilon\,\phi_0$ is*

$$C_m(b) \quad \overset{def}{=} \quad 2\left(\frac{n}{m} + 2b\right)\max\left\{\frac{\mathcal{L}(b) + 2\rho(b)}{\mu}, m\right\}\log\left(\frac{1}{\epsilon}\right). \tag{18}$$

Now (3) results from plugging (13) and (14) into (18). As an immediate sanity check, we check the two extremes $b = n$ and $b = 1$. When $b = n$ we would expect to recover the iteration complexity of gradient descent, as we do in the next corollary[6].

Corollary 4.2. *Consider the setting of Corollary 4.1 with $b = n$ and $m = 1$, thus $\alpha = \frac{1}{2(\mathcal{L}(n)+2\rho(n))} = \frac{1}{2L}$. Hence, the resulting total complexity* (18) *is given by $C_1(n) = 6n\frac{L}{\mu}\log\left(\frac{1}{\epsilon}\right)$.*

In practice, the most common setting is choosing $b = 1$ and the size of the inner loop $m = n$. Here we recover a complexity that is common to other non-accelerated algorithms [23], [5], [12], and for a range of values of m including $m = n$.

[6]Though the resulting complexity is 6 times the tightest gradient descent complexity, it is of the same order.

Corollary 4.3. *Consider the setting of Corollary 4.1 with $b = 1$ and thus $\alpha = \frac{1}{2(\mathcal{L}(1)+2\rho(1))} = \frac{1}{6L_{\max}}$. Hence the resulting total complexity* (18) *is given by $C_m(1) = 18\left(n + \frac{L_{\max}}{\mu}\right)\log\left(\frac{1}{\epsilon}\right)$, so long as $m \in \left[\min(n, \frac{L_{\max}}{\mu}), \max(n, \frac{L_{\max}}{\mu})\right]$.*

Thus total complexity is essentially invariant for $m = n$, $m = L_{\max}/\mu$ and everything inbetween.

5 *L-SVRG-D*: a decreasing step size approach

Although *Free-SVRG* solves multiple issues regarding the construction and analysis of SVRG, it still suffers from an important issue: it requires the knowledge of the strong convexity constant, as is the case for the original SVRG algorithm [11]. One can of course use an explicit small regularization parameter as a proxy, but this can result in a slower algorithm.

Recently a new loopless variant of SVRG was proposed and analysed in [14]. At each iteration, their method makes a coin toss. With (a low) probability p, typically $\frac{1}{n}$, the reference point is reset to a recent iterate, and with probability $1-p$, the reference point remains the same. This method does not require knowledge of the strong convexity constant.

Our method, *L-SVRG-D*, uses the same loopless structure as in [14] but introduces different step sizes at each iteration, see Algorithm 2. We initialize the step size to a fixed value $\alpha > 0$. At each iteration we toss a coin, and if it lands heads (with probability $1-p$) the step size decreases by a factor $\sqrt{1-p}$. If it lands tails (with probability p) the reference point is reset to the most recent iterate and the stepsize is reset to its initial value α.

This allows us to take larger steps than *L-SVRG* when we update the reference point, *i.e.,* when the variance of the unbiased estimate of the gradient is low, and smaller steps when this variance increases.

Algorithm 2 L-SVRG-D

Parameters step size α, $p \in (0,1]$, and a sampling vector $v \sim \mathcal{D}$
Initialization $w^0 = x^0 \in \mathbb{R}^d,\ \alpha_0 = \alpha$
for $k = 0, 1, 2, \ldots$ **do**

Sample $v_k \sim \mathcal{D}$
$g^k = \nabla f_{v_k}(x^k) - \nabla f_{v_k}(w^k) + \nabla f(w^k)$
$x^{k+1} = x^k - \alpha_k g^k$
$(w^{k+1}, \alpha_{k+1}) = \begin{cases} (x^k, \alpha) & \text{with probability } p \\ (w^k, \sqrt{1-p}\,\alpha_k) & \text{with probability } 1-p \end{cases}$

Theorem 5.1. *Consider the iterates of Algorithm 2 and the following Lyapunov function*

$$\phi^k \overset{def}{=} \left\|x^k - x^*\right\|_2^2 + \psi^k \quad \textit{where} \quad \psi^k \overset{def}{=} \frac{8\alpha_k^2\mathcal{L}}{p(3-2p)}\left(f(w^k) - f(x^*)\right), \quad \forall k \in \mathbb{N}. \tag{19}$$

If Assumption 3.1 holds and

$$\alpha \leq \frac{1}{2\zeta_p\mathcal{L}}, \quad \textit{where} \quad \zeta_p \overset{def}{=} \frac{(7-4p)(1-(1-p)^{\frac{3}{2}})}{p(2-p)(3-2p)}, \tag{20}$$

then

$$\mathbb{E}\left[\phi^k\right] \leq \beta^k\phi^0, \quad \textit{where} \quad \beta = \max\left\{1 - \frac{2}{3}\alpha\mu, 1 - \frac{p}{2}\right\}. \tag{21}$$

Remark 5.1. *To get a sense for the size of the step size given in* (20)*, it is easy to show that ζ_p is an increasing function of p such that $7/4 \leq \zeta_p \leq 3$. Since typically $p \approx 0$, we often take a step which is approximately $\alpha \leq 2/(7\mathcal{L})$.*

Corollary 5.1. *Consider the setting of Algorithm 2 and suppose that we use b–nice sampling. Let $\alpha = \frac{1}{2\zeta_p\mathcal{L}(b)}$. We have that the total complexity of finding an $\epsilon > 0$ approximate solution that satisfies*

$\mathbb{E}\left[\left\|x^k - x^*\right\|_2^2\right] \leq \epsilon\, \phi^0$ *is*

$$C_p(b) \overset{def}{=} 2(2b+pn)\max\left\{\frac{3\zeta_p}{2}\frac{\mathcal{L}(b)}{\mu}, \frac{1}{p}\right\}\log\left(\frac{1}{\epsilon}\right). \tag{22}$$

6 Optimal parameter settings: loop, mini-batch and step sizes

In this section, we restrict our analysis to b–nice sampling. First, we will determine the optimal loop size for Algorithm 1. Then, we will examine the optimal mini-batch and step sizes for particular choices of the inner loop size m for Algorithm 1 and of the probability of the reference point's update p in Algorithm 2, that play analogous roles. Note that the steps used in our algorithms depend on b through the expected smoothness constant $\mathcal{L}(b)$ and the expected residual constant $\rho(b)$. Hence, optimizing the total complexity in the mini-batch size also determines the optimal step size.

Examining the complexities of Algorithms 1 and 2, given in (18) and (22), we can see that, setting $p = 1/m$ in Algorithm 2, these complexities only differ by constants. Thus, to avoid redundancy, we present the optimal mini-batch sizes for Algorithm 2 in Appendix C and we will only consider here the complexity of Algorithm 1 given in (18).

6.1 Optimal loop size for Algorithm 1

Here we determine the optimal value of m for a fixed batch size b, denoted by $m^*(b)$, which minimizes the total complexity (18).

Proposition 6.1. *The loop size that minimizes* (18) *and the resulting total complexity is given by*

$$m^*(b) = \frac{\mathcal{L}(b)+2\rho(b)}{\mu} \quad \textit{and} \quad C_{m^*}(b) = 2\left(n + 2b\frac{\mathcal{L}(b)+2\rho(b)}{\mu}\right)\log\left(\frac{1}{\epsilon}\right). \tag{23}$$

For example when $b=1$, we have that $m^*(1) = 3L_{\max}/\mu$ and $C_{m^*}(1) = O((n+L_{\max}/\mu)\log(1/\epsilon))$, which is the same complexity as achieved by the range of m values given in Corollary 4.3. Thus, as we also observed in Corollary 4.3, the total complexity is not very sensitive to the choice of m, and $m = n$ is a perfectly safe choice as it achieves the same complexity as m^*. We also confirm this numerically with a series of experiments in Section G.3.

6.2 Optimal mini-batch and step sizes

In the following proposition, we determine the optimal mini-batch and step sizes for two practical choices of the size of the loop m.

Proposition 6.2. *Let* $b^* \overset{def}{=} \arg\min_{b\in[n]} C_m(b)$*, where* $C_m(b)$ *is defined in* (18)*. For the widely used choice* $m = n$*, we have that* b^* *is given by Table 1. For another widely used choice* $m = n/b$*, which allows to make a full pass over the data during each inner loop, we have*

$$b^* = \begin{cases} \left\lfloor \bar{b} \right\rfloor & \text{if } n > \frac{3L_{\max}}{\mu} \\ 1 & \text{if } \frac{3L_{\max}}{L} < n \leq \frac{3L_{\max}}{\mu} \\ n & \text{otherwise, if } n \leq \frac{3L_{\max}}{L} \end{cases}, \quad \textit{where} \quad \bar{b} \overset{def}{=} \frac{n(n-1)\mu - 3n(L_{\max}-L)}{3(nL - L_{\max})}. \tag{24}$$

Previously theory showed that the total complexity would increase as the mini-batch size increased, and thus established that single-element sampling was optimal. However, notice that for $m = n$ and $m = n/b$, the usual choices for m in practice, the optimal mini-batch size is different than 1 for a range of problem settings. Since our algorithms are closer to the SVRG variants used in practice, we argue that our results explain why mini-batching works in practice, as we verify in the next section.

7 Experiments

We performed a series of experiments on data sets from LIBSVM [4] and the UCI repository [3], to validate our theoretical findings. We tested l_2–regularized logistic regression on *ijcnn1* and *real-sim*,

and ridge regression on *slice* and *YearPredictionMSD*. We used two choices for the regularizer: $\lambda = 10^{-1}$ and $\lambda = 10^{-3}$. All of our code is implemented in `Julia 1.0`. Due to lack of space, most figures have been relegated to Section G in the supplementary material.

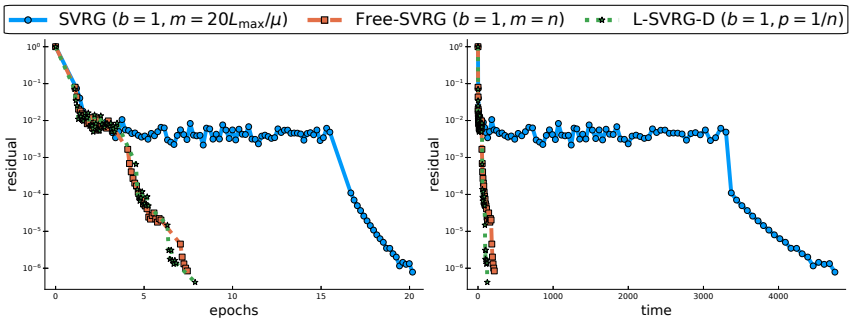


Figure 2: Algorithms comparison with theoretical settings on the *ijcnn1* data set with $\lambda = 10^{-3}$.

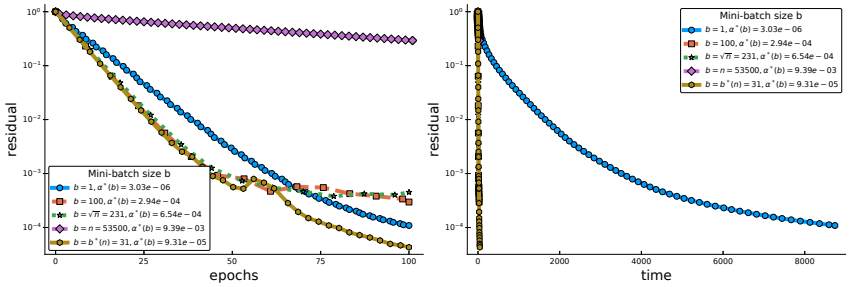


Figure 3: Impact of the mini-batch size on *Free-SVRG*, *slice* data set with $\lambda = 10^{-1}$.

Practical theory. Our first round of experiments aimed at verifying if our theory does indeed result in more practical algorithms. Indeed, we found that *Free-SVRG* and *L-SVRG-D* with the parameter setting given by our theory are often faster than SVRG with settings suggested by the theory in [11], that is $m = 20L_{\max}/\mu$ and $\alpha = 1/10L_{\max}$, see Figure 2, and Section G.1 for more experiments comparing different theoretical parameter settings.

Optimal mini-batch size. We also confirmed numerically that when using *Free-SVRG* with $m = n$, the optimal mini-batch size b^* derived in Table 1 was highly competitive as compared to the range of mini-batch sizes $b \in \{1, 100, \sqrt{n}, n\}$, see Figure 3 and several more such experiments in Section G.2.

Acknowledgments

RMG acknowledges the support by grants from DIM Math Innov Région Ile-de-France (ED574 - FMJH) , reference ANR-11-LABX-0056-LMH, LabEx LMH.

References

[1] Z. Allen-Zhu. "Katyusha: The First Direct Acceleration of Stochastic Gradient Methods". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2017. 2017, pp. 1200–1205.

[2] Z. Allen-Zhu and E. Hazan. "Variance Reduction for Faster Non-Convex Optimization". In: *Proceedings of The 33rd International Conference on Machine Learning*. Vol. 48. 2016, pp. 699–707.

[3] A. Asuncion and D. Newman. *UCI machine learning repository*. 2007.

[4] C.-C. Chang and C.-J. Lin. "LIBSVM: A library for support vector machines". In: *ACM transactions on intelligent systems and technology (TIST)* 2.3 (2011), p. 27.

[5] A. Defazio, F. Bach, and S. Lacoste-Julien. "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives". In: *Advances in Neural Information Processing Systems 27*. 2014, pp. 1646–1654.

[6] N. Gazagnadou, R. M. Gower, and J. Salmon. "Optimal mini-batch and step sizes for SAGA". In: *The International Conference on Machine Learning* (2019).

[7] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik. "SGD: general analysis and improved rates". In: ().

[8] R. M. Gower, P. Richtárik, and F. Bach. "Stochastic Quasi-Gradient methods: Variance Reduction via Jacobian Sketching". In: *arXiv:1805.02632* (2018).

[9] R. Harikandeh, M. O. Ahmed, A. Virani, M. Schmidt, J. Konečný, and S. Sallinen. "StopWasting My Gradients: Practical SVRG". In: *Advances in Neural Information Processing Systems 28*. 2015, pp. 2251–2259.

[10] S. Horváth and P. Richtárik. "Nonconvex Variance Reduced Optimization with Arbitrary Sampling". In: ().

[11] R. Johnson and T. Zhang. "Accelerating stochastic gradient descent using predictive variance reduction". In: *Advances in Neural Information Processing Systems*. 2013, pp. 315–323.

[12] J. Konečný and P. Richtárik. "Semi-stochastic gradient descent methods". In: *Frontiers in Applied Mathematics and Statistics* 3 (2017), p. 9.

[13] J. Konečný, J. Liu, P. Richtárik, and M. Takáč. "Mini-Batch Semi-Stochastic Gradient Descent in the Proximal Setting". In: *IEEE Journal of Selected Topics in Signal Processing* 2 (2016), pp. 242–255.

[14] D. Kovalev, S. Horvath, and P. Richtárik. "Don't Jump Through Hoops and Remove Those Loops: SVRG and Katyusha are Better Without the Outer Loop". In: *arXiv:1901.08689* (2019).

[15] T. Murata and T. Suzuki. "Doubly Accelerated Stochastic Variance Reduced Dual Averaging Method for Regularized Empirical Risk Minimization". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. 2017, pp. 608–617.

[16] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*. Vol. 87. 2013.

[17] Y. Nesterov and J.-P. Vial. "Confidence level solutions for stochastic programming". In: *Automatica*. Vol. 44. 2008, pp. 1559–1568.

[18] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. "SARAH: A Novel Method for Machine Learning Problems Using Stochastic Recursive Gradient". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. 2017, pp. 2613–2621.

[19] A. Nitanda. "Stochastic Proximal Gradient Descent with Acceleration Techniques". In: *Advances in Neural Information Processing Systems 27*. 2014, pp. 1574–1582.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[21] S. J. Reddi, A. Hefny, S. Sra, B. Pczos, and A. J. Smola. "Stochastic Variance Reduction for Nonconvex Optimization." In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 48. 2016, pp. 314–323.

[22] H. Robbins and D. Siegmund. "A convergence theorem for non negative almost supermartingales and some applications". In: *Herbert Robbins Selected Papers*. 1985, pp. 111–135.

[23] N. L. Roux, M. Schmidt, and F. R. Bach. "A Stochastic Gradient Method with an Exponential Convergence _Rate for Finite Training Sets". In: *Advances in Neural Information Processing Systems 25*. 2012, pp. 2663–2671.

[24] S. Shalev-Shwartz and T. Zhang. "Stochastic dual coordinate ascent methods for regularized loss minimization". In: *Journal of Machine Learning Research* 14.Feb (2013), pp. 567–599.

In Section A we present general properties that are used in our proofs. In Section B, we present the proofs for the convergence and the complexities of our algorithms. In Section D, we define several samplings. In Section E, we present the expected smoothness constant for the samplings we present. In Section F, we present the expected smoothness constant for the samplings we present.

A General properties

Lemma A.1. $\forall a, b \in \mathbb{R}^d,\ \|a+b\|_2^2 \leq 2\|a\|_2^2 + 2\|b\|_2^2$.

Lemma A.2. *For any random vector* $X \in \mathbb{R}^d, \mathbb{E}\left[\|X - \mathbb{E}[X]\|_2^2\right] = \mathbb{E}\left[\|X\|_2^2\right] - \|\mathbb{E}[X]\|_2^2 \leq \mathbb{E}\left[\|X\|_2^2\right]$.

Lemma A.3. *For any convex function* f*, we have*

$$f(y) \geq f(x) + \nabla f(x)^\top (x-y) \quad \forall x, y \in \mathbb{R}^d.$$

Lemma A.4 (Logarithm inequality)**.**

$$\log(x) \leq x - 1 \quad \forall x > 0. \tag{25}$$

Lemma A.5 (Complexity bounds)**.** *Consider the sequence* $(\alpha_k)_k \in \mathbb{R}_+$ *of positive scalars that converges to* 0 *according to*

$$\alpha_k \leq \rho^k \alpha_0,$$

where $\rho \in [0,1)$*. For a given* $\epsilon \in (0,1)$*, we have that*

$$k \geq \frac{1}{1-\rho} \log\left(\frac{1}{\epsilon}\right) \implies \alpha_k \leq \epsilon \alpha_0. \tag{26}$$

Lemma A.6. *Consider convex and* L_i*–smooth functions* f_i*, where* $L_i \geq 0$ *for all* $i \in [n]$*, and define* $L_{\max} = \max_{i \in [n]} L_i$*. Let*

$$f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x)$$

for any $x \in \mathbb{R}^d$*. Suppose that* f *is* L*–smooth, where* $L \geq 0$*. Then,*

$$nL \geq L_{\max}. \tag{27}$$

Proof. Let $x, y \in \mathbb{R}^d$. Since f is L–smooth, we have

$$f(x) \leq f(y) + \nabla f(y)^\top (x-y) + \frac{L}{2}\|x-y\|_2^2.$$

Hence, multiplying by n on both sides,

$$\sum_{i=1}^n f_i(x) \leq \sum_{i=1}^n f_i(y) + \sum_{i=1}^n \nabla f_i(y)^\top (x-y) + \frac{nL}{2}\|x-y\|_2^2.$$

Rearranging this inequality,

$$\sum_{i=1}^n \left(f_i(x) - f_i(y) - \nabla f_i(y)^\top (x-y)\right) \leq \frac{nL}{2}\|x-y\|_2^2. \tag{28}$$

Since the functions f_i are convex, we have for all $i \in [n]$,

$$f_i(x) - f_i(y) - \nabla f_i(y)^\top (x-y) \geq 0.$$

Then, as a consequence of (28), we have that for all $i \in [n]$,

$$f_i(x) - f_i(y) - \nabla f_i(y)^\top (x-y) \leq \frac{nL}{2}\|x-y\|_2^2.$$

Rearranging this inequality,

$$f_i(x) \leq f_i(y) + \nabla f_i(y)^\top (x-y) + \frac{nL}{2}\|x-y\|_2^2.$$

But since for all $i \in [n]$, L_i is the smallest positive constant that verifies

$$f_i(x) \leq f_i(y) + \nabla f_i(y)^\top (x-y) + \frac{L_i}{2}\|x-y\|_2^2,$$

we have for all $i \in [n], L_i \leq nL$. Hence $L_{\max} \leq nL$. □

B Proofs of the results of the main paper

In this section, we will use the abreviations $\mathbb{E}_t[X] \overset{\text{def}}{=} \mathbb{E}\left[X|x^t,\ldots,x^1\right]$ for any random variable $X \in \mathbb{R}^d$ and iterates $(x^t)_{t\geq 0}$.

B.1 Proof of Lemma 4.4

Proof.

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[\|g(x,w)\|_2^2\right] &= \mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(x)-\nabla f_v(x^*)+\nabla f_v(x^*)-\nabla f_v(w)+\nabla f(w)\|_2^2\right]\\ &\overset{\text{Lem. } A.1}{\leq} 2\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(x)-\nabla f_v(x^*)\|_2^2\right]\\ &\quad +2\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(w)-\nabla f_v(x^*)-\nabla f(w)\|_2^2\right].\end{aligned}$$

Since $\mathbb{E}_{\mathcal{D}}\left[\nabla f_v(w)-\nabla f_v(x^*)\right]=\nabla f(w)$, we have

$$\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(w)-\nabla f_v(x^*)-\nabla f(w)\|_2^2\right] \overset{\text{Lem.}A.2}{\leq} \mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(w)-\nabla f_v(x^*)\|_2^2\right].$$

Hence,

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[\|g(x,w)\|_2^2\right] &\leq 2\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(x)-\nabla f_v(x^*)\|_2^2\right]+2\mathbb{E}_{\mathcal{D}}\left[\|\nabla f_v(w)-\nabla f_v(x^*)\|_2^2\right]\\ &\overset{(11)+(12)}{\leq} 4\mathcal{L}(f(x)-f(x^*))+4\rho(f(w)-f(x^*)).\end{aligned}$$

□

B.2 Proof of Theorem 4.1

Proof.

$$\begin{aligned}\mathbb{E}_t\left[\|x_s^{t+1}-x^*\|_2^2\right] &= \mathbb{E}_t\left[\|x_s^t-x^*-\alpha g_s^t\|_2^2\right]\\ &= \|x_s^t-x^*\|_2^2-2\alpha\mathbb{E}_t\left[g_s^t\right]^\top(x_s^t-x^*)+\alpha^2\mathbb{E}_t\left[\|g_s^t\|_2^2\right]\\ &\overset{(8)+(15)}{\leq} \|x_s^t-x^*\|_2^2-2\alpha\nabla f(x_s^t)^\top(x_s^t-x^*)\\ &\quad +2\alpha^2\left[2\mathcal{L}(f(x_s^t)-f(x^*))+2\mathcal{L}(f(w_{s-1})-f(x^*))\right]\\ &\overset{(6)}{\leq} (1-\alpha\mu)\|x_s^t-x^*\|_2^2-2\alpha(1-2\alpha\mathcal{L})\left(f(x_s^t)-f(x^*)\right)\\ &\quad +4\alpha^2\rho(f(w_{s-1})-f(x^*)).\end{aligned} \tag{29}$$

Note that since $\alpha \leq \frac{1}{2(\mathcal{L}+2\rho)}$ and $\rho \geq 0$, we have that

$$\alpha \overset{\text{Lemma E.3}}{\leq} \frac{1}{2\mu},$$

and consequently $(1-\alpha\mu) > 0$. Thus by iterating (29) over $t = 0, \ldots, m-1$ and taking the expectation, since $x_s^0 = x_{s-1}^m$, we obtain

$$\begin{aligned}
\mathbb{E}\left[\|x_s^m - x^*\|_2^2\right] &\leq (1-\alpha\mu)^m \mathbb{E}\left[\|x_{s-1}^m - x^*\|_2^2\right] \\
&\quad -2\alpha(1-2\alpha\mathcal{L}) \sum_{t=0}^{m-1} (1-\alpha\mu)^{m-1-t} \mathbb{E}\left[f(x_s^t) - f(x^*)\right] \\
&\quad +4\alpha^2\rho\mathbb{E}\left[f(w_{s-1}) - f(x^*)\right] \sum_{t=0}^{m-1} (1-\alpha\mu)^{m-1-t} \\
&\overset{(10)}{=} (1-\alpha\mu)^m \mathbb{E}\left[\|x_{s-1}^m - x^*\|_2^2\right] - 2\alpha(1-2\alpha\mathcal{L})S_m \sum_{t=0}^{m-1} p_t \mathbb{E}\left[f(x_s^t) - f(x^*)\right] \\
&\quad +4\alpha^2\rho S_m \mathbb{E}\left[f(w_{s-1}) - f(x^*)\right] \\
&\overset{(16)}{=} (1-\alpha\mu)^m \mathbb{E}\left[\|x_{s-1}^m - x^*\|_2^2\right] - 2\alpha(1-2\alpha\mathcal{L})S_m \sum_{t=0}^{m-1} p_t \mathbb{E}\left[f(x_s^t) - f(x^*)\right] \\
&\quad +\frac{1}{2}\mathbb{E}\left[\psi_{s-1}\right]. \qquad (30)
\end{aligned}$$

Since f is convex, we have by Jensen's inequality that

$$\begin{aligned}
f(w_s) - f(x^*) &= f(\sum_{t=0}^{m-1} p_t x_s^t) - f(x^*) \\
&\leq \sum_{t=0}^{m-1} p_t (f(x_s^t) - f(x^*)). \qquad (31)
\end{aligned}$$

Consequently,

$$\mathbb{E}\left[\psi_s\right] \overset{(16)+(31)}{\leq} 8\alpha^2\rho S_m \sum_{t=0}^{m-1} p_t \mathbb{E}\left[(f(x_s^t) - f(x^*))\right]. \qquad (32)$$

As a result,

$$\begin{aligned}
\mathbb{E}\left[\phi_s\right] \overset{(30)+(32)}{\leq} &(1-\alpha\mu)^m \mathbb{E}\left[\|x_{s-1}^m - x^*\|_2^2\right] + \frac{1}{2}\mathbb{E}\left[\psi_{s-1}\right] \\
&-2\alpha(1-2\alpha(\mathcal{L}+2\rho))S_m \sum_{t=0}^{m-1} p_t \mathbb{E}\left[(f(x_s^t) - f(x^*))\right].
\end{aligned}$$

Since $\alpha \leq \frac{1}{6\mathcal{L}}$, the above implies

$$\begin{aligned}
\mathbb{E}\left[\phi_s\right] &\leq (1-\alpha\mu)^m \mathbb{E}\left[\|x_{s-1}^m - x^*\|_2^2\right] + \frac{1}{2}\mathbb{E}\left[\psi_{s-1}\right] \\
&\leq \beta\mathbb{E}\left[\phi_{s-1}\right],
\end{aligned}$$

where $\beta = \max\{(1-\alpha\mu)^m, \frac{1}{2}\}$.

Moreover, if we set $w_s = x_s^t$ with probability p_t, for $t = 0, \ldots, m-1$, the result would still hold. Indeed (31) would hold with equality and the rest of the proof would follow verbatim. □

B.3 Proof of Corollary 4.1

Proof. Noting $\beta = \max\left\{\left(1 - \frac{\mu}{2(\mathcal{L}(b)+2\rho(b))}\right)^m, \frac{1}{2}\right\}$, we need to chose s so that $\beta^s \leq \epsilon$, that is $s \geq \frac{\log(1/\epsilon)}{\log(1/\beta)}$. Since in each inner iteration we evaluate $2b$ of the f_i gradients, and in each outer

iteration we evaluate all n gradients, this means that the total complexity will be given by

$$\begin{aligned} C &\overset{\text{def}}{=} (n+2bm)\frac{\log(1/\epsilon)}{\log(1/\beta)} \\ &= \max\{-\frac{n+2bm}{m\log(1-\frac{\mu}{2(\mathcal{L}(b)+2\rho(b))})}, \frac{n+2bm}{\log 2}\}\log\left(\frac{1}{\epsilon}\right) \\ &\overset{(25)}{\leq} \max\{\frac{n+2bm}{m}\frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}, 2(n+2bm)\}\log\left(\frac{1}{\epsilon}\right). \end{aligned}$$

□

B.4 Proof of Corollary 4.3

Proof. Recall that from (18), using the fact that $\mathcal{L}(1) = \rho(1) = L_{\max}$, we have

$$C_m(1) = 2\left(\frac{n}{m}+2\right)\max\{\frac{3L_{\max}}{\mu}, m\}\log\left(\frac{1}{\epsilon}\right).$$

When $n \geq \frac{L_{\max}}{\mu}$. Then, $m \in \left[\frac{L_{\max}}{\mu}, n\right]$. We can rewrite $C_m(1)$ as

$$C_m(1) = 2(n+2m)\max\{\frac{1}{m}\frac{3L_{\max}}{\mu}, 1\}\log\left(\frac{1}{\epsilon}\right).$$

We have $\frac{1}{m}\frac{3L_{\max}}{\mu} \leq 3$ and $n+2m \leq 3n$. Hence,

$$C_m(1) \leq 18n\log\left(\frac{1}{\epsilon}\right) = O\left(\left(n+\frac{L_{\max}}{\mu}\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

When $n \leq \frac{L_{\max}}{\mu}$. Then, $m \in \left[n, \frac{L_{\max}}{\mu}\right]$. We have $\frac{n}{m} \leq 1$ and $m \leq \frac{3L_{\max}}{\mu}$. Hence,

$$C_m(1) \leq \frac{18L_{\max}}{\mu}\log\left(\frac{1}{\epsilon}\right) = O\left(\left(n+\frac{L_{\max}}{\mu}\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

□

B.5 Proof of Theorem 5.1

Before analyzing Algorithm 2, we present a lemma that allows to compute the expectations $\mathbb{E}[\alpha_k]$ and $\mathbb{E}\left[\alpha_k^2\right]$, that will be used in the analysis.

Lemma B.1. *Consider the step sizes defined by Algorithm 2. We have*

$$\mathbb{E}[\alpha_k] = \frac{(1-p)^{\frac{3k+2}{2}}(1-\sqrt{1-p})+p}{1-(1-p)^{\frac{3}{2}}}\alpha. \tag{33}$$

$$\mathbb{E}\left[\alpha_k^2\right] = \frac{1+(1-p)^{2k+1}}{2-p}\alpha^2. \tag{34}$$

Proof. Taking expectation with respect to the filtration induced by the sequence of step sizes $\{\alpha_1, \ldots, \alpha_k\}$

$$\mathbb{E}_p[\alpha_{k+1}] = (1-p)\sqrt{1-p}\,\alpha_k + p\alpha. \tag{35}$$

Then taking total expectation

$$\mathbb{E}[\alpha_{k+1}] = (1-p)\sqrt{1-p}\mathbb{E}[\alpha_k] + p\alpha. \tag{36}$$

Hence the sequence $(\mathbb{E}[\alpha_k])_{k\geq 1}$ is uniquely defined by

$$\mathbb{E}[\alpha_k] = \frac{(1-p)^{\frac{3k+2}{2}}(1-\sqrt{1-p})+p}{1-(1-p)^{\frac{3}{2}}}\alpha. \tag{37}$$

Indeed, applying (36) recursively gives

$$\mathbb{E}\left[\alpha_k\right] = (1-p)^{\frac{3k}{2}}\alpha + p\alpha\sum_{i=0}^{k-1}(1-p)^{\frac{3i}{2}}.$$

Adding up the geometric series gives

$$\begin{aligned}\mathbb{E}\left[\alpha_k\right] &= \alpha(1-p)^{\frac{3k}{2}} + p\alpha\frac{1-(1-p)^{\frac{3k}{2}}}{1-(1-p)^{\frac{3}{2}}} \\ &= \frac{(1-p)^{\frac{3k}{2}}(1-(1-p)^{\frac{3}{2}}) - (1-p)^{\frac{3k}{2}}p + p}{1-(1-p)^{\frac{3}{2}}}\alpha\ .\end{aligned}$$

Which leads to (37) by factorizing. The same arguments are used to compute $\mathbb{E}\left[\alpha_k^2\right]$. □

We now present a proof of Theorem 5.1.

Proof of Theorem 5.1. First,

$$\begin{aligned}\mathbb{E}_k\left[\|x^{k+1}-x^*\|_2^2\right] &= \mathbb{E}_k\left[\|x^k - x^* - \alpha_k g^k\|_2^2\right] \\ &= \|x^k-x^*\|_2^2 - 2\alpha_k\mathbb{E}_k\left[g^k\right]^\top(x^k-x^*) + \alpha_k^2\mathbb{E}_k\left[\|g^k\|_2^2\right] \\ &\overset{(8)+(15)}{\leq} \|x^k-x^*\|_2^2 - 2\alpha_k\nabla f(x^k)^\top(x^k-x^*) \\ &\quad +2\alpha_k^2\left[2\mathcal{L}(f(x^k)-f(x^*)) + 2\mathcal{L}(f(w^k)-f(x^*))\right] \\ &\overset{(6)}{\leq} (1-\alpha_k\mu)\|x^k-x^*\|_2^2 - 2\alpha_k(1-2\alpha_k\mathcal{L})\left(f(x^k)-f(x^*)\right) \\ &\quad +4\alpha_k^2\mathcal{L}(f(w^k)-f(x^*)) \\ &\overset{(19)}{=} (1-\alpha_k\mu)\|x^k-x^*\|_2^2 - 2\alpha_k(1-2\alpha_k\mathcal{L})\left(f(x^k)-f(x^*)\right) \\ &\quad +p\left(\frac{3}{2}-p\right)\psi^k.\end{aligned}$$

Hence we have, taking total expectation and noticing that the variables α_k and x^k are independent,

$$\begin{aligned}\mathbb{E}\left[\left\|x^{k+1}-x^*\right\|_2^2\right] &\leq (1-\mathbb{E}\left[\alpha_k\right]\mu)\,\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right] - 2\mathbb{E}\left[\alpha_k(1-2\alpha_k\mathcal{L})\right]\mathbb{E}\left[f(x^k)-f(x^*)\right] \\ &\quad +p\left(\frac{3}{2}-p\right)\mathbb{E}\left[\psi^k\right]. \qquad (38)\end{aligned}$$

We have also have

$$\begin{aligned}\mathbf{E}_k\left[\psi^{k+1}\right] &= (1-p)\frac{8(1-p)\alpha_k^2\mathcal{L}}{p(3-2p)}\left(f(w^k)-f(x^*)\right) + p\frac{8\alpha^2\mathcal{L}}{p(3-2p)}\left(f(x^k)-f(x^*)\right) \\ &= (1-p)^2\psi^k + \frac{8\alpha^2\mathcal{L}}{3-2p}\left(f(x^k)-f(x^*)\right).\end{aligned}$$

Hence, taking total expectation gives

$$\mathbb{E}\left[\psi^{k+1}\right] = (1-p)^2\mathbb{E}\left[\psi^k\right] + \frac{8\alpha^2\mathcal{L}}{3-2p}\mathbb{E}\left[f(x^k)-f(x^*)\right] \qquad (39)$$

Consequently,

$$\begin{aligned}\mathbb{E}\left[\phi^k\right] &\overset{(38)+(39)}{\leq} & (1-\mathbb{E}\left[\alpha_k\right]\mu)\,\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right] \\ & & -2\left(\mathbb{E}\left[\alpha_k(1-2\alpha_k\mathcal{L})\right]-4\frac{\alpha^2\mathcal{L}}{3-2p}\right)\mathbb{E}\left[f(x^k)-f(x^*)\right] \\ & & +\left(1-\frac{p}{2}\right)\mathbb{E}\left[\psi^k\right] \\ &= & (1-\mathbb{E}\left[\alpha_k\right]\mu)\,\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right] \\ & & -2\left(\mathbb{E}\left[\alpha_k\right]-2\left(\mathbb{E}\left[\alpha_k^2\right]+\frac{2}{3-2p}\alpha^2\right)\mathcal{L}\right)\mathbb{E}\left[f(x^k)-f(x^*)\right] \\ & & +\left(1-\frac{p}{2}\right)\mathbb{E}\left[\psi^k\right]. \end{aligned} \tag{40}$$

From Lemma B.1, we have $\mathbb{E}\left[\alpha_k\right]=\frac{(1-p)^{\frac{3k+2}{2}}(1-\sqrt{1-p})+p}{1-(1-p)^{\frac{3}{2}}}\alpha$, and we can show that for all k

$$\mathbb{E}\left[\alpha_k\right]\geq\frac{2}{3}\alpha, \tag{41}$$

Letting $q=1-p$ we have that

$$\begin{aligned}\frac{(1-p)^{\frac{3k+2}{2}}(1-\sqrt{1-p})+p}{1-(1-p)^{\frac{3}{2}}} &= \frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{1-q^{\frac{3}{2}}} \\ &= q^{\frac{3k+2}{2}}\frac{1-\sqrt{q}}{1-q^{3/2}}+\frac{1-q}{1-q^{3/2}}\geq\frac{1-q}{1-q^{3/2}} \quad \forall q\geq 0.\end{aligned}$$

Consequently,

$$\begin{aligned}\mathbb{E}\left[\phi^k\right] &\overset{(38)+(39)+(41)}{\leq} & \left(1-\frac{2}{3}\alpha\mu\right)\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right] \\ & & -2\left(\mathbb{E}\left[\alpha_k\right]-2\left(\mathbb{E}\left[\alpha_k^2\right]+\frac{2}{3-2p}\alpha^2\right)\mathcal{L}\right)\mathbb{E}\left[f(x^k)-f(x^*)\right] \\ & & +\left(1-\frac{p}{2}\right)\mathbb{E}\left[\psi^k\right]. \end{aligned} \tag{42}$$

To declutter the notations, Let us define

$$a_k \overset{\text{def}}{=} \frac{(1-p)^{\frac{3k+2}{2}}(1-\sqrt{1-p})+p}{1-(1-p)^{\frac{3}{2}}} \tag{43}$$

$$b_k \overset{\text{def}}{=} \frac{1+(1-p)^{2k+1}}{2-p} \tag{44}$$

so that $\mathbb{E}\left[\alpha_k\right]=a_k\alpha$ and $\mathbb{E}\left[\alpha_k^2\right]=b_k\alpha^2$. Then (42) becomes

$$\begin{aligned}\mathbb{E}\left[\phi^k\right] &\leq & \left(1-\frac{2}{3}\alpha\mu\right)\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right] \\ & & -2\alpha\left(a_k-2\alpha\left(b_k+\frac{2}{3-2p}\right)\mathcal{L}\right)\mathbb{E}\left[f(x^k)-f(x^*)\right] \qquad (45) \\ & & +\left(1-\frac{p}{2}\right)\mathbb{E}\left[\psi^k\right]. \qquad (46)\end{aligned}$$

Next we would like to drop the term (45). For this we need to guarantee that

$$a_k-2\alpha\mathcal{L}\left(b_k+\frac{2}{3-2p}\right)\geq 0$$

Let $q=1-p$ so that the above becomes

$$\frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{1-q^{\frac{3}{2}}}-2\alpha\mathcal{L}\left(\frac{1+q^{2k+1}}{1+q}+\frac{2}{1+2q}\right)\geq 0.$$

In other words, after dividing through by $\left(\frac{1+q^{2k+1}}{1+q}+\frac{2}{1+2q}\right)$ and re-arranging, we require that

$$\begin{aligned} 2\mathcal{L}\alpha &\leq \frac{\frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{1-q^{\frac{3}{2}}}}{\frac{1+q^{2k+1}}{1+q}+\frac{2}{1+2q}} = \frac{\frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{1-q^{\frac{3}{2}}}}{\frac{(1+q^{2k+1})(1+2q)+2(1+q)}{(1+q)(1+2q)}} \\ &= \frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{q^{2k+1}(1+2q)+3+4q}\frac{(1+q)(1+2q)}{1-q^{\frac{3}{2}}}. \end{aligned} \tag{47}$$

We are now going to show that:

$$\frac{q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q}{q^{2k+1}(1+2q)+3+4q} \geq \frac{1-q}{3+4q}. \tag{48}$$

Indeed, multiplying out the denominators of the above gives

$$\begin{aligned} F(q) &\overset{\text{def}}{=} (3+4q)\left(q^{\frac{3k+2}{2}}(1-\sqrt{q})+1-q\right)-(1-q)\left(q^{2k+1}(1+2q)+3+4q\right) \\ &= q^{\frac{3k+2}{2}}(1-\sqrt{q})(3+4q)-q^{2k+1}(1+2q)(1-q) \\ &= q^{\frac{3k+2}{2}}(1-\sqrt{q})\left(3+4q-q^{\frac{k}{2}}(1+2q)(1+\sqrt{q})\right). \end{aligned}$$

And since $q^{\frac{k}{2}} \leq 1$, we have

$$\begin{aligned} F(q) &\geq q^{\frac{3k+2}{2}}(1-\sqrt{q})\left(3+4q-(1+2q)(1+\sqrt{q})\right) \\ &= 2q^{\frac{3k+2}{2}}(1-\sqrt{q})(1-q\sqrt{q}) \\ &\geq 0. \end{aligned}$$

As a result (48) holds, and thus if

$$2\mathcal{L}\alpha \leq \frac{1-q}{3+4q}\frac{(1+q)(1+2q)}{1-q^{\frac{3}{2}}}$$

holds, then Equation (47) is verified for all k. This is why we impose the upperbound on the stepsize given in (20). Finally since (20) ensures that $a_k - 2\alpha(b_k+\frac{2}{3-2p})\mathcal{L} \geq 0$ and from (46) we have that

$$\begin{aligned} \mathbb{E}\left[\phi^{k+1}\right] &\overset{(38)+(39)}{\leq} \left(1-\frac{2}{3}\alpha\mu\right)\mathbb{E}\left[\left\|x^k-x^*\right\|_2^2\right]+\left(1-\frac{p}{2}\right)\mathbb{E}\left[\psi^k\right] \\ &\leq \beta\mathbb{E}\left[\phi^{k+1}\right], \end{aligned} \tag{49}$$

where $\beta = \max\{1-\frac{2}{3}\alpha\mu, 1-\frac{p}{2}\}$. □

B.6 Proof of Corollary 5.1

Proof. We have that

$$\mathbb{E}\left[\phi^k\right] \leq \beta^k\phi^0,$$

where $\beta = \max\left\{1-\frac{1}{3\zeta_p}\frac{\mathcal{L}(b)}{\mu}, 1-\frac{p}{2}\right\}$. Hence using Lemma A.5, we have that the iteration complexity for an $\epsilon > 0$ approximate solution that verifies $\mathbb{E}\left[\phi^k\right] \leq \epsilon\phi^0$ is

$$2\max\left\{\frac{3\zeta_p}{2}\frac{\mathcal{L}(b)}{\mu}, \frac{1}{p}\right\}\log\left(\frac{1}{\epsilon}\right).$$

For the total complexity, one can notice that in expectation, we compute $2b+pn$ stochastic gradients at each iteration. □

B.7 Proof of Proposition 6.1

Proof. Dropping the $\log(1/\epsilon)$ for brevity, we distinguish two cases, $m \geq \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$ and $m \leq \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$.

1. $m \geq \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$: Then $C_m(b) = 2(n+2bm)$, and hence we should use the smallest m possible, that is, $m = \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$.
2. $m \leq \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$: Then $C_m(b) = \frac{2(n+2bm)}{m}\frac{2(\mathcal{L}(b)+2\rho(b))}{\mu} = 2\left(\frac{n}{m}+2b\right)\frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$. Hence $C_m(b)$ is decreasing in m and we should then use the highest possible value for m, that is $m = \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$.

The result now follows by substituting $m = \frac{2(\mathcal{L}(b)+2\rho(b))}{\mu}$ into (18). □

B.8 Proof of Proposition 6.2

Proof. Recall that have from Lemma 4.3:

$$\mathcal{L}(b) = \frac{1}{b}\frac{n-b}{n-1}L_{\max} + \frac{n}{b}\frac{b-1}{n-1}L, \tag{50}$$

$$\rho(b) = \frac{1}{b}\frac{n-b}{n-1}L_{\max}. \tag{51}$$

For brievity, we temporarily drop the term $\log\left(\frac{1}{\epsilon}\right)$ in $C_m(b)$ defined in Equation (18). Hence, we want to find, for different values of m:

$$b^* = \arg\min_{b\in[n]} C_m(b) := 2\left(\frac{n}{m}+2b\right)\max\{\kappa(b), m\}, \tag{52}$$

where $\kappa(b) \overset{\text{def}}{=} \frac{\mathcal{L}(b)+2\rho(b)}{\mu}$.

When $m = n$. In this case we have

$$C_n(b) \overset{(18)}{=} 2(2b+1)\max\{\kappa(b), n\}, \tag{53}$$

Writing $\kappa(b)$ explicitly:

$$\kappa(b) = \frac{1}{\mu(n-1)}\left((3L_{\max}-L)\frac{n}{b} + nL - 3L_{\max}\right).$$

Since $3L_{\max} > L$, $\kappa(b)$ is a decreasing function of b. In the light of this observation, we will determine the optimal mini-batch size. The upcoming analysis is summarized in Table 1.

We distinguish three cases:

- If $n \leq \frac{L}{\mu}$: then $\kappa(n) = \frac{L}{\mu} \geq n$. Since $\kappa(b)$ is decreasing, this means that for all $b \in [n], \kappa(b) \geq n$. Consequently, $C_n(b) = 2(2b+1)\kappa(b)$. Differentiating twice:

  $$C_n^{''}(b) = \frac{4}{\mu(n-1)}\frac{(3L_{\max}-L)n}{b^3} > 0.$$

  Hence $C_n(b)$ is a convex function. Now examining its first derivative:

  $$C_n^{'}(b) = \frac{2}{\mu(n-1)}\left(-\frac{(3L_{\max}-L)n}{b^2} + 2(nL-3L_{\max})\right),$$

  we can see that:

  – If $n \leq \frac{3L_{\max}}{L}$, $C_n(b)$ is a decreasing function, hence

  $$b^* = n.$$

– If $n > \frac{3L_{\max}}{L}$, $C_n(b)$ admits a minimizer, which we can find by setting its first derivative to zero. The solution is

$$\hat{b} \overset{\text{def}}{=} \sqrt{\frac{n}{2}\frac{3L_{\max} - L}{nL - 3L_{\max}}}.$$

Hence,

$$b^* = \left\lfloor \hat{b} \right\rfloor$$

- If $n \geq \frac{3L_{\max}}{\mu}$, then $\kappa(1) = 3\frac{L_{\max}}{\mu}$. Since $\kappa(b)$ is decreasing, this means that for all $b \in [n], \kappa(b) \leq n$. Hence, $C_n(b) = 2(2b+1)n$. $C_n(b)$ is an increasing function of b. Therefore,

$$b^* = 1.$$

- If $\frac{L}{\mu} < n < \frac{3L_{\max}}{\mu}$, we have $\kappa(1) > n$ and $\kappa(n) < n$. Hence there exists $\tilde{b} \in [1,\, n]$ such that $\kappa(b) = n$, and it is given by

$$\tilde{b} \overset{\text{def}}{=} \frac{(3L_{\max} - L)n}{n(n-1)\mu - nL + 3L_{\max}}. \tag{54}$$

Define $G(b) \overset{\text{def}}{=} (2b+1)\kappa(b)$. Then,

$$\arg\min_{b \in [1,\, n]} G(b) = \begin{cases} n & \text{if } n \leq \frac{3L_{\max}}{L}, \\ \hat{b} & \text{if } n > \frac{3L_{\max}}{L}. \end{cases} \tag{55}$$

As a result, we have that

– if $n \leq \frac{3L_{\max}}{L}$, $G(b)$ is decreasing on $[1, n]$, hence $C_n(b)$ is decreasing on $[1, \tilde{b}]$ and increasing on $[\tilde{b}, n]$. Then,

$$b^* = \left\lfloor \tilde{b} \right\rfloor.$$

– if $n > \frac{3L_{\max}}{L}$, $G(b)$ is decreasing on $[1, \hat{b}]$ and increasing on $[\hat{b}, n]$. Hence $C_n(b)$ is decreasing on $[1, \min\{\hat{b}, \tilde{b}\}]$ and increasing on $[\min\{\hat{b}, \tilde{b}\}, 1]$. Then,

$$b^* = \left\lfloor \min\{\hat{b}, \tilde{b}\} \right\rfloor.$$

To summarize, we have for $m = n$,

$$b^* \quad = \quad \begin{cases} 1 & \text{if } n \geq \frac{3L_{\max}}{\mu} \\ \left\lfloor \min(\tilde{b}, \hat{b}) \right\rfloor & \text{if } \max\{\frac{L}{\mu}, \frac{3L_{\max}}{L}\} < n < \frac{3L_{\max}}{\mu} \\ \left\lfloor \hat{b} \right\rfloor & \text{if } \frac{3L_{\max}}{L} < n < \frac{L}{\mu} \\ \left\lfloor \tilde{b} \right\rfloor & \text{if } \frac{L}{\mu} < n \leq \frac{3L_{\max}}{L} \\ n & \text{otherwise, if } n \leq \min\{\frac{L}{\mu}, \frac{3L_{\max}}{L}\} \end{cases} \tag{56}$$

When $m = n/b$. In this case we have

$$C_m(b) \overset{(18)}{=} 6\max\{b\kappa(b), n\},$$

with

$$b\kappa(b) = \frac{1}{\mu(n-1)}\left((3L_{\max} - L)n + (nL - 3L_{\max})b\right),$$

and thus $\kappa(1) = \frac{3L_{\max}}{\mu}$ and $n\kappa(n) = \frac{nL}{\mu} \geq n$. We distinguish two cases:

- if $n \leq \frac{3L_{\max}}{L}$, then $b\kappa(b)$ is decreasing in b. Since $n\kappa(n) \geq n$, $C_m(b) = 6b\kappa(b)$, thus $C_m(b)$ is decreasing in b, hence

$$b^* = n$$

- if $n > \frac{3L_{\max}}{L}$, $b\kappa(b)$ is increasing in b. Thus,
  - if $n \leq \frac{3L_{\max}}{\mu} = \kappa(1)$, then $C_m(b) = 6b\kappa(b)$. Hence $b^* = 1$.
  - if $n > \frac{3L_{\max}}{\mu}$, using the definition of $\tilde{b}$ in Equation (54), we have that

$$C_m(b) = \begin{cases} 6n & \text{for } b \in [1, \bar{b}] \\ 6b\kappa(b) & \text{for } b \in [\bar{b}, n] \end{cases},$$

where

$$\bar{b} = \frac{n(n-1)\mu - (3L_{\max} - L)n}{nL - 3L_{\max}}$$

is the batch size $b \in [n]$ which verifies $b\kappa(b) = n$. Hence b^* can be any point in $\{1, \ldots, \lfloor \bar{b} \rfloor\}$. In light of shared memory parallelism, $b^* = \lfloor \bar{b} \rfloor$ would be the most practical choice.

□

C Optimal mini-batch size for Algorithm 2

By using a similar proof as in Section B.8, we derive the following result.

Proposition C.1. *Note* $b^* \overset{def}{=} \arg\min_{b \in [n]} C_p(b)$*, where* $C_p(b)$ *is defined in* (22)*. For the widely used choice* $p = \frac{1}{n}$*, we have that*

$$b^* = \begin{cases} 1 & \text{if } n \geq \frac{3\zeta_{1/n}}{2} \frac{L_{\max}}{\mu} \\ \left\lfloor \min(\tilde{b}, \hat{b}) \right\rfloor & \text{if } \frac{3\zeta_{1/n}}{2} \frac{L}{\mu} < n < \frac{3\zeta_{1/n}}{2} \frac{L_{\max}}{\mu} \\ n & \text{otherwise, if } n \leq \frac{3\zeta_{1/n}}{2} \frac{L}{\mu} \end{cases}, \tag{57}$$

where ζ_p *is defined in* (20) *for* $p \in (0, 1]$.

Because ζ_p depends on p, optimizing the total complexity with respect to b for the case $p = \frac{b}{n}$ is extremely cumbersome. Thus, we restrain our study for the optimal mini-batch sizes for Algorithm 2 to the case where $p = \frac{1}{n}$.

Proof. For brievity, we temporarily drop the term $\log\left(\frac{1}{\epsilon}\right)$ in $C_p(b)$ defined in Equation (22). Hence, we want to find, for different values of m:

$$b^* = \arg\min_{b \in [n]} C_{1/n}(b) := 2\,(2b+1)\max\{\pi(b), m\}, \tag{58}$$

where $\pi(b) \overset{\text{def}}{=} \frac{3\xi_p}{2} \frac{\mathcal{L}(b)}{\mu}$. We have

$$\pi(b) = \frac{3\xi_p}{2} \frac{1}{\mu(n-1)} \left(\frac{n(L_{\max} - L)}{b} + nL - L_{\max} \right). \tag{59}$$

Since $L_{\max} \geq L$, $\pi(b)$ is a decreasing function on $[1, n]$. We distinguish three cases:

- if $n > \kappa(1) = \frac{3\xi_p}{2} \frac{L_{\max}}{\mu}$, then for all $b \in [1, n]$, $n > \kappa(b)$. Hence,

$$C_n(b) = 2(2b+1)n.$$

  $C_{1/n}(b)$ is an increasing function of b. Hence

$$b^* = 1.$$

- if $n < \pi(n) = \frac{3\xi_p}{2} \frac{L}{\mu}$, then for all $b \in [1, n]$, $n < \pi(b)$. Hence,

$$C_{1/n}(b) = \pi(b).$$

  Since $\pi(b)$ is a decreasing function of b,

$$b^* = n.$$

- if $\frac{3\xi_p}{2}\frac{L}{\mu} = \pi(n) \leq n \leq \pi(1) = \frac{3\xi_p}{2}\frac{L_{\max}}{\mu}$. Then there exists $b \in [1, n]$ such that $\pi(b) = n$ and its expression is given by

$$\tilde{b} = \frac{\frac{3\xi_p}{2}n(L_{\max} - L)}{\mu n(n-1) - \frac{3\xi_p}{2}(nL - L_{\max})}.$$

Now, consider the function

$$\begin{aligned} G(b) &\overset{\text{def}}{=} (2b+1)\pi(b) \\ &= \frac{3\xi_p}{2}\frac{1}{\mu(n-1)}\left(2(nL - L_{\max})b + \frac{n(L_{\max} - L)}{b}\right) + \Omega, \end{aligned}$$

where Ω replaces constants which don't depend on b. The first derivative of $G(b)$ is

$$G'(b) = \frac{3\xi_p}{2}\frac{1}{\mu(n-1)}\left(-\frac{n(L_{\max} - L)}{b^2} + 2(nL - L_{\max})\right),$$

and its second derivative is

$$G''(b) = \frac{3\xi_p n(L_{\max} - L)}{\mu(n-1)b^3} \geq 0.$$

$G(b)$ is a convex function, and we can find its minimizer by setting its first derivative to zero. This minimizer is

$$\hat{b} = \sqrt{\frac{n}{2}\frac{L_{\max} - L}{nL - L_{\max}}}.$$

Indeed, recall that from Lemma A.6, we have $nL \geq L_{\max}$. Consequently, the function $C_n(b)$ is decreasing on $\left[1, \min\left\{\tilde{b}, \hat{b}\right\}\right]$ and increasing on $\left[\min\left\{\tilde{b}, \hat{b}\right\}, n\right]$. Hence,

$$b^* = \left\lfloor \min\left\{\tilde{b}, \hat{b}\right\} \right\rfloor.$$

□

D Samplings

In Definition 3.3, we defined b–nice sampling. For completeness, we present here some other interesting possible samplings.

Definition D.1 (single-element sampling)**.** *Given a set of probabilities $(p_i)_{i\in[n]}$, S is a single-element sampling if $\mathbb{P}(|S| = 1) = 1$ and*

$$\mathbb{P}(S = \{i\}) = p_i \quad \forall i \in [n].$$

Definition D.2 (partition sampling)**.** *Given a partition $\mathcal{B}$ of $[n]$, S is a partition sampling if*

$$p_B \overset{\text{def}}{=} \mathbb{P}(S = B) > 0 \quad \forall B \in \mathcal{B}, \text{ and } \sum_{B\in\mathcal{B}} p_B = 1.$$

Definition D.3 (independent sampling)**.** *S is an independent sampling if it includes every i independently with probability $p_i > 0$.*

In Section E, we will determine for each of these samplings their corresponding expected smoothness constant.

E Expected Smoothness

First, we present two general properties about the expected smoothness constant (4.1): we establish its existence, and we prove that it is always greater than the strong convexity constant. Then, we determine the expected smoothness constant for particular samplings.

E.1 General properties of the expected smoothness constant

The following lemma is an adaptation of Theorem 3.6 in [7]. It establishes the existence of the expected smoothness constant as a result of the smoothness of the functions $f_i, i \in [n]$.

Lemma E.1 (Theorem 3.6 in [7])**.** *Let v be an unbiased sampling vector with $v_i \geq 0$ with probability one. Suppose that $f_v(w) = \frac{1}{n}\sum_{i=1}^n f_i(w)v_i$ is L_v–smooth and convex. It follows that the expected smoothness constant* (4.1) *is given by*

$$\mathcal{L} = \max_{i\in[n]} \mathbb{E}\left[L_v v_i\right].$$

Proof. Since the f_i's are convex, each realization of f_v is convex, and it follows from equation 2.1.7 in [16] that

$$\|\nabla f_v(x) - \nabla f_v(y)\|_2^2 \quad \leq \quad 2L_v\left(f_v(x) - f_v(y) - \langle \nabla f_v(y), x - y\rangle\right). \tag{60}$$

Taking expectation over the sampling gives

$$\begin{aligned}
\mathbb{E}\left[\|\nabla f_v(x) - \nabla f_v(x^*)\|_2^2\right] &\leq 2\mathbb{E}\left[L_v\left(f_v(x) - f_v(x^*) - \langle \nabla f_v(x^*), x - x^*\rangle\right)\right] \\
&\overset{(60)}{=} \frac{2}{n}\mathbb{E}\left[\sum_{i=1}^n L_v v_i\left(f_i(x) - f_i(x^*) - \langle \nabla f_i(x^*), x - x^*\rangle\right)\right] \\
&= \frac{2}{n}\sum_{i=1}^n \mathbb{E}\left[L_v v_i\right]\left(f_i(x) - f_i(y) - \langle \nabla f_i(x^*), x - x^*\rangle\right) \\
&\leq 2\max_{i=1,\ldots,n} \mathbb{E}\left[L_v v_i\right]\left(f(x) - f(x^*) - \langle \nabla f(x^*), x - x^*\rangle\right) \\
&= 2\max_{i=1,\ldots,n} \mathbb{E}\left[L_v v_i\right]\left(f(x) - f(x^*)\right).
\end{aligned}$$

By comparing the above with (11) we have that $\mathcal{L} = \max_{i=1,\ldots,n} \mathbb{E}\left[L_v v_i\right]$. □

Lemma E.2 (PL inequality)**.** *If f is μ–strongly convex, then for all $x, y \in \mathbb{R}^d$*

$$\frac{1}{2\mu}\|\nabla f(x)\|_2^2 \geq f(x) - f(x^*), \quad \forall x \in \mathbb{R}^d. \tag{61}$$

Proof. Since f is μ–strongly convex, we have from, rearranging (6), that for all $x, y \in \mathbb{R}^d$

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|x - y\|_2^2.$$

Minimizing both sides of this inequality in y proves (61). □

The following lemma shows that the expected smoothness constant is always greater than the strong convexity constant.

Lemma E.3. *If the expected smoothness inequality* (11) *holds with constant $\mathcal{L}$ and f is μ–strongly convex, then $\mathcal{L} \geq \mu$.*

Proof. We have, since $\mathbb{E}\left[\nabla f_v(x) - \nabla f_v(x^*)\right] = \nabla f(x)$

$$\begin{aligned}
\mathbb{E}\left[\|\nabla f_v(x) - \nabla f_v(x^*) - \nabla f(x)\|_2^2\right] &= \mathbb{E}\left[\|\nabla f_v(x) - \nabla f_v(x^*)\|_2^2\right] - \|\nabla f(x)\|_2^2 \\
&\overset{(11)+(61)}{\leq} 2(\mathcal{L} - \mu)(f(x) - f(x^*)).
\end{aligned} \tag{62}$$

Hence $2(\mathcal{L} - \mu)(f(x) - f(x^*)) \geq 0$, which means $\mathcal{L} \geq \mu$. □

Remark E.1. *Consider the expected residual constant ρ defined in 4.2. This constant verifies for all $x \in \mathbb{R}^d$,*

$$\mathbb{E}\left[\|\nabla f_v(x) - \nabla f_v(x^*) - \nabla f(x)\|\right] \leq 2\rho(f(x) - f(x^*)).$$

From Equation (62)*, we can see that we can use $\rho = \mathcal{L}$ as the expected residual constant.*

E.2 Expected smoothness constant for particular samplings

The results on the expected smoothness constants related to the samplings we present here are all derived in [7] and thus are given without proof. The expected smoothness constant for b-nice sampling is given in Lemma 4.3. Here, we present this constant for single-element sampling, partition sampling and independent sampling.

Lemma E.4 ($\mathcal{L}$ for single-element sampling. Proposition 3.7 in [7])**.** *Consider S a single-element sampling from Definition D.1. If for all $i \in [n]$, f_i is L_i–smooth, then*

$$\mathcal{L} \quad = \quad \frac{1}{n} \max_{i \in [n]} \frac{L_i}{p_i}$$

where $p_i = \mathbb{P}(S = \{i\})$.

Remark E.2. *Consider S a single-element sampling from Definition D.1. It is then easy to see that the probabilities that maximize $\mathcal{L}$ are*

$$p_i = \frac{L_i}{\sum_{j \in [n]} L_j}.$$

Consequently,

$$\mathcal{L} = \bar{L} \overset{def}{=} \frac{1}{n} \sum_{i=1}^{n} L_i.$$

In contrast, for uniform single-element sampling, *i.e.,* when $p_i = \frac{1}{n}$ for all i, we have $\mathcal{L} = L_{\max}$, which can be significantly larger than $\bar{L}$. Since the step sizes of all our algorithm are a decreasing function of $\mathcal{L}$, importance sampling can lead to much faster algorithms.

Lemma E.5 ($\mathcal{L}$ for partition sampling. Proposition 3.7 in [7])**.** *Given a partition $\mathcal{B}$ of [n], consider S a partition sampling from Definition D.3. For all $B \in \mathcal{B}$, suppose that $f_B(x) \overset{def}{=} \frac{1}{b} \sum_{i \in \mathcal{B}} f_i(x)$ is L_B–smooth. Then, with $p_B = \mathbb{P}(S = B)$*

$$\mathcal{L} \quad = \quad \frac{1}{n} \max_{B \in \mathcal{B}} \frac{L_B}{p_B}$$

Lemma E.6 ($\mathcal{L}$ for independent sampling. Proposition 3.8 in [7])**.** *Consider S a single-element sampling from Definition D.3. Note $p_i = \mathbb{P}(i \in S)$. If for all $i \in [n]$, f_i is L_i–smooth and f is L–smooth, then*

$$\mathcal{L} \quad = \quad L + \max_{i \in [n]} \frac{1 - p_i}{p_i} \frac{L_i}{n}$$

where $p_i = \mathbb{P}(S = \{i\})$.

F Expected residual

In this section, we compute bounds on the expected residual ρ from Lemma 4.2.

Lemma F.1. *Let $v = [v_1, \ldots, v_n] \in \mathbb{R}^n$ be an unbiased sampling vector with $v_i \geq 0$ with probability one. It follows that the expected residual constant exists with*

$$\rho = \frac{\lambda_{\max}(\operatorname{Var}[v])}{n} L_{\max}, \tag{63}$$

where $\operatorname{Var}[v] = \mathbb{E}\left[(v - \mathbb{1})(v - \mathbb{1})^\top\right]$.

Before the proof, let us introduce the following lemma (inspired from `https://www.cs.ubc.ca/~nickhar/W12/NotesMatrices.pdf`).

Lemma F.2 (Trace inequality)**.** *Let A and B be symmetric $n \times n$ such that $A \succcurlyeq 0$. Then,*

$$Tr\,(AB) \leq \lambda_{\max}(B) Tr\,(A)$$

Proof. Let $A = \sum_{i=1}^n \lambda_i(A) U_i U_i^\top$, where $\lambda_1(A) \geq \ldots \geq \lambda_n(A) \geq 0$ denote the ordered eigenvalues of matrix A. Setting $V_i \overset{\text{def}}{=} \sqrt{\lambda_i(A)} U_i$ for all $i \in [n]$, we can write $A = \sum_{i=1}^n V_i V_i^\top$. Then,

$$\begin{aligned}\text{Tr}\left(AB\right) &= \text{Tr}\left(\sum_{i=1}^n V_i V_i^\top B\right) = \sum_{i=1}^n \text{Tr}\left(V_i V_i^\top B\right) = \sum_{i=1}^n \text{Tr}\left(V_i^\top B V_i\right) = \sum_{i=1}^n V_i^\top B V_i \\ &\leq \lambda_{\max}(B) \sum_{i=1}^n V_i^\top V_i = \lambda_{\max}(B) \text{Tr}\left(A\right),\end{aligned}$$

where we use that $B \preccurlyeq \lambda_{\max}(B) I_n$. □

We now turn to the proof of the theorem.

Proof. Let $v = [v_1, \ldots, v_n] \in \mathbb{R}^n$ be an unbiased sampling vector with $v_i \geq 0$ with probability one. We will show that there exists $\rho \in \mathbb{R}_+$ such that:

$$\mathbb{E}\left[\|\nabla f_v(w) - \nabla f_v(x^*) - (\nabla f(w) - \nabla f(x^*))\|_2^2\right] \leq 2\rho \left(f(w) - f(x^*)\right). \tag{64}$$

Let us expand the squared norm first. Define $DF(w)$ as the Jacobian of $F(w) \overset{def}{=} [f_1(w), \ldots, f_n(w)]$ We denote $R \overset{\text{def}}{=} (DF(w) - DF(x^*))$

$$\begin{aligned}C &\overset{\text{def}}{=} \|\nabla f_v(w) - \nabla f_v(x^*) - (\nabla f(w) - \nabla f(x^*))\|_2^2 \\ &= \frac{1}{n^2} \|(DF(w) - DF(x^*))(v - \mathbb{1})\|_2^2 \\ &= \frac{1}{n^2} \langle R(v - \mathbb{1}), R(v - \mathbb{1}) \rangle_{\mathbb{R}^d} \\ &= \frac{1}{n^2} \text{Tr}\left((v - \mathbb{1})^\top R^\top R (v - \mathbb{1})\right) \\ &= \frac{1}{n^2} \text{Tr}\left(R^\top R (v - \mathbb{1})(v - \mathbb{1})^\top\right).\end{aligned}$$

Taking expectation,

$$\begin{aligned}\mathbb{E}\left[C\right] &= \frac{1}{n^2} \text{Tr}\left(R^\top R \text{Var}\left[v\right]\right) \\ &\leq \frac{1}{n^2} \text{Tr}\left(R^\top R\right) \lambda_{\max}(\text{Var}\left[v\right]).\end{aligned} \tag{65}$$

Moreover, since the f_i's are convex and L_i-smooth, it follows from equation 2.1.7 in [16] that

$$\begin{aligned}\text{Tr}\left(R^\top R\right) &= \sum_{i=1}^n \|\nabla f_i(w) - \nabla f_i(x^*)\|_2^2 \\ &\leq 2 \sum_{i=1}^n L_i (f_i(w) - f_i(x^*) - \langle \nabla f_i(x^*), w - x^* \rangle) \\ &\leq 2n L_{\max}(f(w) - f(x^*)).\end{aligned} \tag{66}$$

Therefore,

$$\mathbb{E}\left[C\right] \overset{(65)+(66)}{\leq} 2 \frac{\lambda_{\max}(\text{Var}\left[v\right])}{n} L_{\max}(f(w) - f(x^*)). \tag{67}$$

Which means

$$\rho = \frac{\lambda_{\max}(\text{Var}\left[v\right])}{n} L_{\max} \tag{68}$$

□

Hence depending on the sampling S, we need to study the eigenvalues of the matrix $\mathrm{Var}\left[v\right]$, whose general term is given by

$$\left(\mathrm{Var}\left[v\right]\right)_{ij} \quad = \quad \begin{cases} \frac{1}{p_i} - 1 & \text{if } i = j \\ \frac{P_{ij}}{p_i p_j} - 1 & \text{otherwise,} \end{cases} \tag{69}$$

with

$$p_i \overset{\text{def}}{=} \mathbb{P}(i \in S) \text{ and } P_{ij} \overset{\text{def}}{=} \mathbb{P}(i \in S, j \in S) \text{ for } i, j \in [n] \tag{70}$$

To specialize our results to particular samplings, we introduce some notations:

- $\mathcal{B}$ designates all the possible sets for the sampling S,
- $b = |B|$, where $B \in \mathcal{B}$, when the sizes of all the elements of $\mathcal{B}$ are equal.

F.1 Expected residual for uniform b-nice sampling

Lemma F.3 (ρ for b-nice sampling)**.** *Consider b-nice sampling from Definition 3.3. If each f_i is $L_{\max}$-smooth, then*

$$\rho = \frac{n-b}{(n-1)b} L_{\max}. \tag{71}$$

Proof. For uniform b-nice sampling, we have using notations from (70)

$$\begin{aligned} \forall i \in [n], p_i \quad &= \quad \frac{c_1}{|\mathcal{B}|}, \\ \forall i, j \in [n], P_{ij} \quad &= \quad \frac{c_2}{|\mathcal{B}|}, \end{aligned}$$

with $c_1 = \binom{n-1}{b-1}$, $c_2 = \binom{n-2}{b-2}$ and $|\mathcal{B}| = \binom{n}{b}$. Hence,

$$\mathrm{Var}\left[v\right] \overset{(69)}{=} \begin{bmatrix} \frac{|\mathcal{B}|}{c_1} - 1 & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \dots & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 \\ \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \frac{|\mathcal{B}|}{c_1} - 1 & \dots & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 \\ \vdots & & \ddots & & \vdots \\ \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \dots & \dots & \frac{|\mathcal{B}|}{c_1} - 1 & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 \\ \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \dots & \dots & \frac{|\mathcal{B}|c_2}{c_1^2} - 1 & \frac{|\mathcal{B}|}{c_1} - 1 \end{bmatrix}.$$

As noted in Appendix C of [8], $\mathrm{Var}\left[v\right]$ is then a circulant matrix with associated vector

$$\left(\frac{|\mathcal{B}|}{c_1} - 1, \frac{|\mathcal{B}|c_2}{c_1^2} - 1, \dots, \frac{|\mathcal{B}|c_2}{c_1^2} - 1\right),$$

and, as such, it has two eigenvalues

$$\begin{aligned} \lambda_1 \quad &\overset{\text{def}}{=} \quad \frac{|\mathcal{B}|}{c_1}\left(1 + (n-1)\frac{c_2}{c_1}\right) - n = 0, \\ \lambda_2 \quad &\overset{\text{def}}{=} \quad \frac{|\mathcal{B}|}{c_1}\left(1 - \frac{c_2}{c_1}\right) = \frac{n(n-b)}{b(n-1)}. \end{aligned} \tag{72}$$

Hence, the expected residual can be computed explicitely as

$$\rho \overset{(68)}{=} \frac{n-b}{(n-1)b} L_{\max}. \tag{73}$$

□

We can see that the residual constant is a descreasing function of b and in particular: $\rho(1) = L_{\max}$ and $\rho(n) = 0$.

F.2 Expected residual for uniform partition sampling

Lemma F.4 (ρ for uniform partition sampling)**.** *Suppose that b divises n and consider partition sampling from Definition D.2. Given a partition $\mathcal{B}$ of $[n]$ of size $\frac{b}{n}$, if each f_i is $L_{\max}$-smooth, then,*

$$\rho = \left(1 - \frac{b}{n}\right) L_{\max}. \tag{74}$$

Proof. Recall that for partition sampling, we choose *a priori* a partition $\mathcal{B} = B_1 \sqcup \cdots \sqcup B_{\frac{n}{b}}$ of $[n]$. Then, for $k \in [\frac{n}{b}]$,

$$\forall i \in [n], p_i = \begin{cases} p_{B_k} = \frac{b}{n} & \text{if } i \in B_k \\ 0 & \text{otherwise,} \end{cases} \tag{75}$$

$$\forall i, j \in [n], P_{ij} = \begin{cases} p_{B_k} = \frac{b}{n} & \text{if } i, j \in B_k \\ 0 & \text{otherwise.} \end{cases} \tag{76}$$

Let $k \in [\frac{n}{b}]$. If $i, j \in B_k$, then $\frac{1}{p_i} - 1 = \frac{P_{ij}}{p_i p_j} - 1 = \frac{n}{b} - 1$.

As a result, up to a reordering of the observations, $\operatorname{Var}[v]$ is a block diagonal matrix, whose diagonal matrices, which are all equal, are given by, for $k \in [\frac{n}{b}]$,

$$V_k = (\frac{n}{b} - 1)\mathbb{1}_b\mathbb{1}_b^\top = \begin{bmatrix} \frac{n}{b} - 1 & \frac{n}{b} - 1 & \dots & \frac{n}{b} - 1 & \frac{n}{b} - 1 \\ \frac{n}{b} - 1 & \frac{n}{b} - 1 & \dots & \frac{n}{b} - 1 & \frac{n}{b} - 1 \\ \vdots & & \ddots & & \vdots \\ \frac{n}{b} - 1 & \dots & \dots & \frac{n}{b} - 1 & \frac{n}{b} - 1 \\ \frac{n}{b} - 1 & \dots & \dots & \frac{n}{b} - 1 & \frac{n}{b} - 1 \end{bmatrix} \in \mathbb{R}^{b \times b}.$$

Since all the matrices on the diagonal are equal, the eigenvalues of $\operatorname{Var}[v]$ are simply those of one of these matrices. Any matrix $V_k = (\frac{n}{b} - 1)\mathbb{1}_b\mathbb{1}_b^\top$ we consider has two eigenvalues: 0 and $n - b$. Then,

$$\rho \overset{(68)}{=} \left(1 - \frac{b}{n}\right) L_{\max}. \tag{77}$$

□

If $b = n$, SVRG with uniform partition sampling boils down to gradient descent as we recover $\rho = 0$. For $b = 1$, we have $\rho = \left(1 - \frac{1}{n}\right) L_{\max}$.

F.3 Expected residual for independent sampling

Lemma F.5 (ρ for independent sampling)**.** *Consider independent sampling from Definition D.2. Let $p_i = \mathbb{P}(i \in S)$. If each f_i is $L_{\max}$-smooth, then*

$$\rho = \left(\frac{1}{\min\limits_{i \in [n]} p_i} - 1\right) \frac{L_{\max}}{n}. \tag{78}$$

Proof. Using the notations from (70), we have

$$\begin{aligned} \forall i \in [n], p_i &= p_i, \\ \forall i, j \in [n], P_{ij} &= p_i p_j \quad \text{when } i \neq j. \end{aligned}$$

Thus, according to (69):

$$\operatorname{Var}[v] = \operatorname{Diag}\left(\frac{1}{p_1} - 1, \frac{1}{p_2} - 1, \dots, \frac{1}{p_n} - 1\right).$$

whose largest eigenvalue is

$$\lambda_{\max}(\operatorname{Var}[v]) = \max_{i \in [n]} \frac{1}{p_i} - 1 = \frac{1}{\min\limits_{i \in [n]} p_i} - 1.$$

Consequently,

$$\rho \overset{(68)}{=} \left(\frac{1}{\min\limits_{i \in [n]} p_i} - 1 \right) \frac{L_{\max}}{n}. \tag{79}$$

□

If $p_i = \frac{1}{n}$ for all $i \in [n]$, which corresponds in expectation to uniform single-element sampling SVRG since $\mathbb{E}[|S|] = 1$, we have $\rho = \frac{n-1}{n} L_{\max}$. While if $p_i = 1$ for all $i \in [n]$, this leads to gradient descent and we recover $\rho = 0$.

The following remark gives a condition to construct an independent sampling with $E|S| = b$.

Remark F.1. *One can add the following condition on the probabilities: $\sum_{i=1}^n p_i = b$, such that $\mathbb{E}[|S|] = b$. Such a sampling is called b-independent sampling. This condition is obviously met if $p_i = \frac{b}{n}$ for all $i \in [n]$.*

Lemma F.6. *Let S be a independent sampling from $[n]$ and let $p_i = \mathbb{P}[i \in S]$ for all $i \in [n]$. If $\sum_{i=1}^n p_i = b$, then $\mathbb{E}[|S|] = b$.*

Proof. Let us model our sampling by a tossing of n independent rigged coins. Let $X_1, \ldots, X_n$ be n Bernoulli random variables representing these tossed coin, *i.e.,* $X_i \sim \mathcal{B}(p_i)$, with $p_i \in [0, 1]$ for $i \in [n]$. If $X_i = 1$, then the point i is selected in the sampling S. Thus the number of selected points in the mini-batch $|S|$ can be denoted as the following random variable $\sum_{i=1}^n X_i$, and its expectation equals

$$\mathbb{E}[|S|] = \mathbb{E}\left[\sum_{i=1}^n X_i\right] = \sum_{i=1}^n \mathbb{E}[X_i] = \sum_{i=1}^n p_i = b \ .$$

□

Remark F.2. *Note that one does not need the independence of the $(X_i)_{i=1,\ldots,n}$.*

F.4 Expected residual for single-element sampling

From Remark E.1, we can take $\mathcal{L}$ as the expected residual constant. Thus, we simply use the expected smoothness constant from Lemma E.4.

Lemma F.7 (ρ for single-element sampling)**.** *Consider single-element sampling from Definition D.1. If for all $i \in [n]$, f_i is L_i-smooth, then*

$$\rho = \frac{1}{n} \max_{i \in [n]} \frac{L_i}{p_i}.$$

G Additional experiments

G.1 Comparison of theoretical variants of SVRG

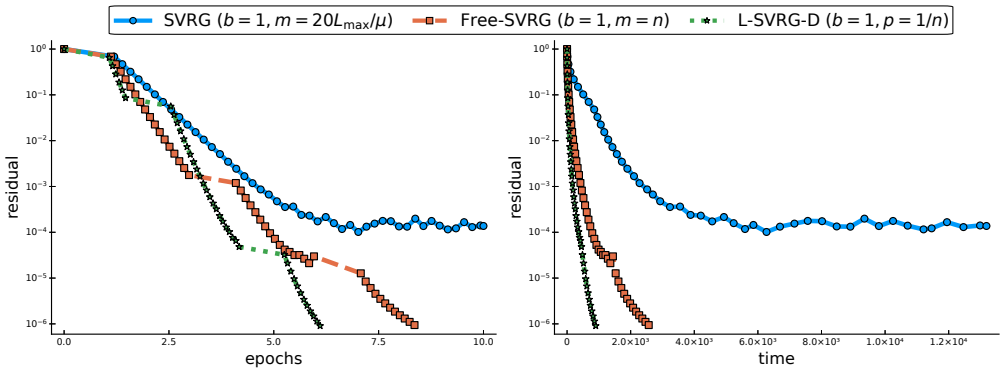
In this series of experiments, we compare the performance of the SVRG algorithm with the settings of [11] against *Free-SVRG* and *L-SVRG-D* with the settings given by our theory.

G.1.1 Experiment 1.a: Comparison without mini-batching ($b = 1$)

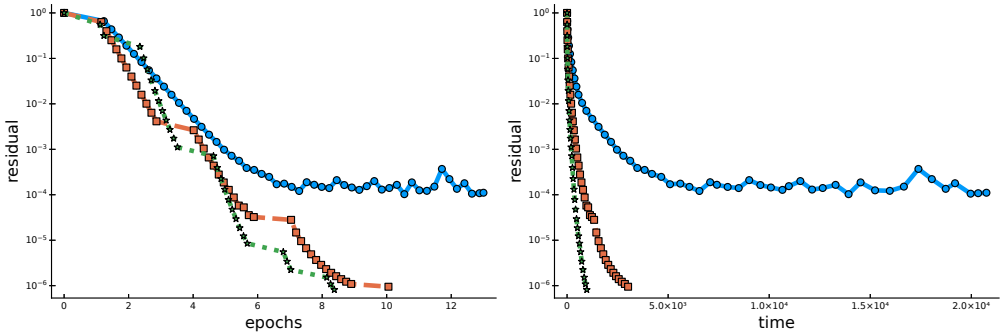
A widely used choice for the size of the inner loop is $m = n$. Since our algorithms allow for a free choice of the size of the inner loop, we set $m = n$ for *Free-SVRG* and $p = 1/n$ for *L-SVRG-D*, and use a mini-batch size $b = 1$, see Figures 4, 5, 6 and 7.

G.1.2 Experiment 1.b: optimal mini-batching

Here we use the optimal mini-batch sizes we derived for *Free-SVRG* in Table 1 and *L-SVRG-D* in (57). Since the original SVRG theory has no analysis for mini-batching, and the current existing theory shows that the total complexity of the algorithm is an increasing function of the mini-batch size, we use $b = 1$ for SVRG, see Figures 8, 9, 10 and 11.



(a) $\lambda = 10^{-1}$



(b) $\lambda = 10^{-3}$

Figure 4: $b = 1$, $m = n$ for *Free-SVRG* and *L-SVRG-D*, *YearPredictionMSD* data set.

G.1.3 Experiment 1.c: theoretical inner loop size or update probability without mini-batching

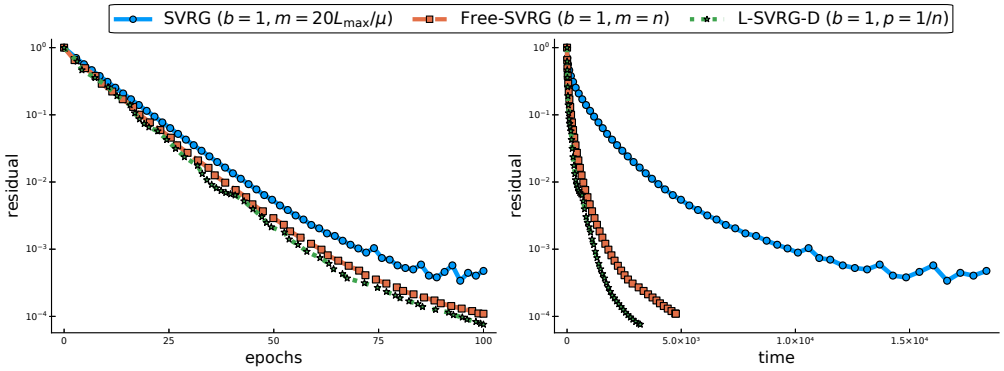
Here, using a batch size $b = 1$, we set the inner loop sizes for *Free-SVRG* to the optimal value $m^* = 3L_{\max}/\mu$ that we derived in Proposition 6.1. We set $p = 1/m^*$ for *L-SVRG-D*, see Figures 12, 13, 14 and 15.

G.2 Experiment 2.a: Comparing different choices for the mini-batch size

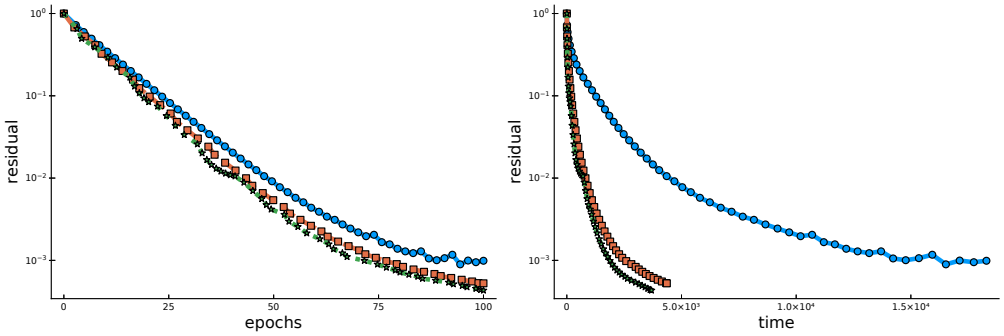
Here we consider *Free-SVRG* and compare its performance for different batch sizes: the optimal one b^*, 1, 100, $\sqrt{n}$ and n. In Figure 16, 17, 18 and 19 we show that we are able to predict the best optimal mini-batch size a priori using Table 1.

G.3 Experiment 2.b: Comparing different choices for the inner loop size

We set $b = 1$ and compare different values for the inner loop size: the optimal one m^*, $L_{\max}/\mu$, $3L_{\max}/\mu$ and $2n$ in order to validate our theory in Proposition 6.1, that is, that the overall performance of *Free-SVRG* is not sensitive to the range of values of m, so long as m is close to n, $L_{\max}/\mu$ or anything in between. And indeed, this is what we confirmed in Figures 20, 21, 22 and 23. The choice $m = 2n$ is the one suggested by [11] in their practical SVRG (Option II).

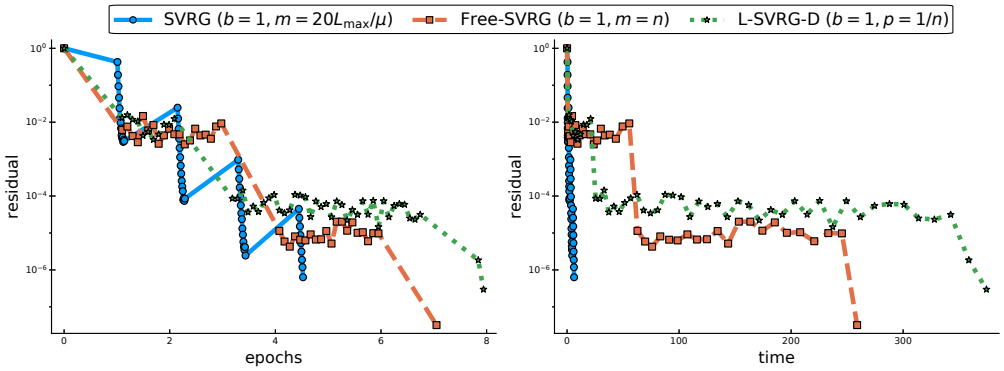


(a) $\lambda = 10^{-1}$

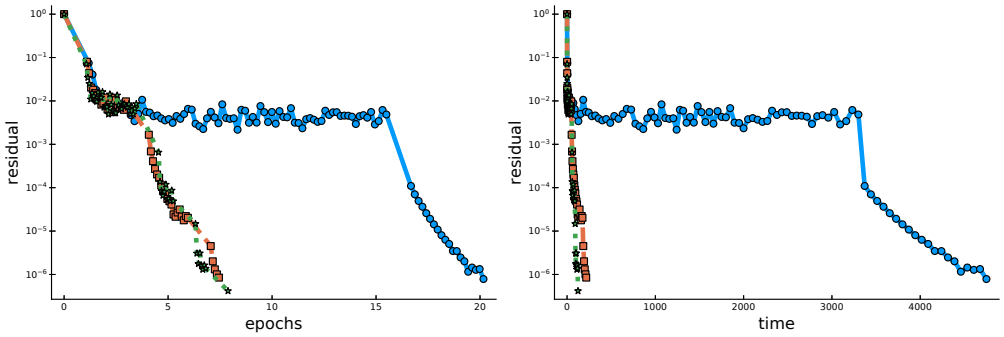


(b) $\lambda = 10^{-3}$

Figure 5: $b = 1$, $m = n$ for *Free-SVRG* and *L-SVRG-D*, *slice* data set.

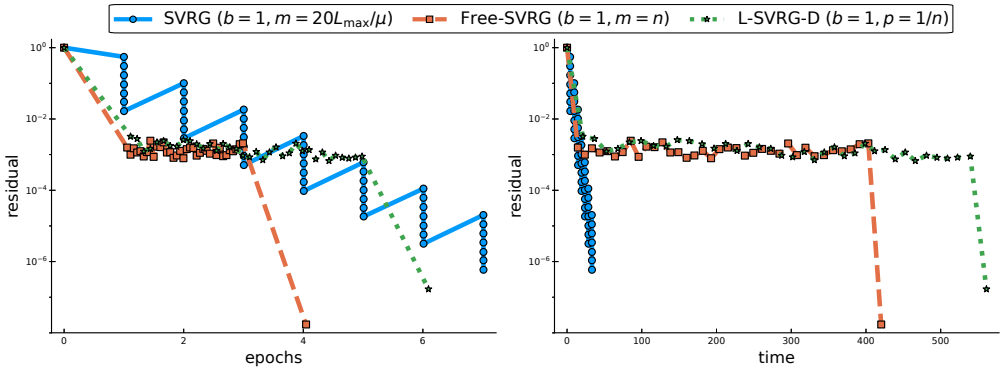


(a) $\lambda = 10^{-1}$

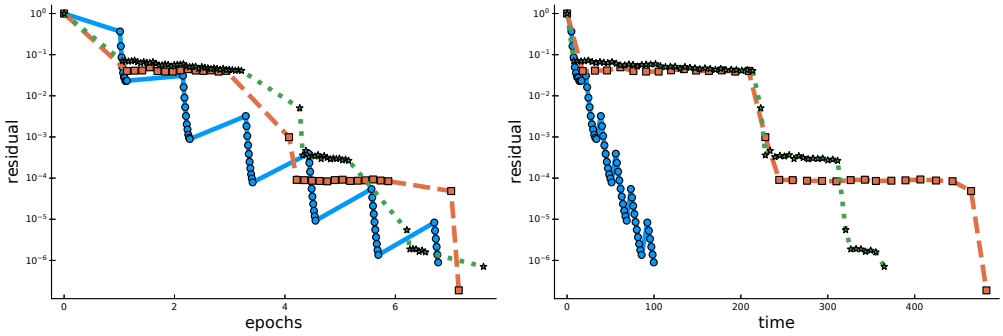


(b) $\lambda = 10^{-3}$

Figure 6: $b = 1$, $m = n$ for *Free-SVRG* and *L-SVRG-D*, *ijcnn1* data set.

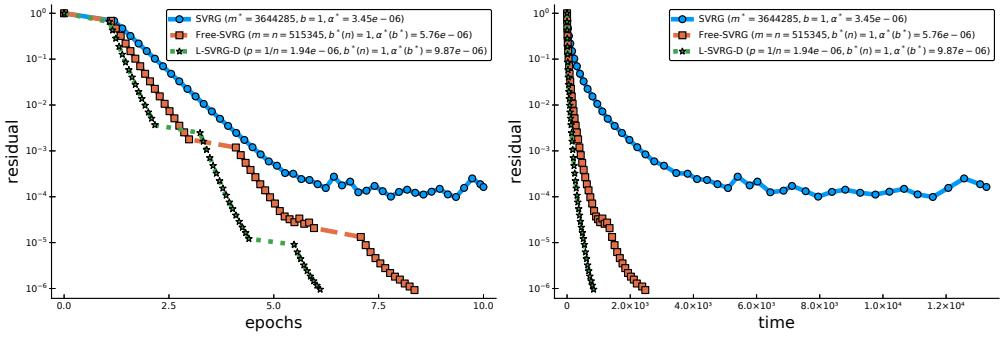


(a) $\lambda = 10^{-1}$

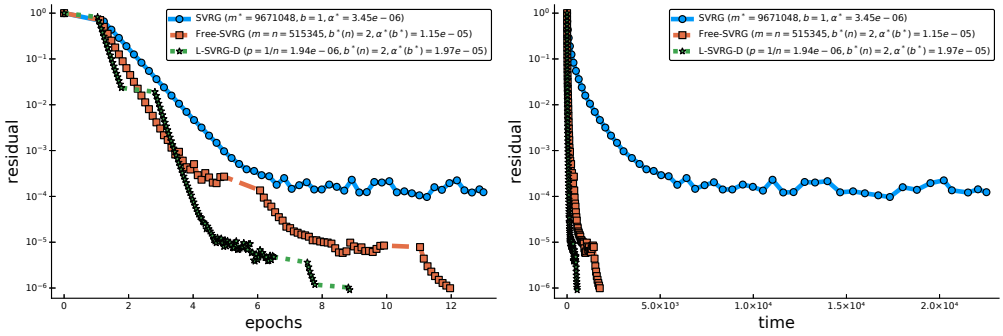


(b) $\lambda = 10^{-3}$

Figure 7: $b = 1$, $m = n$ for *Free-SVRG* and *L-SVRG-D*, *real-sim* data set.

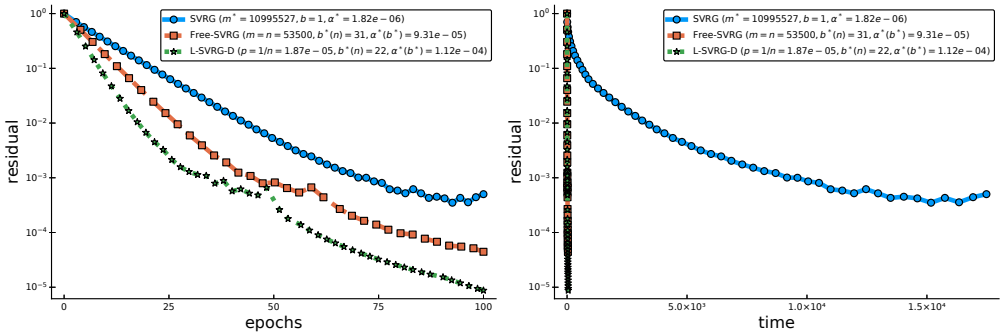


(a) $\lambda = 10^{-1}$

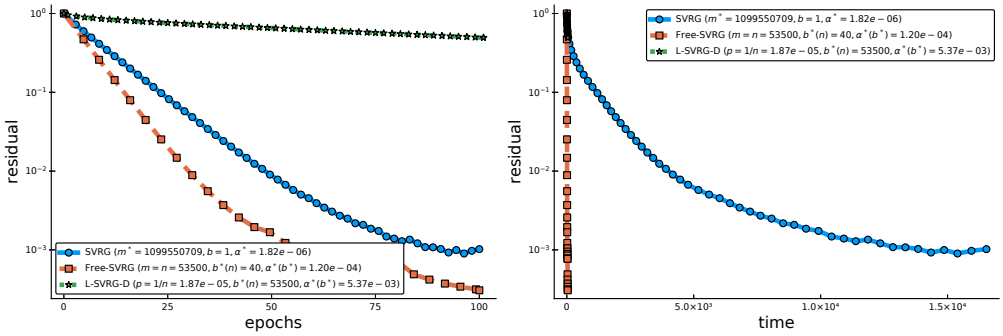


(b) $\lambda = 10^{-3}$

Figure 8: Optimal mini-batching when theoretically available, *YearPredictionMSD* data set.

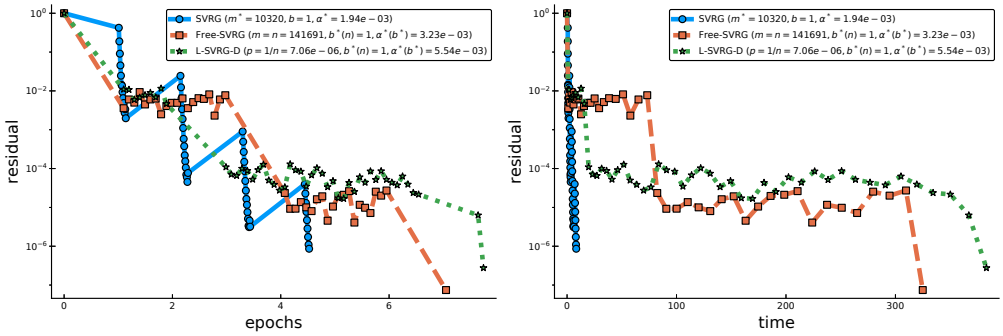


(a) $\lambda = 10^{-1}$

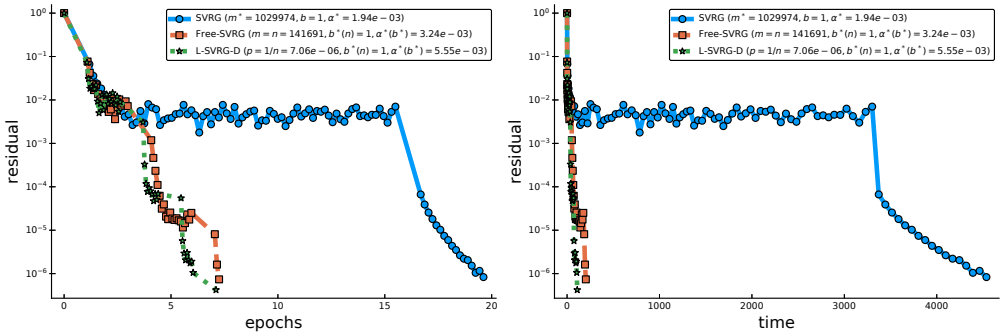


(b) $\lambda = 10^{-3}$

Figure 9: Optimal mini-batching when theoretically available, *slice* data set.

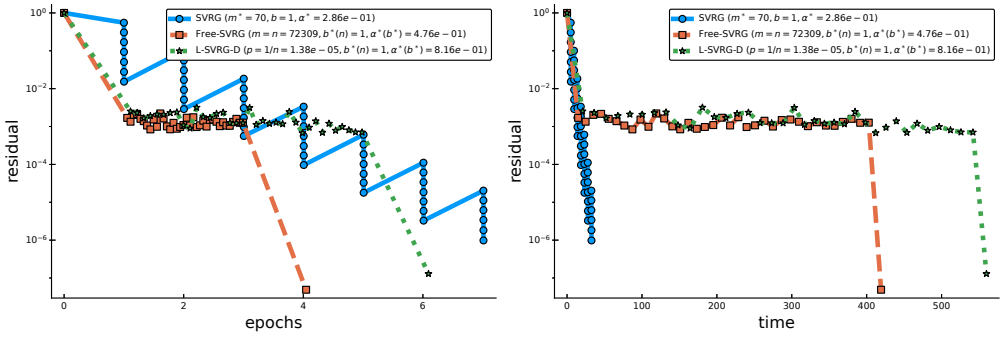


(a) $\lambda = 10^{-1}$

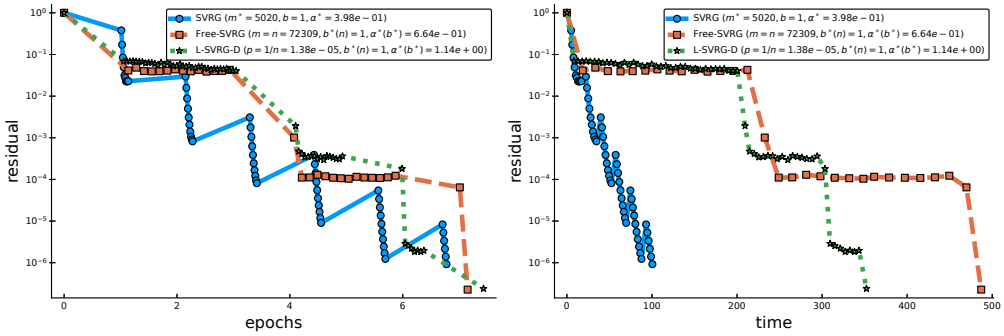


(b) $\lambda = 10^{-3}$

Figure 10: Optimal mini-batching when theoretically available, *ijcnn1* data set.

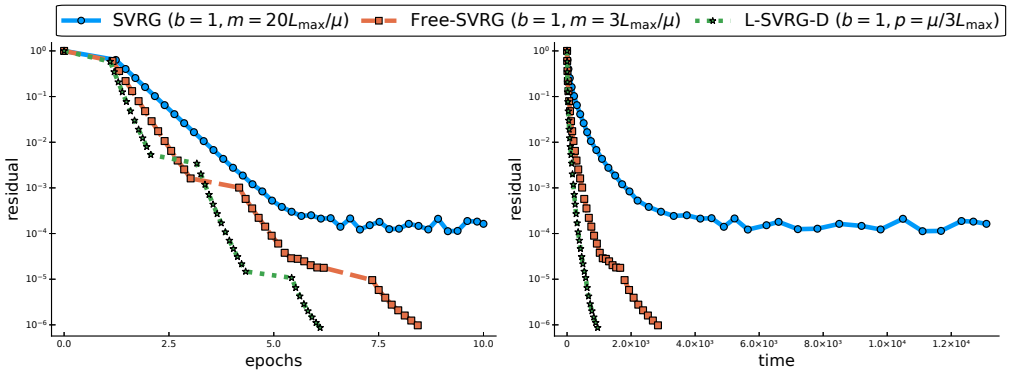


(a) $\lambda = 10^{-1}$

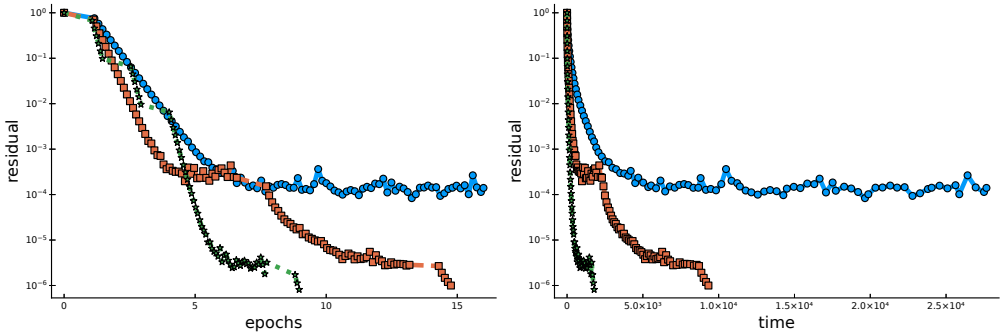


(b) $\lambda = 10^{-3}$

Figure 11: Optimal mini-batching when theoretically available, *real-sim* data set.

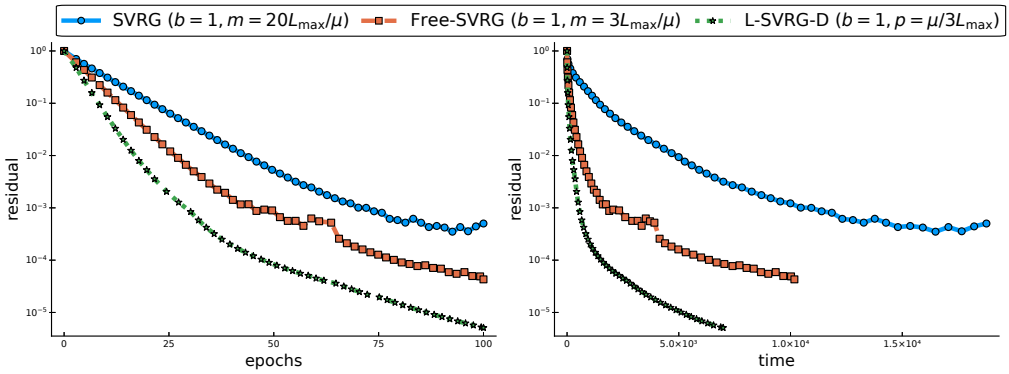


(a) $\lambda = 10^{-1}$

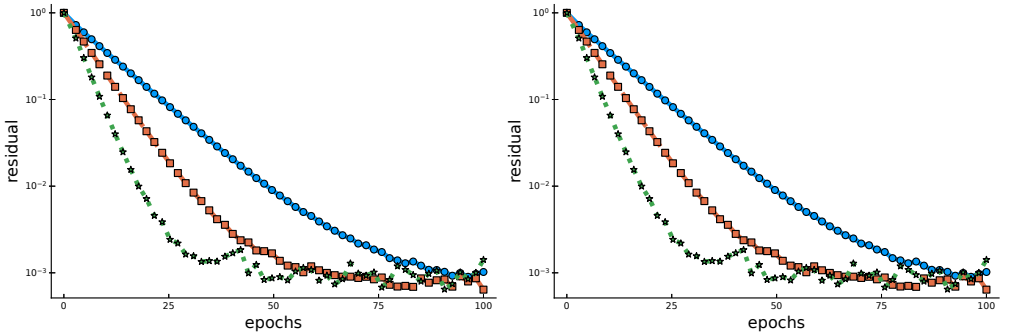


(b) $\lambda = 10^{-3}$

Figure 12: Optimal inner loop size when theoretically available, *YearPredictionMSD* data set.

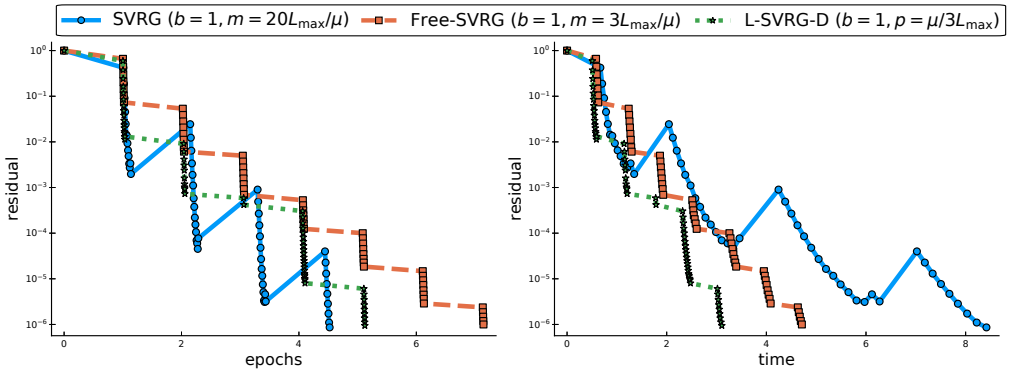


(a) $\lambda = 10^{-1}$

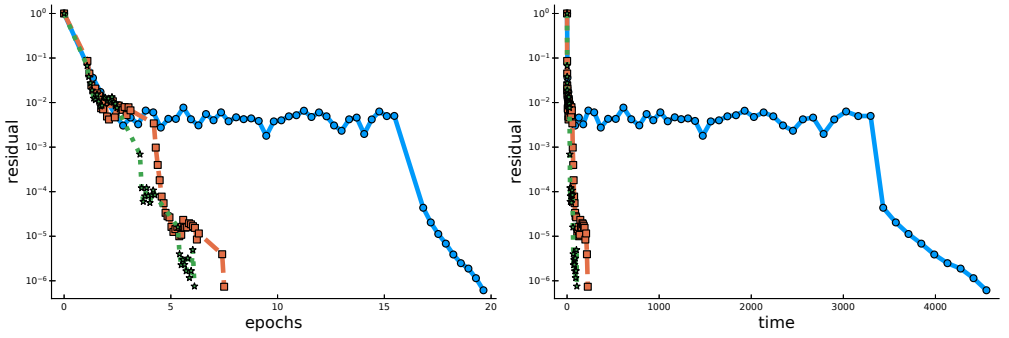


(b) $\lambda = 10^{-3}$

Figure 13: Optimal inner loop size when theoretically available, *slice* data set.

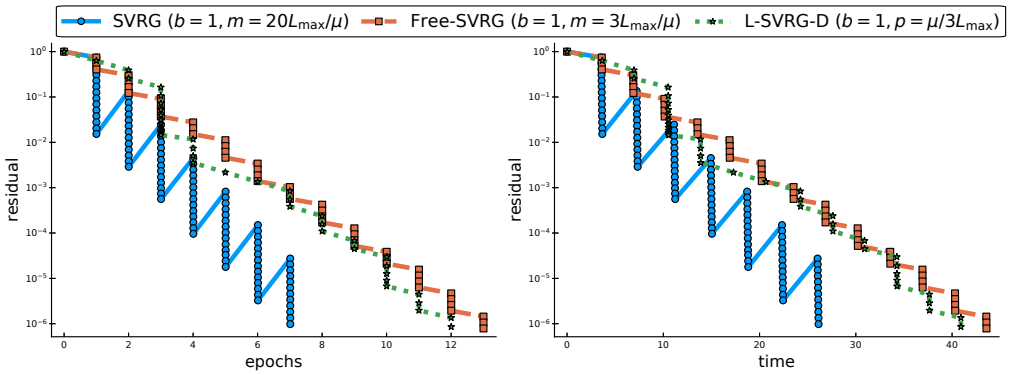


(a) $\lambda = 10^{-1}$

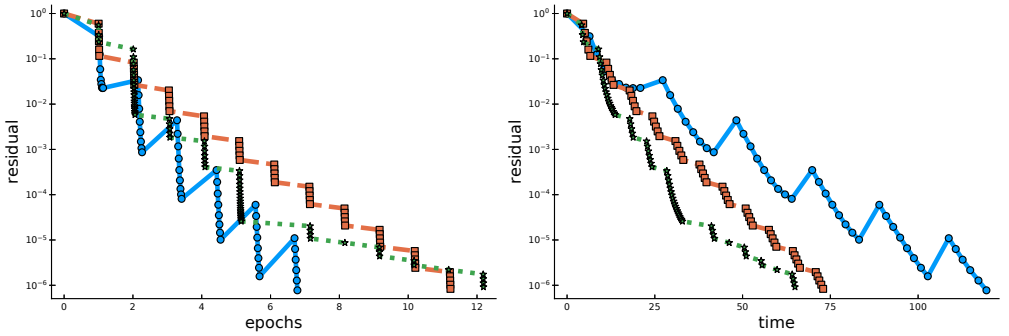


(b) $\lambda = 10^{-3}$

Figure 14: Optimal inner loop size when theoretically available, *ijcnn1* data set.

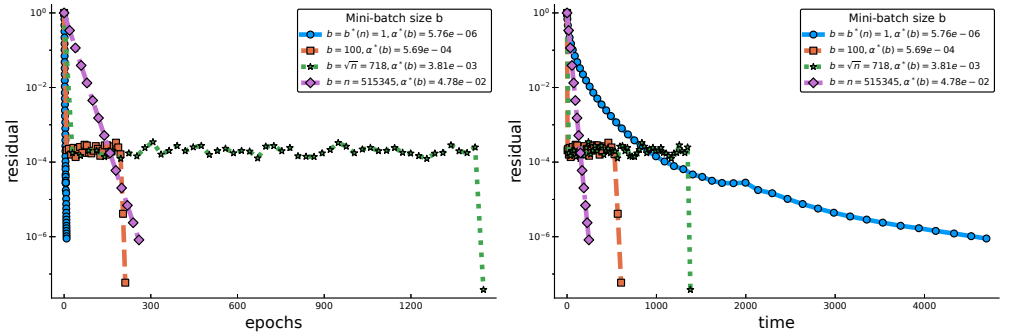


(a) $\lambda = 10^{-1}$

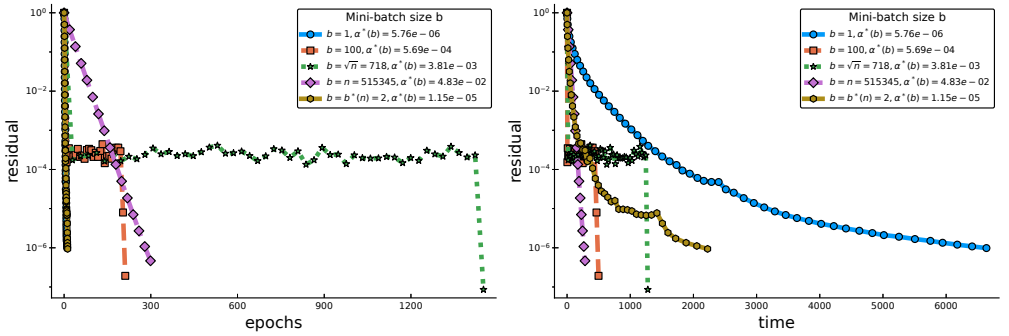


(b) $\lambda = 10^{-3}$

Figure 15: Optimal inner loop size when theoretically available, *real-sim* data set.

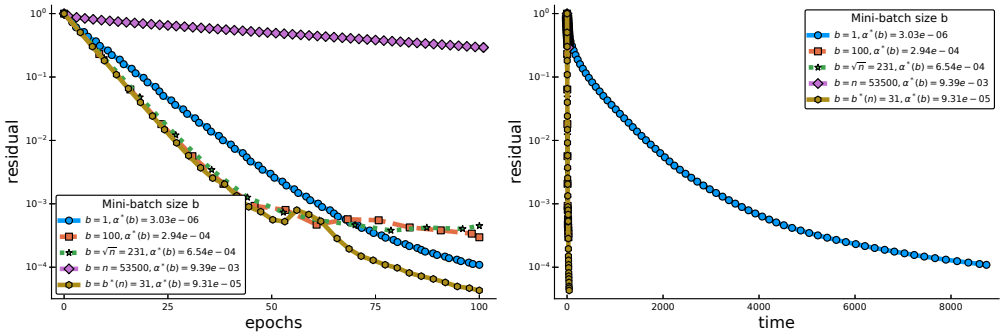


(a) $\lambda = 10^{-1}$.

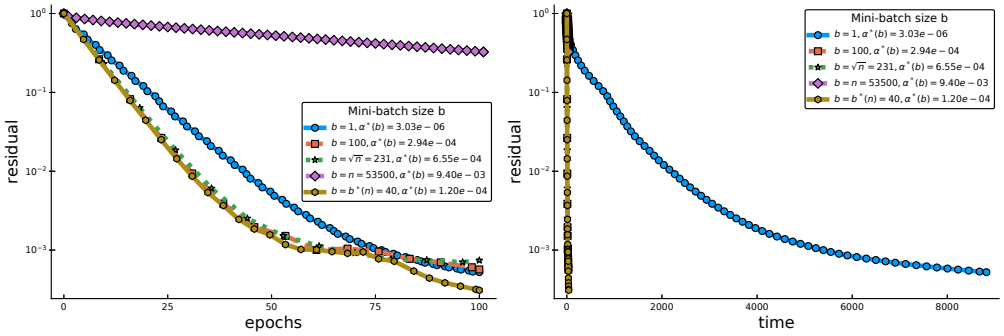


(b) $\lambda = 10^{-3}$.

Figure 16: Impact of the mini-batch size on *Free-SVRG* for the *YearPredictionMSD* data set.

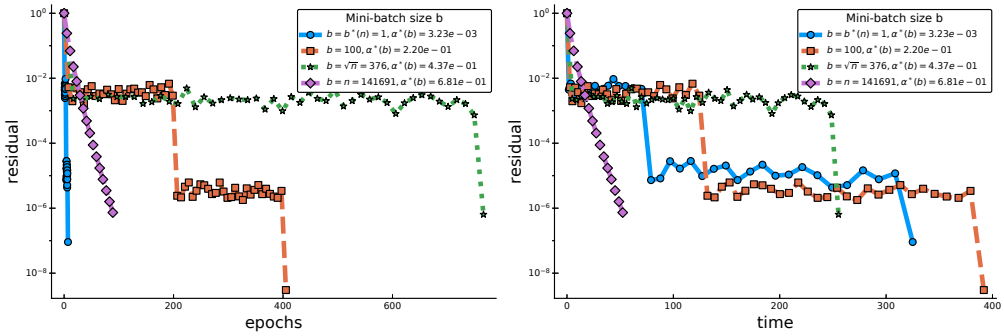


(a) $\lambda = 10^{-1}$.

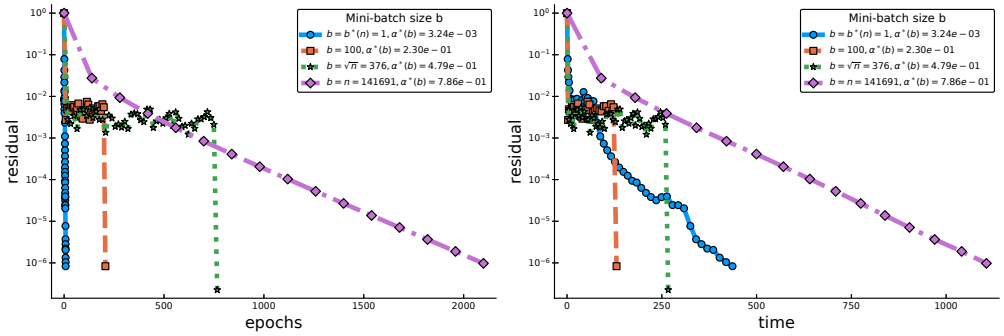


(b) $\lambda = 10^{-3}$.

Figure 17: Impact of the mini-batch size on *Free-SVRG* for the *slice* data set.

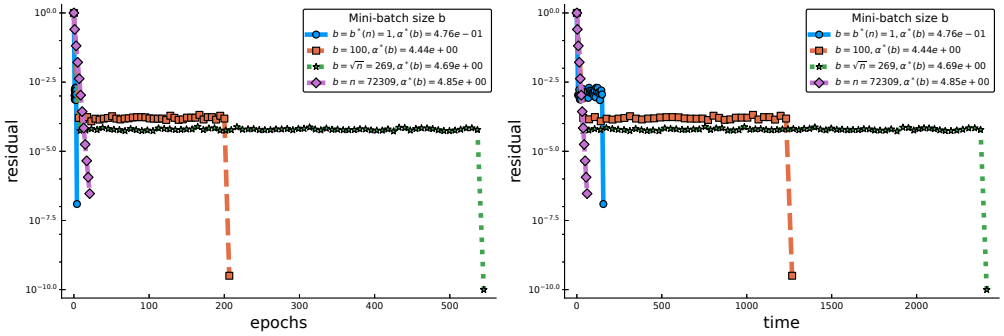


(a) $\lambda = 10^{-1}$.

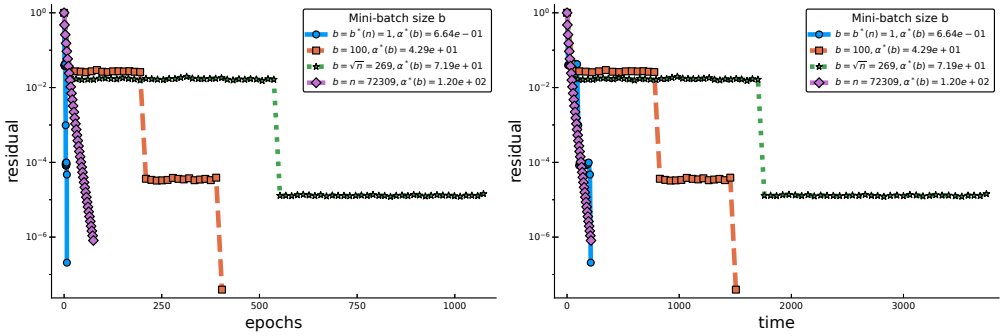


(b) $\lambda = 10^{-3}$.

Figure 18: Impact of the mini-batch size on *Free-SVRG* for the *ijcnn1* data set.

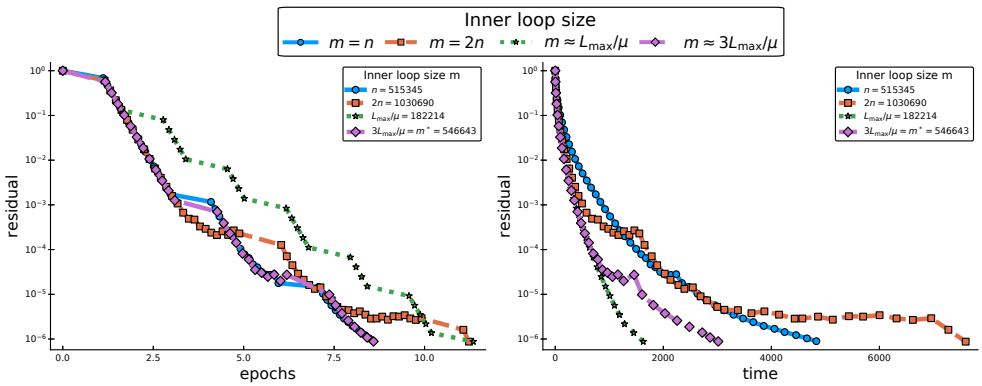


(a) $\lambda = 10^{-1}$.

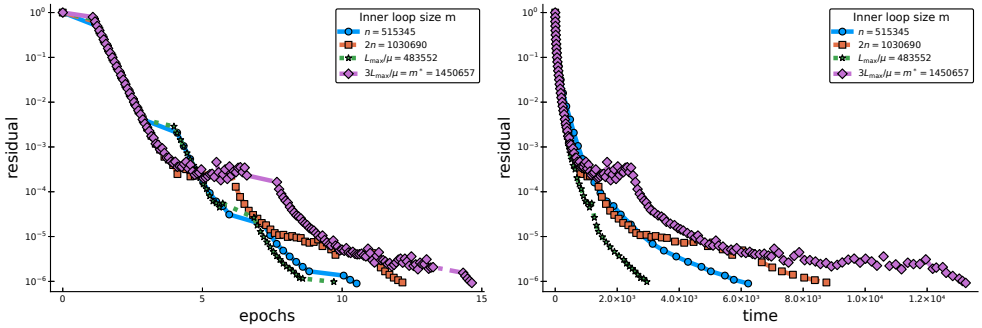


(b) $\lambda = 10^{-3}$.

Figure 19: Impact of the mini-batch size on *Free-SVRG* for the *real-sim* data set.

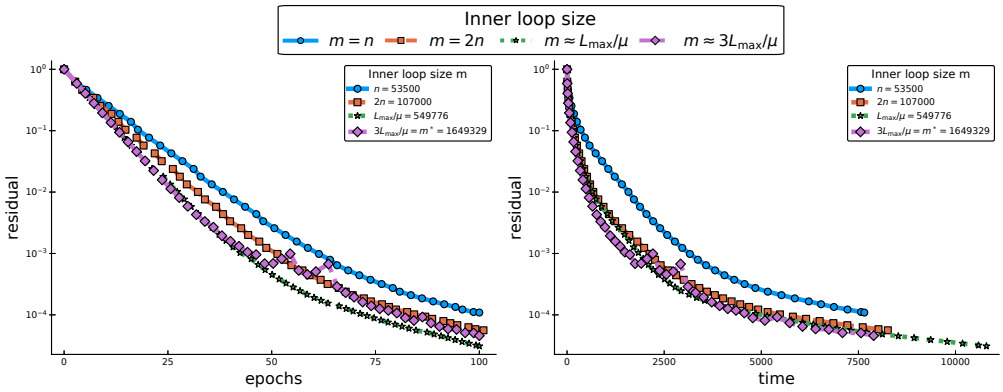


(a) $\lambda = 10^{-1}$.

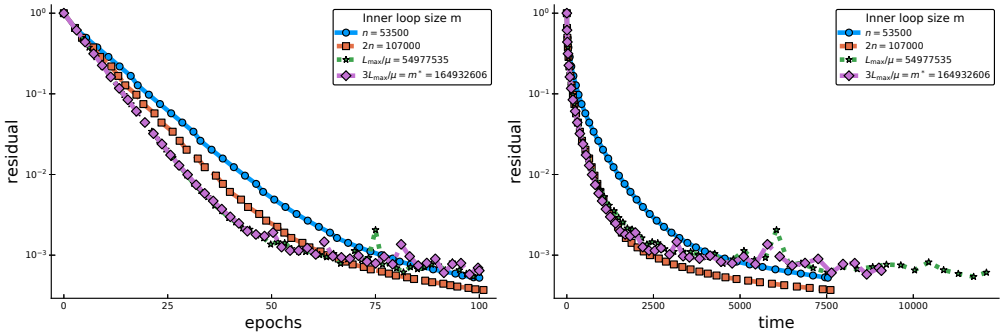


(b) $\lambda = 10^{-3}$.

Figure 20: Impact of the inner loop size on *Free-SVRG* for the *YearPredictionMSD* data set.

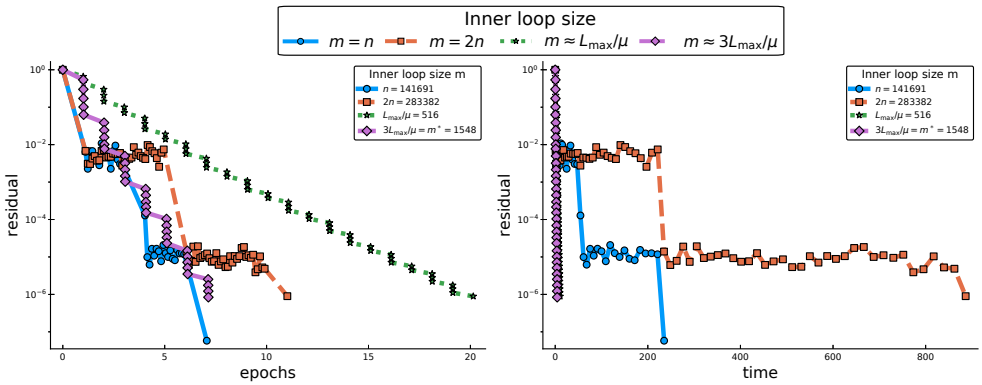


(a) $\lambda = 10^{-1}$

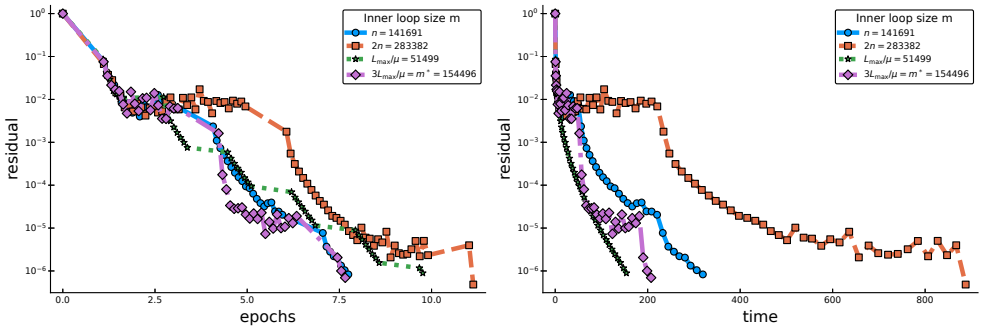


(b) $\lambda = 10^{-3}$

Figure 21: Impact of the inner loop size on *Free-SVRG* for the *slice* data set.

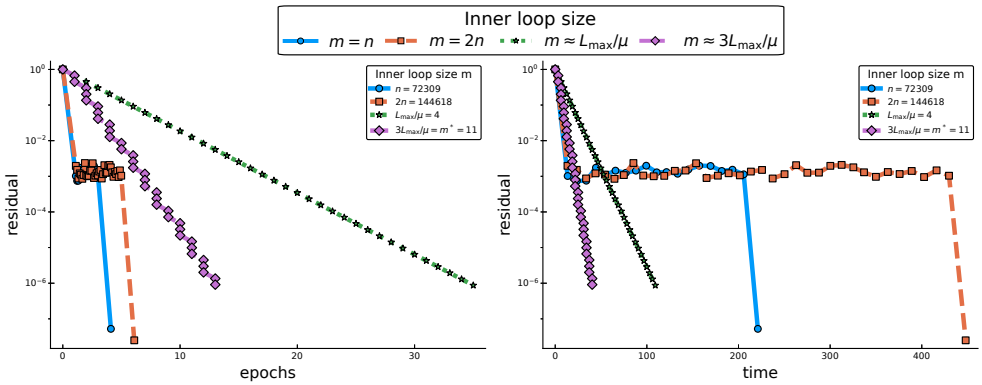


(a) $\lambda = 10^{-1}$

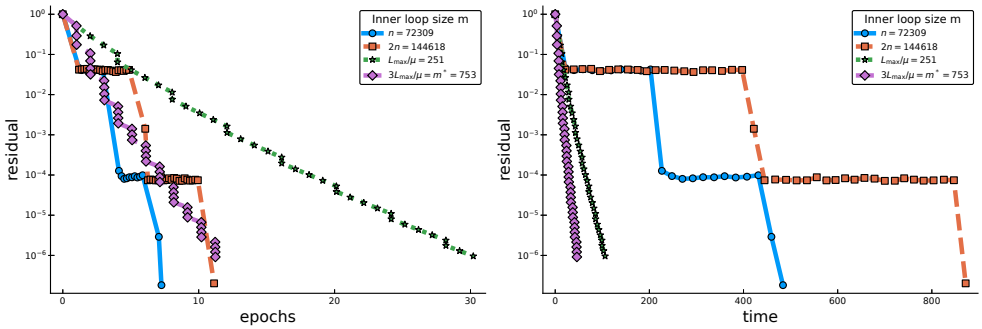


(b) $\lambda = 10^{-3}$

Figure 22: Impact of the inner loop size on *Free-SVRG* for the *ijcnn1* data set.



(a) $\lambda = 10^{-1}$



(b) $\lambda = 10^{-3}$

Figure 23: Impact of the inner loop size on *Free-SVRG* for the *real-sim* data set.