Towards closing the gap between the theory and practice of SVRG
Othmane Sebbouh, Nidham Gazagnadou, Samy Jelassi, Francis Bach, Robert Gower

To cite this version:
Othmane Sebbouh, Nidham Gazagnadou, Samy Jelassi, Francis Bach, Robert Gower. Towards closing the gap between the theory and practice of SVRG. Conference on Neural Information Processing Systems, Dec 2019, Vancouver, Canada. hal-02365285

HAL Id: hal-02365285
https://hal.telecom-paris.fr/hal-02365285
Submitted on 15 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
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 Supérieure
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 $\mu$-strongly convex and $L$-smooth function $f$ where

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

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 $i$th 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 original 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. 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

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

where $\alpha > 0$ is the stepsize, $B \subseteq [n] \overset{\text{def}}{=} \{1, \ldots, n\}$ 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] 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 $n$, $m$, $b$, $\mu$, $L$ and $L_{\max} \overset{\text{def}}{=} \max_{i \in [n]} L_i$:

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

so long as the stepsize is

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

By total complexity we mean the total number of individual $\nabla f_i$ 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 [4] for an example for how total complexity evolves as we increase the mini-batch size.

---

Footnote: 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 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 \( m \geq 20L_{\text{max}}/\mu \) in order to ensure a resulting iteration complexity of \( O((n + L_{\text{max}}/\mu) \log(1/\epsilon)) \). 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 \( O((n + L_{\text{max}}/\mu) \log(1/\epsilon)) \).

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 \( O((n + \sqrt{nL_{\text{max}}}/\mu) \log(1/\epsilon)) \). 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 \( L_{\text{max}}/\mu \geq n \), in other words for problems that are sufficiently ill-conditioned.

<table>
<thead>
<tr>
<th>( n \leq \frac{L}{\mu} )</th>
<th>( \frac{L}{\mu} &lt; n &lt; \frac{3L_{\text{max}}}{\mu} )</th>
<th>( n \geq \frac{3L_{\text{max}}}{\mu} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_{\text{max}} \geq \frac{nL}{3} )</td>
<td>( L_{\text{max}} &lt; \frac{nL}{3} )</td>
<td>( L_{\text{max}} \geq \frac{nL}{3} )</td>
</tr>
<tr>
<td>( L_{\text{max}} &lt; \frac{nL}{3} )</td>
<td>( \hat{b} )</td>
<td>( \frac{(3L_{\text{max}} - L)n}{n(n - 1)\mu - nL + 3L_{\text{max}}} )</td>
</tr>
<tr>
<td>( \hat{b} )</td>
<td>( \min{\hat{b}, \hat{b}} )</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1: Optimal mini-batch sizes for Algorithm 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: \( \hat{b} = \sqrt{n \frac{3L_{\text{max}} - L}{nL - 3L_{\text{max}}} \hat{b}} = \frac{(3L_{\text{max}} - L)n}{n(n - 1)\mu - nL + 3L_{\text{max}}} \).
We say that $f$ is $L$-smooth \(^{(5)}\) and $\mu$-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 simultaneously 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

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

Indeed these are unbiased estimators since

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

Likewise we can show that $\mathbb{E}_\mathcal{D} [\nabla f_v(x)] = \nabla f(x)$. Computing $\nabla f_v$ is cheaper than computing the full gradient $\nabla f$ 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 $p_B \overset{\text{def}}{=} \mathbb{P}(S = B), \quad \forall B \subseteq [n].$ A sampling $S$ is called proper if for every $i \in [n]$, we have that $p_i \overset{\text{def}}{=} \mathbb{P} [i \in S] = \sum_{C : i \in C} p_C > 0$.

We can build a sampling vector using sampling as follows.

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

$$ v(S) = \mathcal{P}^{-1} \sum_{i \in S} c_i \overset{\text{def}}{=} \mathcal{P}^{-1} c_S. \quad (9) $$

It follows that $v$ is a sampling vector.

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

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

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

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

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 $x, y \in \mathbb{R}^d$,

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

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

We say that $f$ is $L$-smooth \(^{(5)}\) and $\mu$-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.
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 points: 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.

To declutter the notation, we define for a given step size $\alpha > 0$:

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

Algorithm 1 Free-SVRG

<table>
<thead>
<tr>
<th>Parameters</th>
<th>inner-loop length $m$, step size $\alpha$, a sampling vector $v \sim \mathcal{D}$, and $p_t$ defined in (10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>$w_0 = x_0^m \in \mathbb{R}^d$</td>
</tr>
<tr>
<td>for $s = 1, 2, \ldots$ do</td>
<td></td>
</tr>
<tr>
<td>for $t = 0, 1, \ldots, m - 1$ do</td>
<td></td>
</tr>
<tr>
<td>Sample $v_t \sim \mathcal{D}$</td>
<td></td>
</tr>
<tr>
<td>$g_t^s = \nabla f_{v_t}(x_t^s) - \nabla f_{v_t}(w_{s-1}) + \nabla f(w_{s-1})$</td>
<td></td>
</tr>
<tr>
<td>$x_{t+1}^s = x_t^s - \alpha g_t^s$</td>
<td></td>
</tr>
<tr>
<td>$w_s = \sum_{t=0}^{m-1} p_t x_t^s$</td>
<td></td>
</tr>
</tbody>
</table>

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 $x \in \mathbb{R}^d$,

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

**Lemma 4.2** (Expected residual. Lemma F.1). Let $v \sim \mathcal{D}$ be a sampling vector and assume that Assumption 5.1 holds. There exists $\rho \geq 0$ such that for all $x \in \mathbb{R}^d$,

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

$^3$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 4.1 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 \( \rho \) have simple closed formulae that depend on \( b \).

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

\[
\mathcal{L} = \mathcal{L}(b) \quad \text{def} = \frac{1}{\log_{\frac{b}{\rho}}(1 - \frac{b}{\rho})} \leq \frac{n - b}{b} L_{\max} + \frac{n b - 1}{b} L, \\
\rho = \rho(b) \quad \text{def} = \frac{1}{\frac{b}{\rho}} - \frac{1}{\log_{\frac{b}{\rho}}(1 - \frac{b}{\rho})} L_{\max}.
\]

(13)

(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 \( \rho \) 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 \( g(x, w) \equalsdef \nabla f_v(x) - \nabla f_v(w) + \nabla f(w) \). As a consequence of (11) and (12) we have that

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

(15)

Next we present a new Lyapunov 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

\[
\phi_s \equalsdef \frac{1}{2} \|x^n_s - x^*\|_2^2 + \psi_s \quad \text{where} \quad \psi_s \equalsdef 8\alpha \beta S_m(f(w_s) - f(x^*)).
\]

(16)

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

\[
\mathbb{E} [\phi_s] \leq \beta^s \phi_0, \quad \text{where} \quad \beta = \max \{1 - \alpha \mu, \frac{1}{2}\}.
\]

(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 3.2.

**Corollary 4.1.** Consider the setting of Algorithm 1 and suppose that we use \( b \)-nice sampling. Let \( \alpha = \frac{1}{2(\mathcal{L} + 2\rho)} \), 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 \( \mathbb{E} \left[ \|x^n_s - x^*\|_2^2 \right] \leq \epsilon \phi_0 \) is

\[
C_m(b) \quad \text{def} = 2 \left( \frac{n}{m} + 2b \right) \max \left\{ \frac{\mathcal{L}(b) + 2\rho(b)}{\mu}, m \right\} \log \left( \frac{1}{\epsilon} \right).
\]

(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.

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

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], [9], [12], and for a range of values of \( m \) including \( m = n \).

\[\text{6 Though the resulting complexity is 6 times the tightest gradient descent complexity, it is of the same order.}\]
Algorithm 2 L-SVRG-D

Parameters step size $\alpha, p \in (0, 1]$, and a sampling vector $v \sim D$

Initialization $w^0 = x^0 \in \mathbb{R}^d, \alpha_0 = \alpha$

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

Sample $v_k \sim D$

$g^k = \nabla f_{v_k}(x^k) - \nabla f_{v_k}(w^k) + \nabla f(w^k)$

$x^{k+1} = x^k - \alpha_k g^k$

$(w^{k+1}, \alpha_{k+1}) = \begin{cases} (x^k, \alpha) & \text{with probability } p \\ (w^k, \sqrt{1-p} \alpha_k) & \text{with probability } 1-p \end{cases}$


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

$$\phi^k \overset{\text{def}}{=} \|x^k - x^*\|_2^2 + \psi^k$$

where

$$\psi^k \overset{\text{def}}{=} \frac{8\alpha_k^2 \mathcal{L}}{p(3-2p)(f(w^k) - f(x^*))}, \quad \forall k \in \mathbb{N}. \quad (19)$$

If Assumption 3.1 holds and

$$\alpha \leq \frac{1}{2\zeta_p \mathcal{L}},$$

where

$$\zeta_p \overset{\text{def}}{=} \frac{(7-4p)(1-(1-p)^{\frac{3}{2}})}{p^2 - p}, \quad (20)$$

then

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

where

$$\beta = \max \left\{1 - \frac{2}{3} \alpha \mu, 1 - \frac{p}{2} \right\}. \quad (21)$$

Remark 5.1. To get a sense for the size of the step size given in (20), it is easy to show that $\zeta_p$ 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
We performed a series of experiments on data sets from LIBSVM [4] and the UCI repository [3], to validate our theoretical findings. We tested Proposition 6.2. Let

\[ C_p(b) \overset{\text{def}}{=} 2(2b + pn) \max \left\{ \frac{3c_p L(b)}{2 \mu} , \frac{1}{p} \right\} \log \left( \frac{1}{\epsilon} \right). \]

\section{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 \( L(b) \) 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).

\subsection{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).

\begin{proposition}
The loop size that minimizes (18) and the resulting total complexity is given by

\[ m^*(b) = \frac{L(b) + 2\rho(b)}{\mu} \quad \text{and} \quad C_{m^*}(b) = 2 \left( n + 2b \frac{L(b) + 2\rho(b)}{\mu} \right) \log \left( \frac{1}{\epsilon} \right). \]

\end{proposition}

For example when \( b = 1 \), we have that \( m^*(1) = 1 + L_{\max}/\mu \) 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 6.3.

\subsection{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 \( m \).

\begin{proposition}
Let \( b^* = \arg \min_{b \in [n]} C_m(b) \), 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

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

\end{proposition}

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.

\section{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 \( real-sim \),

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

\[ C_p(b) \overset{\text{def}}{=} 2(2b + pn) \max \left\{ \frac{3c_p L(b)}{2 \mu} , \frac{1}{p} \right\} \log \left( \frac{1}{\epsilon} \right). \]
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 icnn1 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 $m = 20L_{\text{max}}/\mu$ and $\alpha = 1/10L_{\text{max}}$, 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 I 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


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 C, we define several samplings. In Section D, we present the expected smoothness constant for the samplings we present. In Section E, we present the expected smoothness constant for the samplings we present.

A General properties

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

Lemma A.2. For any random vector \( X \in \mathbb{R}^d, E[\|X - E[X]\|^2] = E[\|X\|^2] - \|E[X]\|^2 \leq E[\|X\|^2]. \)

Lemma A.3. For any convex function \( f \), we have\[ f(y) \geq f(x) + \nabla f(x)^\top (y-x) \quad \forall x, y \in \mathbb{R}^d. \]

Lemma A.4 (Logarithm inequality).\[ \log(x) \leq x - 1 \quad \forall x > 0. \] (25)

Lemma A.5 (Complexity bounds). Consider the sequence \( (\alpha_k) \in \mathbb{R}^+ \) 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
\[ k \geq \frac{1}{1 - \rho} \log \left( \frac{1}{\epsilon} \right) \implies \alpha_k \leq \epsilon \alpha_0. \] (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 \( L_{\text{max}} = \max_{i \in [n]} L_i \). 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,
\[ nL \geq L_{\text{max}}. \] (27)

\[ \textbf{Proof.} \] Let \( x, y \in \mathbb{R}^d \). Since \( f \) is \( L \)-smooth, we have\[ f(x) \leq f(y) + \nabla f(y)^\top (x-y) + \frac{L}{2} \|x-y\|^2. \]
Hence, multiplying by \( n \) on both sides,
\[ \sum_{i=1}^{n} f_i(x) \leq \sum_{i=1}^{n} f_i(y) + \sum_{i=1}^{n} \nabla f_i(y)^\top (x-y) + \frac{nL}{2} \|x-y\|^2. \]
Rearranging this inequality,
\[ \sum_{i=1}^{n} (f_i(x) - f_i(y) - \nabla f_i(y)^\top (x-y)) \leq \frac{nL}{2} \|x-y\|^2. \] (28)
Since the functions \( f_i \) are convex, we have for all \( i \in [n], \)
\[ f_i(x) \leq f_i(y) - \nabla f_i(y)^\top (x-y) \geq 0. \]
Then, as a consequence of (28), we have that for all \( i \in [n], \)
\[ f_i(x) - f_i(y) - \nabla f_i(y)^\top (x-y) \leq \frac{nL}{2} \|x-y\|^2. \]
Rearranging this inequality,
\[ f_i(x) \leq f_i(y) + \nabla f_i(y)^\top (x-y) + \frac{nL}{2} \|x-y\|^2. \]
But since for all \( i \in [n], \) \( L_i \) is the smallest positive constant that verifies
\[ f_i(x) \leq f_i(y) + \nabla f_i(y)^\top (x-y) + \frac{L_i}{2} \|x-y\|^2, \]
we have for all \( i \in [n], \) \( L_i \leq nL \). Hence \( L_{\text{max}} \leq nL. \)
B Proofs of the results of the main paper

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

B.1 Proof of Lemma 4.4

Proof.

\[
\mathbb{E}_D \left[ \|g(x, w)\|_2^2 \right] = \mathbb{E}_D \left[ \|\nabla f_v(x) - \nabla f_v(x^*) + \nabla f_v(x^*) - \nabla f_v(w) + \nabla f(w)\|_2^2 \right] \\
\overset{\text{Lem.}\ref{lem:4.4}}{\leq} 2\mathbb{E}_D \left[ \|\nabla f_v(x) - \nabla f_v(x^*)\|_2^2 \right] \\
+ 2\mathbb{E}_D \left[ \|\nabla f_v(w) - \nabla f_v(x^*) - \nabla f(w)\|_2^2 \right].
\]

Since $\mathbb{E}_D [\nabla f_v(w) - \nabla f_v(x^*)] = \nabla f(w)$, we have

\[
\mathbb{E}_D \left[ \|\nabla f_v(w) - \nabla f_v(x^*) - \nabla f(w)\|_2^2 \right] \overset{\text{Lem.}\ref{lem:4.2}}{\leq} 2\mathbb{E}_D \left[ \|\nabla f_v(w) - \nabla f_v(x^*)\|_2^2 \right].
\]

Hence,

\[
\mathbb{E}_D \left[ \|g(x, w)\|_2^2 \right] \leq 2\mathbb{E}_D \left[ \|\nabla f_v(x) - \nabla f_v(x^*)\|_2^2 \right] + 2\mathbb{E}_D \left[ \|\nabla f_v(w) - \nabla f_v(x^*)\|_2^2 \right] \\
\overset{\text{(11) + (12)}}{\leq} 4\mathcal{L}(f(x) - f(x^*)) + 4\rho(f(w) - f(x^*)).
\]

\[\square\]

B.2 Proof of Theorem 4.1

Proof.

\[
\mathbb{E}_t \left[ \|x^{t+1}_s - x^*\|_2^2 \right] = \mathbb{E}_t \left[ \|x^t_s - x^* - \alpha g^t_s\|_2^2 \right] \\
= \|x^t_s - x^*\|_2^2 - 2\alpha \mathbb{E}_t \left[ g^t_s \right]^\top (x^t_s - x^*) + \alpha^2 \mathbb{E}_t \left[ \|g^t_s\|_2^2 \right] \\
\overset{\text{(8) + (13)}}{\leq} \|x^t_s - x^*\|_2^2 - 2\alpha \nabla f(x^t_s)^\top (x^t_s - x^*) \\
+ 2\alpha^2 \left[ 2\mathcal{L}(f(x^t_s) - f(x^*)) + 2\mathcal{L}(f(w_{s-1}) - f(x^*)) \right] \\
\overset{\text{(9)}}{\leq} (1 - \alpha\mu) \|x^t_s - x^*\|_2^2 - 2\alpha(1 - 2\alpha\mathcal{L}) (f(x^t_s) - f(x^*)) \\
+ 4\alpha^2 \rho(f(w_{s-1}) - f(x^*)). \quad (29)
\]

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

\[
\alpha \overset{\text{Lemma}\ref{lem:4.3}}{\leq} \frac{1}{2\mu}.
\]
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
\[
\mathbb{E} \left[ \|x_s^m - x^*\|_2^2 \right] \leq \left( 1 - \alpha \mu \right)^m \mathbb{E} \left[ \|x_s^{m-1} - x^*\|_2^2 \right]
- 2\alpha (1 - 2\alpha \mathcal{L}) \sum_{t=0}^{m-1} (1 - \alpha \mu)^{m-1-t} \mathbb{E} \left[ f(x_s^t) - f(x^*) \right]
+ 4\alpha^2 \rho \mathbb{E} \left[ f(w_{s-1}) - f(x^*) \right] \sum_{t=0}^{m-1} (1 - \alpha \mu)^{m-1-t}
\]

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

Since \(f\) is convex, we have by Jensen’s inequality that
\[
f(w_s) - f(x^*) = f\left( \sum_{t=0}^{m-1} p_t x_s^t \right) - f(x^*)
\leq \sum_{t=0}^{m-1} p_t (f(x_s^t) - f(x^*)).
\] (31)

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

As a result,
\[
\mathbb{E} \left[ \phi_s \right] \aleq (1 - \alpha \mu)^m \mathbb{E} \left[ \|x_{s-1}^m - x^*\|_2^2 \right] + \frac{1}{2} \mathbb{E} \left[ \psi_{s-1} \right]
- 2\alpha (1 - 2\alpha (\mathcal{L} + 2\rho)) S_m \sum_{t=0}^{m-1} p_t \mathbb{E} \left[ (f(x_s^t) - f(x^*)) \right].
\]

Since \(\alpha \leq \frac{1}{2\mathcal{L}}\), the above implies
\[
\mathbb{E} \left[ \phi_s \right] \leq (1 - \alpha \mu)^m \mathbb{E} \left[ \|x_{s-1}^m - x^*\|_2^2 \right] + \frac{1}{2} \mathbb{E} \left[ \psi_{s-1} \right]
\leq \beta \mathbb{E} \left[ \phi_{s-1} \right],
\]
where \(\beta = \max \left\{ (1 - \alpha \mu)^m, \frac{1}{2} \right\}\).

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. \(\square\)

**B.3 Proof of Corollary 4.1**

*Proof.* Noting \(\beta = \max \left\{ \left(1 - \frac{\mu}{2(1+2\rho b)} \right)^m, \frac{1}{2} \right\}\), 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 \(f_t\) gradients, and in each outer

14
Iteration we evaluate all $n$ gradients, this means that the total complexity will be given by

$$C \overset{\text{def}}{=} (n + 2bm) \frac{\log(1/\epsilon)}{\log(1/\beta)} = \max\left\{-\frac{n + 2bm}{m \log(1 - \frac{\mu}{2\rho(b) + \mu b})}, \frac{n + 2bm}{\log 2}\right\} \log \left(\frac{1}{\epsilon}\right).$$

\[\text{(25)}\]

B.4 Proof of Corollary 4.3

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

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

When $n \geq \frac{L_{\text{max}}\mu}{2}$. Then, $m \in \left[\frac{L_{\text{max}}\mu}{n}, n\right]$. We can rewrite $C_m(1)$ as

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

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

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

When $n \leq \frac{L_{\text{max}}\mu}{2}$. Then, $m \in \left[n, \frac{L_{\text{max}}\mu}{n}\right]$. We have $\frac{n}{m} \leq 1$ and $m \leq \frac{3L_{\text{max}}}{\mu}$. Hence,

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

\[\text{(36)}\]

B.5 Proof of Theorem 5.1

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

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

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

$$E[\alpha_k^2] = \frac{1 + (1 - p)^{2k+1}}{2 - p} \alpha^2. \quad \text{(34)}$$

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

$$E_p[\alpha_{k+1}] = (1 - p)\sqrt{1 - p} \alpha_k + p\alpha. \quad \text{(35)}$$

Then taking total expectation

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

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

$$E[\alpha_k] = \frac{(1 - p)^{\frac{3k+2}{2}} (1 - \sqrt{1 - p}) + p}{1 - (1 - p)^{\frac{3}{2}}} \alpha. \quad \text{(37)}$$
Indeed, applying (36) recursively gives

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

Adding up the geometric series gives

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

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

We now present a proof of Theorem 5.1.

**Proof of Theorem 5.1.** First,

\[
\mathbb{E}_k [\|x^{k+1} - x^*\|_2^2] = \mathbb{E}_k [\|x^k - x^* - \alpha_k g_k\|_2^2] = \|x^k - x^*\|_2^2 - 2\alpha_k \mathbb{E}_k [g_k^T (x^k - x^*)] + \alpha_k^2 \mathbb{E}_k [\|g_k\|_2^2] \leq \|x^k - x^*\|_2^2 - 2\alpha_k \mathbb{E}_k [\nabla f(x^k)^T (x^k - x^*)] + 2\alpha_k^2 \mathbb{E}_k [2\mathcal{L}(f(x^k) - f(x^*)) + 2\mathcal{L}(f(w^k) - f(x^*))]
\]

\[
\leq (1 - \alpha_k \mu) \|x^k - x^*\|_2^2 - 2\alpha_k (1 - 2\alpha_k \mathcal{L}) (f(x^k) - f(x^*)) + 4\alpha_k^2 \mathcal{L}(f(w^k) - f(x^*)) + (1 - \alpha_k \mu) \|x^k - x^*\|_2^2 - 2\alpha_k (1 - 2\alpha_k \mathcal{L}) (f(x^k) - f(x^*)) + p \left(\frac{3}{2} - p\right) \psi^k.
\]

Hence we have, taking total expectation and noticing that the variables \( \alpha_k \) and \( x^k \) are independent,

\[
\mathbb{E} [\|x^{k+1} - x^*\|_2^2] \leq (1 - \mathbb{E} [\alpha_k] \mu) \mathbb{E} [\|x^k - x^*\|_2^2] - 2\mathbb{E} [\alpha_k (1 - 2\alpha_k \mathcal{L})] \mathbb{E} [f(x^k) - f(x^*)] + p \left(\frac{3}{2} - p\right) \mathbb{E} [\psi^k]. \tag{38}
\]

We have also have

\[
\mathbb{E}_k [\psi^{k+1}] = (1 - p) \frac{8(1 - p)\alpha_k^2 \mathcal{L}}{p(3 - 2p)} (f(w^k) - f(x^*)) + p \frac{8\alpha_k^2 \mathcal{L}}{p(3 - 2p)} (f(x^k) - f(x^*))
\]

\[
= (1 - p)^2 \psi^k + \frac{8\alpha_k^2 \mathcal{L}}{3 - 2p} (f(x^k) - f(x^*)).
\]

Hence, taking total expectation gives

\[
\mathbb{E} [\psi^{k+1}] = (1 - p)^2 \mathbb{E} [\psi^k] + \frac{8\alpha_k^2 \mathcal{L}}{3 - 2p} \mathbb{E} [f(x^k) - f(x^*)]. \tag{39}
\]
Consequently, 

\[ \mathbb{E} [\phi^k] \leq (1 - \mathbb{E} [\alpha_k \mu]) \mathbb{E} \left[ ||x^k - x^*||^2_2 \right] 
- 2 \left( \mathbb{E} [\alpha_k (1 - 2\alpha_k \mathcal{L})] - 4 \frac{\alpha^2 \mathcal{L}}{3 - 2p} \right) \mathbb{E} [f(x^k) - f(x^*)] 
+ \left( 1 - \frac{p}{2} \right) \mathbb{E} [\psi^k] 
= (1 - \mathbb{E} [\alpha_k \mu]) \mathbb{E} \left[ ||x^k - x^*||^2_2 \right] 
- 2 \left( \mathbb{E} [\alpha_k] - 2 \left( \mathbb{E} [\alpha_k^2] + \frac{2}{3 - 2p} \alpha^2 \right) \mathcal{L} \right) \mathbb{E} [f(x^k) - f(x^*)] 
+ \left( 1 - \frac{p}{2} \right) \mathbb{E} [\psi^k]. \] 

(40)

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

\[ \mathbb{E} [\alpha_k] \geq \frac{2}{3} \alpha, \] 

(41)

Letting \( q = 1 - p \) we have that

\[ \frac{(1-p)^{3k+2} (1 - \sqrt{1-p}) + p}{1 - (1-p)^2} = \frac{q^{3k+2} (1 - \sqrt{q}) + 1 - q}{1 - q^2} \]

\[ = q^{\frac{3k+2}{2}} \left( 1 - \sqrt{q} \right) + \frac{1 - q}{1 - q^{3/2}} \geq \frac{1 - q}{1 - q^{3/2}} \forall q \geq 0. \]

Consequently,

\[ \mathbb{E} [\phi^k] \leq \left( 1 - \frac{2}{3} \alpha \mu \right) \mathbb{E} \left[ ||x^k - x^*||^2_2 \right] 
- 2 \left( \mathbb{E} [\alpha_k] - 2 \left( \mathbb{E} [\alpha_k^2] + \frac{2}{3 - 2p} \alpha^2 \right) \mathcal{L} \right) \mathbb{E} [f(x^k) - f(x^*)] 
+ \left( 1 - \frac{p}{2} \right) \mathbb{E} [\psi^k]. \] 

(42)

To declutter the notations, Let us define

\[ a_k \overset{\text{def}}{=} \frac{(1-p)^{3k+2} (1 - \sqrt{1-p}) + p}{1 - (1-p)^2}, \]

\[ b_k \overset{\text{def}}{=} \frac{1 + (1-p)^{2k+1}}{2 - p}, \]

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

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

(45)

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

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

\[
2\mathcal{L}\alpha \leq \frac{q^{3k+2}(1 - \sqrt{q}) + 1 - q}{1 - q^2} \leq \frac{q^{3k+2}(1 - \sqrt{q}) + 1 - q}{1 - q^2}.
\] (47)

We are now going to show that:

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

Indeed, multiplying out the denominators of the above gives

\[
F(q) \overset{\text{def}}{=} (3 + 4q) \left( q^{3k+2}(1 - \sqrt{q}) + 1 - q \right) - (1 - q) \left( q^{2k+1}(1 + 2q) + 3 + 4q \right)
\]

\[
= q^{3k+2}(1 - \sqrt{q})(3 + 4q) - q^{2k+1}(1 + 2q)(1 - q)
\]

\[
= q^{3k+2}(1 - \sqrt{q}) \left( 3 + 4q - q^2(1 + 2q)(1 + \sqrt{q}) \right).
\]

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

\[
F(q) \geq q^{3k+2}(1 - \sqrt{q})(3 + 4q - (1 + 2q)(1 + \sqrt{q}))
\]

\[
= 2q^{3k+2}(1 - \sqrt{q})(1 - q\sqrt{q})
\]

\[
\geq 0.
\]

As a result (48) holds, and thus if

\[
2\mathcal{L}\alpha \leq \frac{1 - q}{3 + 4q} \leq \frac{1 - q}{1 - q^2}
\]

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 \( a_k - 2\alpha(b_k + 2\alpha)\mathcal{L} \geq 0 \) and from (46) we have that

\[
E[\phi^{k+1}] \overset{\text{Lemma A.5}}{\leq} \left( 1 - \frac{2}{3}\alpha\mu \right) E[\|x^k - x^*\|^2] + \left( 1 - \frac{p}{2} \right) E[\psi^k]
\]

\[
\leq \beta E[\phi^{k+1}],
\] (49)

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

\[\Box\]

### B.6 Proof of Corollary 5.1

**Proof.** We have that

\[
E[\phi^k] \leq \beta^k \phi^0,
\]

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

\[
2\max\left\{ \frac{3\alpha}{2} \frac{L(b)}{\mu}, \frac{1}{p} \right\} \log \left( \frac{1}{\epsilon} \right).
\]

For the total complexity, one can notice that in expectation, we compute \( 2b + pm \) stochastic gradients at each iteration. \( \Box \)
We distinguish three cases:

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 we have from Lemma 4.3:

\begin{align*}
\mathcal{L}(b) &= \frac{1}{b} \frac{n - b}{b - 1} L_{\text{max}} + \frac{n b - 1}{b - 1} L, \quad (50) \\
\rho(b) &= \frac{1}{b} \frac{n - b}{b - 1} L_{\text{max}}. \quad (51)
\end{align*}

For brevity, 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 $m$:

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

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

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

\[ C_n(b) \overset{\text{def}}{=} 2(2b + 1) \max \{\kappa(b), n\}, \quad (53) \]

Writing $\kappa(b)$ explicitly:

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

Since $3L_{\text{max}} > L$, $\kappa(b)$ 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:

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

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

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

we can see that:

- If $n \leq \frac{3L_{\text{max}}}{\mu}$, $C_n(b)$ is a decreasing function, hence $b^* = n$. 


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

\[
\hat{b} = \sqrt{\frac{n}{2nL - 3L_{\text{max}}}}.
\]

Hence,

\[
b^* = \left\lfloor \hat{b} \right\rfloor.
\]

- If \( n \geq \frac{3L_{\text{max}}}{\mu} \), then \( \kappa(1) = \frac{3L_{\text{max}}}{\mu} \). 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 \( \frac{L}{\mu} < n < \frac{3L_{\text{max}}}{\mu} \), 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

\[
\tilde{b} = \frac{(3L_{\text{max}} - L)n}{n(n - 1)\mu - nL + 3L_{\text{max}}}.
\]

Define \( G(b) : = (2b + 1)\kappa(b) \). Then,

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

As a result, we have that

- if \( n \leq \frac{3L_{\text{max}}}{\mu} \), \( G(b) \) 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 \( n > \frac{3L_{\text{max}}}{\mu} \), \( G(b) \) is decreasing on \([1, \tilde{b}]\) and increasing on \([\tilde{b}, n]\). Hence \( C_n(b) \) is decreasing on \([1, \min\{\tilde{b}, b\}]\) and increasing on \([\min\{\tilde{b}, b\}, 1]\). Then,

\[
b^* = \left\lfloor \min\{\tilde{b}, b\} \right\rfloor.
\]

To summarize, we have for \( m = n/b \),

\[
b^* = \begin{cases} 1 & \text{if } n \geq \frac{3L_{\text{max}}}{\mu}, \\ \min\{\tilde{b}, \tilde{b} \} & \text{if } \max\{\frac{L}{\mu}, \frac{3L_{\text{max}}}{L} \} < n < \frac{3L_{\text{max}}}{\mu}, \\ \tilde{b} & \text{if } \frac{3L_{\text{max}}}{L} < n < \frac{L}{\mu}, \\ \tilde{b} & \text{if } \frac{L}{\mu} < n \leq \frac{3L_{\text{max}}}{L}, \\ n & \text{otherwise, if } n \leq \min\{\frac{L}{\mu}, \frac{3L_{\text{max}}}{L} \}. \end{cases}
\]

When \( m = n/b \). In this case we have

\[
C_m(b) = 6 \max\{bn\kappa(b), n\},
\]

with

\[
bn\kappa(b) = \frac{1}{\mu(n - 1)} ((3L_{\text{max}} - L)n + (nL - 3L_{\text{max}})b),
\]

and thus \( \kappa(1) = \frac{3L_{\text{max}}}{\mu} \) and \( n\kappa(n) = \frac{2L}{\mu} \geq n \). We distinguish two cases:

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

\[
b^* = n
\]
• if \( n > \frac{3L_{\text{max}}}{L} \), \( b_*(b) \) is increasing in \( b \). Thus,
  - if \( n \leq \frac{3L_{\text{max}}}{L} = \kappa(1) \), then \( C_m(b) = 6b_*(b) \). Hence \( b^* = 1 \).
  - if \( n > \frac{3L_{\text{max}}}{L} \), using the definition of \( \bar{b} \) in Equation (54), we have that
    \[
    C_m(b) = \begin{cases} 
    6n & \text{for } b \in [1, \bar{b}], \\
    6b_*(b) & \text{for } b \in [\bar{b}, n], 
    \end{cases}
    \]
where
\[
\bar{b} = \frac{n(n-1)\mu - (3L_{\text{max}} - L)n}{nL - 3L_{\text{max}}}
\]
is the batch size \( b \in [n] \) which verifies \( b_*(b) = n \). Hence \( b^* \) can be any point in \( \{1, \ldots, \bar{b} \} \). In light of shared memory parallelism, \( b^* = \lceil \bar{b} \rceil \) would be the most practical choice.

\[ \square \]

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 \( b^* \overset{\text{def}}{=} \arg \min_{b \in [n]} C_p(b) \), where \( C_p(b) \) is defined in (22). For the widely used choice \( p = \frac{1}{n} \), we have that
\[
b^* = \begin{cases} 
1 & \text{if } n \geq \frac{3\zeta_p}{2} \frac{L_{\text{max}}}{\mu} \\
\min(\bar{b}, \hat{b}) & \text{if } \frac{3\zeta_p}{2} \frac{L}{\mu} < n < \frac{3\zeta_p}{2} \frac{L_{\text{max}}}{\mu}, \\
\lceil \frac{\min(\bar{b}, \hat{b})}{n} \rceil & \text{otherwise, if } n \leq \frac{3\zeta_p}{2} \frac{L}{\mu},
\end{cases}
\tag{57}
\]
where \( \zeta_p \) is defined in (20) for \( p \in (0, 1] \).

Because \( \zeta_p \) depends on \( p \), optimizing the total complexity with respect to \( b \) for the case \( p = \frac{1}{n} \) 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 \( \log \left( \frac{L}{\mu} \right) \) 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 \( \pi(b) \overset{\text{def}}{=} \frac{3\zeta_p}{2} \frac{L(b)}{\mu^*} \). We have
\[
\pi(b) = \frac{3\zeta_p}{2} \frac{1}{\mu(n-1)} \left( \frac{n(L_{\text{max}} - L)}{b} + nL - L_{\text{max}} \right).
\tag{59}
\]
Since \( L_{\text{max}} \geq L \), \( \pi(b) \) is a decreasing function on \([1, n]\). We distinguish three cases:

• if \( n > \kappa(1) = \frac{3\zeta_p}{2} \frac{L_{\text{max}}}{\mu} \), then for all \( b \in [1, n] \), \( n > \kappa(b) \). Hence,
  \[
  C_{1/n}(b) = 2(2b + 1)n.
  \]
  \( C_{1/n}(b) \) is an increasing function of \( b \). Hence
  \[
  b^* = 1.
  \]

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

\[
\hat{b} = \frac{\frac{3\xi}{2} n(L_{\max} - L)}{\mu(n - 1) - \frac{3\xi}{2} (nL - L_{\max})}.
\]

Now, consider the function

\[
G(b) \overset{\text{def}}{=} (2b + 1)\pi(b)
\]

\[
= \frac{3\xi}{2} \frac{1}{\mu(n - 1)} \left( 2(nL - L_{\max})b + \frac{n(L_{\max} - L)}{b} \right) + \Omega,
\]

where \( \Omega \) replaces constants which don’t depend on \( b \). The first derivative of \( G(b) \) is

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

and its second derivative is

\[
G''(b) = \frac{3\xi \rho n(L_{\max} - L)}{\mu(n - 1)b^3} \geq 0.
\]

\( G(b) \) 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 \( [1, \min \{ \hat{b}, \tilde{b} \}] \) and increasing on \( [\min \{ \hat{b}, \tilde{b} \}, n] \). Hence,

\[
b^* = \lfloor \min \{ \hat{b}, \tilde{b} \} \rfloor.
\]

\( \square \)

## 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** \( \Pr(|S| = 1) = 1 \) **and**

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

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

\[
p_B \overset{\text{def}}{=} \Pr(S = B) > 0 \quad \forall B \in B, \text{ and } \sum_{B \in B} p_B = 1.
\]

**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 \( \mathcal{E} \) is given by

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

**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

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

Taking expectation over the sampling gives

$$\mathbb{E} \left[ \|\nabla f_v(x) - \nabla f_v(x^\ast)\|_2^2 \right] \leq 2 \mathbb{E} \left[ L_v (f_v(x) - f_v(x^\ast) - \langle \nabla f_v(x^\ast), x - x^\ast \rangle) \right]$$

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

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

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

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

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

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 $\mu$-strongly convex, then $\mathcal{L} \geq \mu$.

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

$$\mathbb{E} \left[ \|\nabla f_v(x) - \nabla f_v(x^\ast) - \nabla f(x)\|_2^2 \right] = \mathbb{E} \left[ \|\nabla f_v(x) - \nabla f_v(x^\ast)\|_2^2 - \|\nabla f(x)\|_2^2 \right] \overset{(11)}{\leq} 2(\mathcal{L} - \mu)(f(x) - f(x^\ast)). \tag{62}$$

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

**Remark E.1.** Consider the expected residual constant $\rho$ defined in (2). This constant verifies for all $x \in \mathbb{R}^d,$

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

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** (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

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

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 $L$ are

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

Consequently,

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

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

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

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

**Lemma E.6** (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

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

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

F Expected residual

In this section, we compute bounds on the expected residual $\rho$ 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

$$\rho = \frac{\lambda_{\text{max}}(\text{Var}[v])}{\lambda_{\text{max}}},$$

(63)

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

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 $A \succeq 0$. Then,

$$\text{Tr}(AB) \leq \lambda_{\text{max}}(B)\text{Tr}(A).$$
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 $V_i \defeq \sqrt{\lambda_i(A)} U_i$ for all $i \in [n]$, we can write $A = \sum_{i=1}^{n} V_i V_i^\top$. Then,

$$
\text{Tr} (AB) = \text{Tr} \left( \sum_{i=1}^{n} V_i V_i^\top B \right) = \sum_{i=1}^{n} \text{Tr} \left( V_i V_i^\top B \right) = \sum_{i=1}^{n} \text{Tr} \left( V_i^\top B V_i \right) = \sum_{i=1}^{n} V_i^\top B V_i
$$

\[ \leq \lambda_{\text{max}}(B) \sum_{i=1}^{n} V_i^\top V_i = \lambda_{\text{max}}(B) \text{Tr} (A), \]

where we use that $B \preceq \lambda_{\text{max}}(B) I_n$. \hfill \qed

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:

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

Let us expand the squared norm first. Define $D F(w)$ as the Jacobian of $F(w) \defeq [f_1(w), \ldots, f_n(w)]$. We denote $R \defeq (D F(w) - D F(x^*))$

$$
C \defeq \| \nabla f_v(w) - \nabla f_v(x^*) - (\nabla f(w) - \nabla f(x^*)) \|^2 \\
= \frac{1}{n^2} \| (D F(w) - D F(x^*)) (v - 1) \|^2 \\
= \frac{1}{n^2} (R(v - 1), R(v - 1))_{\mathbb{R}^d} \\
= \frac{1}{n^2} \text{Tr} \left( (v - 1)^\top R^\top R (v - 1) \right) \\
= \frac{1}{n^2} \text{Tr} \left( R^\top R (v - 1)(v - 1)^\top \right).
$$

Taking expectation,

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

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

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

Therefore,

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

Which means

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

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

\[
(\text{Var} [v])_{ij} = \begin{cases} 
\frac{1}{p_i} - 1 & \text{if } i = j \\
\frac{1}{P_{ij}} - 1 & \text{otherwise,}
\end{cases}
\]  

(69)

with

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

(70)

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

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

### F.1 Expected residual for uniform $b$-nice sampling

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

\[ \rho = \frac{n - b}{(n - 1)b} L_{\text{max}}. \]

(71)

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

\[
\forall i \in [n], p_i = \frac{c_1}{|B|},
\]

\[
\forall i, j \in [n], P_{ij} = \frac{c_2}{|B|},
\]

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

\[
\text{Var} [v] = \begin{bmatrix}
\frac{|B|}{c_1} & \frac{|B|c_2}{c_1} - 1 & \cdots & \frac{|B|c_2}{c_1} - 1 \\
\frac{|B|c_2}{c_1} - 1 & \frac{|B|}{c_1} & \cdots & \frac{|B|c_2}{c_1} - 1 \\
\vdots & \ddots & \ddots & \vdots \\
\frac{|B|c_2}{c_1} - 1 & \cdots & \frac{|B|c_2}{c_1} - 1 & \frac{|B|}{c_1} - 1
\end{bmatrix}.
\]

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

\[
\left( \frac{|B|}{c_1} - 1, \frac{|B|c_2}{c_1^2} - 1, \ldots, \frac{|B|c_2}{c_1^2} - 1 \right),
\]

and, as such, it has two eigenvalues

\[
\lambda_1 \overset{\text{def}}{=} \frac{|B|}{c_1} \left( 1 + (n - 1) \frac{c_2}{c_1} \right) - n = 0,
\]

\[
\lambda_2 \overset{\text{def}}{=} \frac{|B|}{c_1} \left( 1 - \frac{c_2}{c_1} \right) = \frac{n(n - b)}{b(n - 1)}.
\]

(72)

Hence, the expected residual can be computed explicitly as

\[ \rho = \frac{n - b}{(n - 1)b} L_{\text{max}}. \]

(73)

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

\[ \square \]
F.2 Expected residual for uniform partition sampling

**Lemma F.4 (ρ for uniform partition sampling).** Suppose that \( b \) divides \( n \) 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 \( L_{\text{max}} \)-smooth, then,

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

**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 \( k \in \left[ \frac{n}{b} \right] \),

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

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

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

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

\[
V_k = \left( \frac{n}{b} - 1 \right) 1_k \, 1_k^\top \in \mathbb{R}^{b \times b}.
\]

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

\[
\rho = \left( 1 - \frac{b}{n} \right) L_{\text{max}}. \tag{77}
\]

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

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 \( L_{\text{max}} \)-smooth, then

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

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

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

Thus, according to (69):

\[
\text{Var}[v] = \text{Diag} \left( \frac{1}{p_1} - 1, \frac{1}{p_2} - 1, \ldots, \frac{1}{p_n} - 1 \right),
\]

whose largest eigenvalue is

\[
\lambda_{\text{max}}(\text{Var}[v]) = \max_{i \in [n]} \frac{1}{p_i} - 1 = \frac{1}{\min_{i \in [n]} p_i} - 1.
\]

27
Consequently,

\[
\rho = \left( \frac{1}{\min_i p_i} - 1 \right) \frac{L_{\max}}{n}.
\] (79)

If \( p_i = \frac{1}{n} \) for all \( i \in [n] \), which corresponds in expectation to uniform single-element sampling SVRG since \( E[|S|] = 1 \), 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 \( E[|S|] = b \). 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 \( p_i = P[i \in S] \) for all \( i \in [n] \). If \( \sum_{i=1}^{n} p_i = b \), then \( E[|S|] = b \).

**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

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

**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 \( L \) as the expected residual constant. Thus, we simply use the expected smoothness constant from Lemma E.4.

**Lemma F.7** (\( \rho \) 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.
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 $m^* = \frac{3L_{\text{max}}}{\mu}$ that we derived in Proposition 6.1. We set $p = \frac{1}{m^*}$ 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 $m^*$, $L_{\text{max}}/\mu$, $3L_{\text{max}}/\mu$ 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 $n$, $L_{\text{max}}/\mu$ 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).
Figure 5: $b = 1$, $m = n$ for Free-SVRG and L-SVRG-D, slice data set.

Figure 6: $b = 1$, $m = n$ for Free-SVRG and L-SVRG-D, ijcnnI data set.
Figure 7: $b = 1, m = n$ for Free-SVRG and L-SVRG-D, real-sim data set.

Figure 8: Optimal mini-batching when theoretically available, YearPredictionMSD data set.
Figure 9: Optimal mini-batching when theoretically available, *slice* data set.

Figure 10: Optimal mini-batching when theoretically available, *ijcnn1* data set.
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) $\lambda = 10^{-3}$

---

Free-SVRG ($b = 1, m = 20L_{\max}/\mu$) Free-SVRG ($b = 1, m = 3L_{\max}/\mu$) L-SVRG-D ($b = 1, p = 3L_{\max}/\mu$)
Figure 13: Optimal inner loop size when theoretically available, slice data set.

Figure 14: Optimal inner loop size when theoretically available, ijcnn1 data set.
Figure 15: Optimal inner loop size when theoretically available, real-sim data set.

Figure 16: Impact of the mini-batch size on Free-SVRG for the YearPredictionMSD data set.
Figure 17: Impact of the mini-batch size on Free-SVRG for the slice data set.

Figure 18: Impact of the mini-batch size on Free-SVRG for the ijcn1 data set.
Figure 19: Impact of the mini-batch size on Free-SVRG for the real-sim data set.

Figure 20: Impact of the inner loop size on Free-SVRG for the YearPredictionMSD data set.
Figure 21: Impact of the inner loop size on Free-SVRG for the slice data set.
\[ m = n \quad m = 2n \quad m = L_{\max}/\mu \quad m = 3L_{\max}/\mu \]

Figure 22: Impact of the inner loop size on Free-SVRG for the \textit{ijcnn1} data set.

Figure 23: Impact of the inner loop size on Free-SVRG for the \textit{real-sim} data set.