Random-reshuffled SARAH does not need full gradient computations

Aleksandr Beznosikov ANBEZNOSIKOV@GMAIL.COM
Moscow Institute of Physics and Technology (MIPT), Moscow, Russian Federation
Higher School of Economics (HSE University), Moscow, Russian Federation
Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), Masdar City, Abu Dhabi, UAE

Martin Takáč TAKAC.MT@GMAIL.COM
Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), Masdar City, Abu Dhabi, UAE

Abstract

The StochAstic Recursive grAdient algoritHm (SARAH) algorithm is a variance reduced variant of the Stochastic Gradient Descent (SGD) algorithm that needs a gradient of the objective function from time to time. In this paper, we remove the necessity of a full gradient computation. This is achieved by using a randomized reshuffling strategy and aggregating stochastic gradients obtained in each epoch. The aggregated stochastic gradients serve as an estimate of a full gradient in the SARAH algorithm. We provide a theoretical analysis of the proposed approach and conclude the paper with numerical experiments that demonstrate the efficiency of this approach.

1. Introduction

In this paper we address the problem of minimizing a finite-sum problem of the form

$$\min_{w\in\mathbb{R}^d}\left\{P(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w)\right\}, \tag{1}$$

where $\forall i \in [n] := \{1,2,\ldots,n\}$ the f_i is a convex function. We will further assume that $w^* = \arg\min P(w)$ exists.

Problems of this form are very common in e.g., supervised learning [25]. Let a training dataset consists of n pairs, i.e., $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ is a feature vector for a datapoint i and y_i is the corresponding label. Then for example, the least squares regression problem corresponds to (1) with $f_i(w) = \frac{1}{2}(x_i^T w - y_i)^2$. If $y_i \in \{-1, 1\}$ would indicate a class, then a logistic regression is obtained by choosing $f_i(w) = \log(1 + \exp(-y_i x_i^T w))$.

Recently, many algorithms have been proposed for solving (1). In this paper, we are interested in a subclass of these algorithms that fall into a stochastic gradient descent (SGD) framework originating from a work of Robbins and Monro in '50s [23]. Let v_t will be some sort of (possibly stochastic and very rough) approximation of $\nabla P(w_t)$, then many SGD type algorithms update the w as follows:

$$w_{t+1} = w_t - \eta_t v_t, \tag{2}$$

where $\eta_t > 0$ is a predefined step-size. The classical SGD defines $v_t = \nabla f_i(w_t)$, where $i \in [n]$ is chosen randomly [26] or its mini-batch version [27], where $v_t = \frac{1}{|S_t|}\sum_{i\in S_t}\nabla f_i(w_t)$, with $S_t \subset [n]$. Even with an unbiased gradient estimates of SGD, where $\mathbb{E}[v_t|w_t] = \nabla P(w_t)$, the variance of v_t is the main source of slower convergence [2, 9, 19].

1.1. Brief literature review

Recently, many variance-reduced variants of SGD have been proposed, including SAG/SAGA [5, 22, 24], SVRG [1, 8, 30], MISO [14], SARAH [7, 17, 18, 21], SPIDER [6], STORM [4], PAGE [12], and many others. Generally speaking, the variance reduced variants of SGD still aim to sample $\mathcal{O}(1)$ functions and use their gradients to update v_t. For example, SVRG [8] will fix a point $\tilde{w}$, in which a full gradient $\nabla P(\tilde{w})$ is computed and subsequently stochastic gradient is defined as $v_t = \nabla f_i(w_t) - \nabla f_i(\tilde{w}) + \nabla P(\tilde{w})$, where $i \in [n]$ is picked at random.

SARAH Algorithm. The SARAH algorithm [17], on the other hand, updates v_t recursively. It starts with a full gradient computation $v_0 = \nabla P(w_0)$, then taking a step (2) and updating the gradient estimate recursively as $v_{t+1} = \nabla f_i(w_t) - \nabla f_i(w_{t-1}) + v_t$. For smooth and strongly convex problem, the procedure highlighted above converge, but not to the optimal solution w^*. Therefore, similarly to SVRG, the process is restarted after i) a predefined number of iterations, ii) randomly [10–12], or iii) decided in run-time by computing the ratio $\|v_t\|/\|v_0\|$ (SARAH+ [17]), and a new full gradient estimate has to be computed. To elevate this issue, e.g., in [21], they proposed inexact SARAH (iSARAH), where the full gradient estimate is replaced by a mini-batch gradient estimate $v_0 = \frac{1}{|S|}\sum_{i\in S} f_i(w_0)$, where $S \subset [n]$. To find a point $\hat{w}$ such that $\|\nabla P(\hat{w})\|^2 \le \epsilon$, the mini-batch size has to be chosen as $|S| \sim \mathcal{O}(\frac{1}{\epsilon})$, and the step-size will be $\eta \sim \mathcal{O}(\frac{\epsilon}{L})$.

There are a few variants of SARAH that do not need any restart and no full gradient estimate. E.g., the Hybrid Variance-Reduce variant [13] defines

$$v_t = \beta \nabla f_i(w_t) + (1-\beta)\left(\nabla f_i(w_t) - \nabla f_i(w_{t-1}) + v_{t-1}\right), \tag{3}$$

where $\beta \in (0, 1)$ is a hyper-parameter. A STORM variant [4] uses (3) not with a fixed value of parameter β, but in STORM, the value of β_t is diminishing to 0. The ZeroSARAH [11] is another variant, where the v_t is a combination of (3) with SAG/SAGA.

Random Sampling vs. Random Reshuffle. All the stochastic algorithms discussed so far sample function f_i randomly. However, it is a standard practice, for a finite-sum problem, not to choose functions f_i randomly with replacement, but rather make a data permutation/shuffling and then choose the f_is in a cyclic fashion. In [16] a few basic shuffling are discussed, including

- **Random Reshuffling (RR)** - reshuffle data before each epoch;
- **Shuffle-Once (SO)** - shuffle data only once before optimizing;
- **Incremental Gradient (IG)** - access data in a cycling fashion over the given dataset.

There are a few recent papers that provide a theoretical analysis of some SGD type algorithms (e.g., SGD, SVRG) in this settings, including [15, 16, 20, 28].

1.2. Contribution

The main contribution of this paper is the modification of the SARAH algorithm to remove the requirement of computing a full gradient $\nabla P(w)$, while achieving a linear convergence with a fixed step-size for strongly convex objective. The crucial algorithmic modification that was needed to achieve this goal, was to replace the random selection of functions by either of the shuffle options (**RR**, **SO**, **IG**) and designing a mechanism that can build a progressively better approximation of a full gradient $\nabla P(w_t)$ as $w_t \to w^*$

2. `Shuffled-SARAH`

2.1. Building Gradient Estimate While Optimizing

An intuition. Accessing data in a cyclic order (using any alternative described above) allows us to estimate a full gradient $\tilde{v} \approx \nabla P$. Indeed, if the step-size η_t in (2) would be zero, and v_t would be a stochastic gradient ∇f_i, then by averaging all of the stochastic gradients in one pass, we would obtain exact full gradient $\nabla P(w)$. As η increases, the stochastic gradients would be computed at different points

$$\tilde{v} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_{\pi^i}(w_i), \tag{4}$$

and hence we would not obtain the exact full gradient of $P(w)$ but rather just a rough estimate. But is it just the η that affects how good the $\tilde{v}$ will be? Of course not, as w_t is updated using (2), one can see that the radius of a set of ws that are used to compute gradient estimates is dependent on v_t. Ideally, as we will converge go w^*, then also $v_t \to \nabla P(w^*) = 0$ and hence $\tilde{v}$ will be getting closer to $\nabla P(w_t)$.

Building the gradient estimate. Our proposed approach to eliminate the need to compute the full gradient is based on a simple recursive update. Let us initialize $\tilde{v}_0 = \mathbf{0} \in \mathbb{R}^d$. Then while making a pass $i = \{1, 2, \ldots, n\}$ over the data, we will keep updating $\tilde{v}$ using the gradient estimates as follows

$$\tilde{v}_i = \frac{i-1}{i}\tilde{v}_{i-1} + \frac{1}{i}\nabla f_{\pi^i}(w_i), \quad \text{for } i \in \{1, 2, \ldots, n\}.$$

It is easy exercise to see that $\tilde{v}_i$ will be the average of gradients seen so far, and moreover, after n updates, it will be exactly as in (4). Let us note that making the pass over the dataset is a crucial to build a good estimate of the gradient and random selection of functions would not achieve this goal.

The Algorithm. We are now ready to explain the `Shuffled-SARAH` algorithm (shown in Algorithm 1) in detail. The algorithm starts by choosing an intial solution w^-, which can be done randomly and setting to e.g., $\mathbf{0}$. We will then define $v_0 = \mathbf{0}$ which will always serve as a full gradient estimate of ∇P. In line 5 we are defining $\tilde{v}$ to point to the same memory address as v_0. This basically means, that v_0 and $\tilde{v}$ will be always identical during the first pass $s = 0$, and any change to $\tilde{v}$ will be also made to v_0. Note that after lines 18,19 are executed, the v_s and $\tilde{v}$ will be two different vectors. The reason why we put *in place* in line 13 is again only to ensure that for $s = 0$ both v_0 and $\tilde{v}$ will be the same.

The random permutation in line 11 could have one of the three options mentioned in Section 1.1. For **RR**, we will permute the $[n]$ each time, for **SO** we will only shuffle one for $s = 0$ and define $\pi_s = \pi_0$ for any $s > 0$. In **IG** option we have $\pi_s = (1, 2, \ldots, n)\ \forall s$.

3. Theoretical Analysis

Before present out theoretical results we introduce some notations and assumptions.

We use $\langle x, y\rangle := \sum_{i=1}^n x_i y_i$ to define standard inner product of $x, y \in \mathbb{R}^d$. It induces ℓ_2-norm in $\mathbb{R}^d$ in the following way $\|x\| := \sqrt{\langle x, x\rangle}$.

Algorithm 1: `Shuffled-SARAH`

Input: $0 < \eta$ step-size
choose $w^- \in \mathbb{R}^d$
$w = w^-$
$v_0 = \mathbf{0} \in \mathbb{R}^d$
$\tilde{v} = \&(v_0)$ `//` $\tilde{v}$ `will point to` v_0
$\Delta = \mathbf{0} \in \mathbb{R}^d$
for $s = 0, 1, 2, \dots$ **do**
  define $w_s := w$
  $w^- = w$
  $w = w - \eta v_s$
  obtain permutation $\pi_s = (\pi_s^1, \dots, \pi_s^n)$ of $[n]$ by some rule
  for $i = 1, 2, \dots, n$ **do**
    $\tilde{v} = \frac{i-1}{i}\tilde{v} + \frac{1}{i}\nabla f_{\pi_s^i}(w)$
    $\Delta = \Delta + \nabla f_{\pi_s^i}(w) - \nabla f_{\pi_s^i}(w^-)$
    $w^- = w$
    $w = w - \eta(v_s + \Delta)$
  end
  $v_{s+1} = \tilde{v}$
  $\tilde{v} = \mathbf{0} \in \mathbb{R}^d$
  $\Delta = \mathbf{0} \in \mathbb{R}^d$
end
Return: w

Assumption 1 *For problem* (1) *the following hold:*

(i) Each $f_i : \mathcal{R}^d \to \mathcal{R}$ is convex and twice differentiable, with L-smooth gradient:

$$\|\nabla f_i(w_1) - \nabla f_i(w_2)\| \leq L\|w_1 - w_2\|,$$

for all $w_1, w_2 \in \mathcal{R}^d$;

(ii) $P(w)$ is μ-strongly convex function with minimizer x^* and optimal value P^*;

(iii) Each f_i is δ-similar with P, i.e. for all $w \in \mathcal{R}^d$ it holds that

$$\|\nabla^2 f_i(w) - \nabla^2 P(w)\| \leq \delta/2.$$

The last assumption means the similarity of $\{f_i\}$ For example, this effect is observed when the data is divided uniformly across batches f_i, then with a high probability of $\delta \sim \frac{L}{\sqrt{b}}$, where b is a size of local batch f_i (number of data points in f_i) [29].

The following theorem presents the convergence guarantees of `Shuffled-SARAH`.

Theorem 1 *Suppose that Assumption 1 hold. Consider `Shuffled-SARAH` (Algorithm 1) with the choice of η such that*

$$\eta \leq \min\left[\frac{1}{8nL}; \frac{1}{8n^2\delta}\right]. \tag{5}$$

Then, we have

$$P(w_{s+1}) - P^* + \frac{\eta(n+1)}{16}\|v_s\|^2 \leq \left(1 - \frac{\eta\mu(n+1)}{2}\right)\left(P(w_s) - P^* + \frac{\eta(n+1)}{16}\|v_{s-1}\|^2\right).$$

Hence, it is easy to obtain an estimate for the number of outer iterations in Algorithm 1.

Corollary 2 *Fix ε, and let us run* `Shuffled-SARAH` *with η from (5). Then we can obtain an ε-accuracy solution on f after*

$$S = \mathcal{O}\left(\max\left[\frac{L}{\mu}; \frac{\delta n}{\mu}\right]\log\frac{1}{\varepsilon}\right) \quad \textit{iterations}.$$

4. Numerical experiments

Trajectory We start with a toy experiment in $\mathbb{R}^2$ with a quadratic function. We compare the trajectories of the classical SARAH (two random and average), the average trajectory of the RR-SARAH (see Algorithm 2 in Appendix B), and the random trajectory `Shuffled-SARAH` with Random Reshuffling.

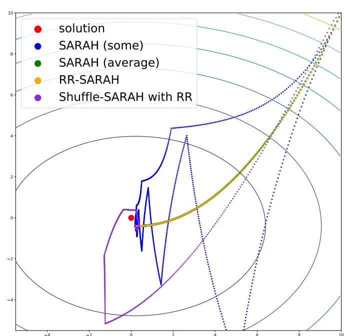


Figure 1: Trajectories on quadratic function.

Logistic regression Next, we consider the logistic regression problem with ℓ_2-regularization for binary classification with

$$f_i(w) = \frac{1}{b}\sum_{k=1}^{b}\log\left(1 + \exp\left(-y_k \cdot (X_b w)_k\right)\right) + \frac{\lambda}{2}\|w\|^2,$$

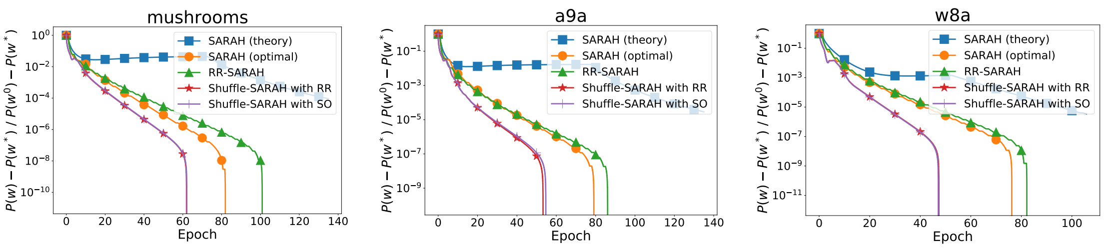
where $X_b \in \mathbb{R}^{b \times d}$ is a matrix of objects, $y_1, \ldots, y_b \in \{-1, 1\}$ are labels for these objects, b is the size of the local datasets and $w \in \mathbb{R}^d$ is a vector of weights. We optimize this problem for `mushrooms`, `a9a`, `w8a` datasets from LIBSVM library[3]. More details on the dataset parameters can be found in Table 1. We compare the following method settings: 1) SARAH with theoretical parameters $n = 4.5\kappa$, $\eta = 1/(2L)$ (see [17]), 2) SARAH with optimal parameters (is selected by brute force - see Table 2), 3) RR-SARAH with optimal step-size, 4) `Shuffled-SARAH` (Random Reshuffling) with optimal step-size, 5) `Shuffled-SARAH` (Shuffle Once) with optimal step-size. All methods are run 20 times, and the convergence results are averaged. We are interested in how these methods converge in terms of the epochs number (1 epoch is a call the full gradient P or the number of gradients f_i equivalent to the call ∇P). For results see Figures 2, 4, 5. One can note that in these cases our new methods are superior to the original SARAH.

	full size	b	d	L
mushrooms	8124	64	112	5,3
a9a	32561	256	123	3,5
w8a	49749	256	300	28,5

Table 1: Summary of datasets.

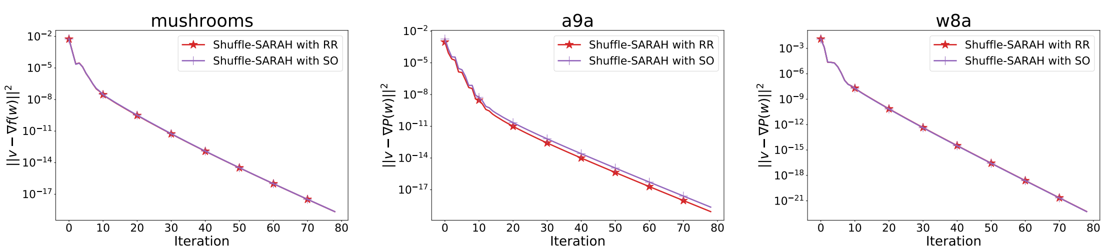
	n	η
mushrooms	$0,5 \cdot (L/\mu)$	$1/L$
a9a	$0,25 \cdot (L/\mu)$	$1/L$
w8a	L/μ	$1/L$

Table 2: Optimal parameters for SARAH.



(a) mushrooms (b) a9a dataset (c) w8a dataset

Figure 2: Convergence of SARAH-type methods on various LiBSVM datasets. Convergence on the function.



(a) mushrooms (b) a9a dataset (c) w8a dataset

Figure 3: $\|v_s - \nabla P(w_s)\|^2$ changes.

The v_s is getting closer to $\nabla P(w_s)$ The goal of this experiment is to show that v is good the approximation of ∇P and improves with each iteration. To do this, we analyze the changes of $\|v_s - \nabla P(w_s)\|^2$ on the logistic regression problem (see the previous paragraph). See the results in Figure 3. It can be seen that the difference is decreasing $\|v_s - \nabla P(w_s)\|^2$.

Acknowledgments

The research of A. Beznosikov was supported by Russian Science Foundation (project No. 21-71-30005). This work was partially conducted while A. Beznosikov, was visiting research assistants in Mohamed bin Zayed University of Artificial Intelligence (MBZUAI).

References

[1] Zeyuan Allen-Zhu and Yang Yuan. Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *International conference on machine learning*, pages 1080–1089. PMLR, 2016.

[2] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[3] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[4] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. *arXiv preprint arXiv:1905.10018*, 2019.

[5] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

[6] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path integrated differential estimator. *arXiv preprint arXiv:1807.01695*, 2018.

[7] Wenqing Hu, Chris Junchi Li, Xiangru Lian, Ji Liu, and Huizhuo Yuan. Efficient smooth non-convex stochastic compositional optimization via stochastic recursive gradient descent. 2019.

[8] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26:315–323, 2013.

[9] Ahmed Khaled and Peter Richtárik. Better theory for sgd in the nonconvex world. *arXiv preprint arXiv:2002.03329*, 2020.

[10] Bingcong Li, Meng Ma, and Georgios B Giannakis. On the convergence of sarah and beyond. In *International Conference on Artificial Intelligence and Statistics*, pages 223–233. PMLR, 2020.

[11] Zhize Li and Peter Richtárik. Zerosarah: Efficient nonconvex finite-sum optimization with zero full gradient computation. *arXiv preprint arXiv:2103.01447*, 2021.

[12] Zhize Li, Hongyan Bao, Xiangliang Zhang, and Peter Richtárik. Page: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In *International Conference on Machine Learning*, pages 6286–6295. PMLR, 2021.

[13] Deyi Liu, Lam M Nguyen, and Quoc Tran-Dinh. An optimal hybrid variance-reduced algorithm for stochastic composite nonconvex optimization. *arXiv preprint arXiv:2008.09055*, 2020.

[14] Julien Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

[15] Grigory Malinovsky, Alibek Sailanbayev, and Peter Richtárik. Random reshuffling with variance reduction: New analysis and better rates. *arXiv preprint arXiv:2104.09342*, 2021.

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

[17] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: a novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning*, pages 2613–2621. PMLR, 2017.

[18] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint arXiv:1705.07261*, 2017.

[19] Lam M Nguyen, Phuong Ha Nguyen, Peter Richtárik, Katya Scheinberg, Martin Takác, and Marten van Dijk. New convergence aspects of stochastic gradient algorithms. *J. Mach. Learn. Res.*, 20:176–1, 2019.

[20] Lam M Nguyen, Quoc Tran-Dinh, Dzung T Phan, Phuong Ha Nguyen, and Marten van Dijk. A unified convergence analysis for shuffling-type gradient methods. *arXiv preprint arXiv:2002.08246*, 2020.

[21] Lam M Nguyen, Katya Scheinberg, and Martin Takáč. Inexact SARAH algorithm for stochastic optimization. *Optimization Methods and Software*, 36(1):237–258, 2021.

[22] Xun Qian, Zheng Qu, and Peter Richtárik. Saga with arbitrary sampling. In *International Conference on Machine Learning*, pages 5190–5199. PMLR, 2019.

[23] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[24] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

[25] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[26] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

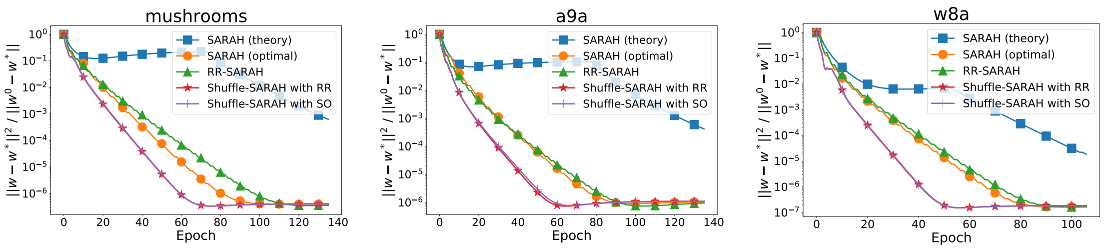
[27] Martin Takác, Avleen Bijral, Peter Richtárik, and Nati Srebro. Mini-batch primal and dual methods for svms. In *International Conference on Machine Learning*, pages 1022–1030. PMLR, 2013.

[28] Trang H Tran, Lam M Nguyen, and Quoc Tran-Dinh. SMG: a shuffling gradient-based method with momentum. In *International Conference on Machine Learning*, pages 10379–10389. PMLR, 2021.

[29] Joel A Tropp. An introduction to matrix concentration inequalities. *arXiv preprint arXiv:1501.01571*, 2015.

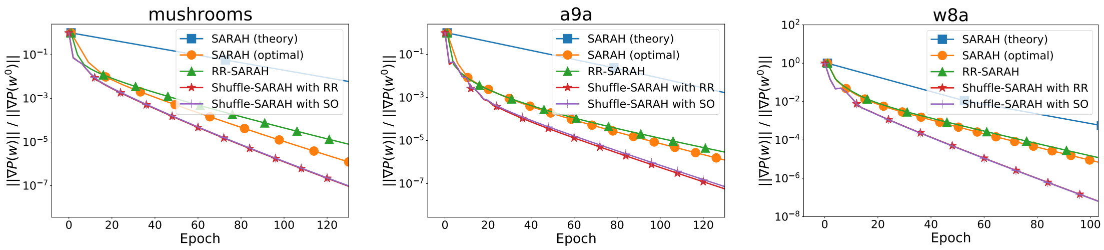
[30] Zhuang Yang, Zengping Chen, and Cheng Wang. Accelerating mini-batch SARAH by step size rules. *Information Sciences*, 558:157–173, 2021.

Appendix A. Additional experimental results



(a) mushrooms (b) a9a dataset (c) w8a dataset

Figure 4: Convergence of SARAH-type methods on various LiBSVM datasets. Convergence on the distance to the solution.



(a) mushrooms (b) a9a dataset (c) w8a dataset

Figure 5: Convergence of SARAH-type methods on various LIBSVM datasets. Convergence on the norm og the gradient.

Appendix B. RR-SARAH

This Algorithm is a modification of the original SARAH using Random Reshuffling. Unlike Algorithm 1, this algorithm uses the full gradient ∇P.

Theorem 3 *Suppose that Assumption 1 hold. Consider RR-SARAH (Algorithm 2) with the choice of η such that*

$$\eta \leq \min\left[\frac{1}{8nL}; \frac{1}{8n^2\delta}\right]. \tag{6}$$

Then, we have

$$P(w_{s+1}) - P^* \leq \left(1 - \frac{\eta\mu(n+1)}{2}\right)(P(w_s) - P^*).$$

Algorithm 2: RR-SARAH

Input: $0 < \eta$ step-size
choose $w^- \in \mathbb{R}^d$
$w = w^-$
for $s = 0, 1, 2, \dots$ **do**
  define $w_s := w$
  $v = \nabla P(w)$
  $w^- = w$
  $w = w - \eta v$
  sample a permutation $\pi_s = (\pi_s^1, \dots, \pi_s^n)$ of $[n]$
  for $i = 1, 2, \dots, n$ **do**
    $v = v + \nabla f_{\pi_s^i}(w) - \nabla f_{\pi_s^i}(w^-)$
    $w^- = w$
    $w = w - \eta v$
  end
end
Return: w

Corollary 4 *Fix ε, and let us run RR-SARAH with η from (6). Then we can derive an ε-accuracy solution on f after*

$$S = \mathcal{O}\left(\max\left[\frac{L}{\mu}; \frac{\delta n}{\mu}\right] \log \frac{1}{\varepsilon}\right) \quad \textit{iterations.}$$

Appendix C. Missing proofs for Section 3 and Appendix B

Before we start to prove, let us note that δ-similarity from Assumption 1 gives $\delta/2$-smoothness of function $f_i - P$ for any i. Then this implies δ-smoothness of function $f_i - f_j$ for any i, j

$$\begin{aligned} &\|\nabla f_i(w_1) - \nabla f_j(w_1) - (\nabla f_i(w_2) - \nabla f_j(w_2))\| \\ &\qquad \le \|\nabla f_i(w_1) - \nabla P(w_1) - (\nabla f_i(w_2) - \nabla P(w_2))\| \\ &\qquad\quad + \|\nabla P(w_1) - \nabla f_j(w_1) - (\nabla P(w_2) - \nabla f_j(w_2))\| \\ &\qquad \le 2 \cdot (\delta/2)\|w_1 - w_2\|^2 = \delta\|w_1 - w_2\|^2 \end{aligned} \tag{7}$$

Next we introduce additional notation for simplicity. If we consider Algorithm 1 in iteration $s \neq 0$, one can note that update rule is nothing more than

$$\begin{aligned} w_s &= w_s^0 = w_{s-1}^{n+1}, \\ v_s &= v_s^0 = \frac{1}{n}\sum_{i=1}^{n} f_{\pi_{s-1}^i}(w_{s-1}^i), \\ w_s^1 &= w_s^0 - \eta v_s^0, \\ v_s^i &= v_s^{i-1} + f_{\pi_s^i}(w_s^i) - f_{\pi_s^i}(w_s^{i-1}), \\ w_s^{i+1} &= w_s^i - \eta v_s^i. \end{aligned}$$

These new notations will be used further in the proofs. For Algorithm 2, one can do exactly the same notations with $v_s = v_s^0 = \nabla P(w_s)$.

Lemma 5 *Under Assumption 1, for Algorithms 1 and 2 with η from (5) the following holds*

$$P(w_{s+1}) \leq P(w_s) - \frac{\eta n}{2}\|\nabla P(w_s)\|^2 + \frac{\eta n}{2}\left\|\nabla P(w_s) - \frac{1}{n}\sum_{i=1}^n v_s^i\right\|^2.$$

Proof: Using L-smoothness of function P, we have

$$\begin{aligned}
P(w_{s+1}) &\leq P(w_s) + \langle \nabla P(w_s), w_{s+1} - w_s\rangle + \frac{L}{2}\|w_{s+1} - w_s\|^2 \\
&= P(w_s) - \eta(n+1)\left\langle \nabla P(w_s), \frac{1}{n+1}\sum_{i=0}^n v_s^i \right\rangle + \frac{\eta^2(n+1)^2 L}{2}\left\|\frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2 \\
&= P(w_s) - \frac{\eta(n+1)}{2}\left(\|\nabla P(w_s)\|^2 + \left\|\frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2 - \left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2\right) \\
&\quad + \frac{\eta^2(n+1)^2 L}{2}\left\|\frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2 \\
&= P(w_s) - \frac{\eta(n+1)}{2}\|\nabla P(w_s)\|^2 - \frac{\eta(n+1)}{2}(1 - \eta(n+1)L)\left\|\frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2 \\
&\quad + \frac{\eta(n+1)}{2}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2.
\end{aligned}$$

With $\eta \leq \frac{1}{8nL} \leq \frac{1}{(n+1)L}$ we get

$$P(w_{s+1}) \leq P(w_s) - \frac{\eta(n+1)}{2}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2.$$

Which completes the proof.

□

Lemma 6 *Under Assumption 1, for Algorithms 1 and 2 the following holds*

$$\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^n v_s^i\right\|^2 \leq 2\|\nabla P(w_s) - v_s\|^2 + \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=0}^n \|w_s^i - w_s\|^2.$$

Proof: Using the rule for v_s^i, we get

$$\begin{aligned}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 &= \frac{1}{(n+1)^2}\left\|(n+1)\nabla P(w_s) - (v_s^n + \ldots + v_t^0)\right\|^2 \\ &= \frac{1}{(n+1)^2}\big\|(n+1)\nabla P(w_s) \\ &\quad - \big[\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s^{n-1}) + 2v_s^{n-1} + v_s^{n-2} \ldots + v_s^0\big]\big\|^2 \\ &= \frac{1}{(n+1)^2}\big\|(n+1)\nabla P(w_s) - \big[\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s^{n-1}) \\ &\quad + 2\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - 2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) \\ &\quad + 3v_s^{n-2} + v_s^{n-3} \ldots + v_s^0\big]\big\|^2.\end{aligned}$$

Continuing further

$$\begin{aligned}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 &= \frac{1}{(n+1)^2}\big\|(n+1)\nabla P(w_s) - \big[\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s^{n-1}) \\ &\quad + 2\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - 2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) \\ &\quad + 3\nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\ &\quad \ldots \\ &\quad + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0) + (n+1)v_s^0\big]\big\|^2 \\ &\le \frac{2}{(n+1)^2}\left\|(n+1)\nabla P(w_s) - (n+1)v_s\right\|^2 \\ &\quad + \frac{2}{(n+1)^2}\big\|\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s^{n-1}) \\ &\quad + 2\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - 2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) \\ &\quad + 3\nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\ &\quad \ldots \\ &\quad + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.\end{aligned}$$

In last we use $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$. Then

$$\begin{aligned}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 &\le \frac{2}{(n+1)^2}\left\|(n+1)\nabla P(w_s) - (n+1)v_s\right\|^2 \\ &\quad + \frac{2}{(n+1)^2}\big\|\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s) \\ &\quad + \nabla f_{\pi_s^n}(w_s) - \nabla f_{\pi_s^{n-1}}(w_s) - (\nabla f_{\pi_s^n}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s^{n-1})) \\ &\quad + \nabla f_{\pi_s^{n-1}}(w_s) + \nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - 2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) \\ &\quad + 3\nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\ &\quad \ldots \\ &\quad + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.\end{aligned}$$

Using $\|a+b\|^2 \leq (1+c)\|a\|^2 + (1+1/c)\|b\|^2$ with $c = n$, we have

$$
\begin{aligned}
\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 \leq\; & 2\|\nabla P(w_s) - v_s\|^2 \\
& + \frac{2}{n+1}\big\|\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s) \\
& + \nabla f_{\pi_s^n}(w_s) - \nabla f_{\pi_s^{n-1}}(w_s) - (\nabla f_{\pi_s^n}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s^{n-1}))\big\|^2 \\
& + \frac{2}{(n+1)^2}\left(1+\frac{1}{n}\right)\big\|\nabla f_{\pi_s^{n-1}}(w_s) + \nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - 2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) \\
& + 3\nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2 \\
\leq\; & 2\|\nabla f(x_t) - v_t^0\|^2 \\
& + \frac{4}{n+1}\big\|\nabla f_{\pi_s^n}(w_s^n) - \nabla f_{\pi_s^n}(w_s)\big\|^2 \\
& + \frac{4}{n+1}\big\|\nabla f_{\pi_s^n}(w_s) - \nabla f_{\pi_s^{n-1}}(w_s) - (\nabla f_{\pi_s^n}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s^{n-1}))\big\|^2 \\
& + \frac{2}{n(n+1)}\big\|\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s) \\
& + 2\nabla f_{\pi_s^{n-1}}(w_s) - 2\nabla f_{\pi_s^{n-2}}(w_s) - (2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) - 2\nabla f_{\pi_s^{n-2}}(w_s^{n-2})) \\
& + 2\nabla f_{\pi_s^{n-2}}(w_s) + \nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.
\end{aligned}
$$

Using δ-similarity (7) and L-smoothness (Assumption 1)

$$
\begin{aligned}
\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 \leq\; & 2\|\nabla P(w_s) - v_s\|^2 2 \\
& + \frac{4L^2}{n+1}\big\|w_s^n - w_s\big\|^2 + \frac{4\delta^2}{n+1}\big\|w_s - w_s^{n-1}\big\|^2 \\
& + \frac{2}{n(n+1)}\big\|\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s) \\
& + 2\nabla f_{\pi_s^{n-1}}(w_s) - 2\nabla f_{\pi_s^{n-2}}(w_s) - (2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) - 2\nabla f_{\pi_s^{n-2}}(w_s^{n-2})) \\
& + 2\nabla f_{\pi_s^{n-2}}(w_s) + \nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.
\end{aligned}
$$

Using $\|a+b\|^2 \leq (1+c)\|a\|^2 + (1+1/c)\|b\|^2$ with $c = n-1$

$$
\begin{aligned}
\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 \leq\ & 2\|\nabla P(w_s) - v_s\|^2 \\
& + \frac{4L^2}{n+1}\left\|w_s^n - w_s\right\|^2 + \frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-1}\right\|^2 \\
& + \frac{2}{n+1}\left\|\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s)\right. \\
& + 2\nabla f_{\pi_s^{n-1}}(w_s) - 2\nabla f_{\pi_s^{n-2}}(w_s) - (2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) - 2\nabla f_{\pi_s^{n-2}}(w_s^{n-2}))\big\|^2 \\
& + \frac{2}{(n+1)(n-1)}\big\|2\nabla f_{\pi_s^{n-2}}(w_s) + \nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2 \\
\leq\ & 2\|\nabla P(w_s) - v_s\|^2 \\
& + \frac{4L^2}{n+1}\left\|w_s^n - w_s\right\|^2 + \frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-1}\right\|^2 \\
& + \frac{4}{n+1}\left\|\nabla f_{\pi_s^{n-1}}(w_s^{n-1}) - \nabla f_{\pi_s^{n-1}}(w_s)\right\|^2 \\
& + \frac{4}{n+1}\left\|\nabla f_{\pi_s^{n-1}}(w_s) - 2\nabla f_{\pi_s^{n-2}}(w_s) - (2\nabla f_{\pi_s^{n-1}}(w_s^{n-2}) - 2\nabla f_{\pi_s^{n-2}}(w_s^{n-2}))\right\|^2 \\
& + \frac{2}{(n+1)(n-1)}\big\|2\nabla f_{\pi_s^{n-2}}(w_s) + \nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.
\end{aligned}
$$

Again with δ-similarity and L-smoothness

$$
\begin{aligned}
\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 \leq\ & 2\|\nabla P(w_s) - v_s\|^2 \\
& + \frac{4L^2}{n+1}\left\|w_s^n - w_s\right\|^2 + \frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-1}\right\|^2 \\
& + \frac{4L^2}{n+1}\left\|w_s^{n-1} - w_s\right\|^2 + 2^2\cdot\frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-2}\right\|^2 \\
& + \frac{2}{(n+1)(n-1)}\big\|2\nabla f_{\pi_s^{n-2}}(w_s) + \nabla f_{\pi_s^{n-2}}(w_s^{n-2}) - 3\nabla f_{\pi_t^{n-2}}(w_s^{n-3}) \\
& \dots \\
& + n\nabla f_{\pi_s^1}(w_s^1) - n\nabla f_{\pi_s^1}(w_s^0)\big\|^2.
\end{aligned}
$$

Continuing further we have

$$\begin{aligned}\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 &\leq 2\|\nabla P(w_s) - v_s\|^2 \\ &\quad + \frac{4L^2}{n+1}\left\|w_s^n - w_s\right\|^2 + 1^2 \cdot \frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-1}\right\|^2 \\ &\quad + \frac{4L^2}{n+1}\left\|w_s^{n-1} - w_s\right\|^2 + 2^2 \cdot \frac{4\delta^2}{n+1}\left\|w_s - w_s^{n-2}\right\|^2 \\ &\quad \dots \\ &\quad + \frac{4L^2}{n+1}\left\|w_s^1 - w_s\right\|^2 + n^2 \cdot \frac{4\delta^2}{n+1}\left\|w_s - w_s^0\right\|^2 \\ &\leq 2\|\nabla P(w_s) - v_s\|^2 + \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=1}^{n}\|w_s^i - w_s\|^2.\end{aligned}$$

Which completes the proof.

□

Proof of Theorem 3. For RR-SARAH $v_s = \nabla P(w_s)$, then by Lemma 6 we get

$$\left\|\nabla P(w_s) - \frac{1}{n+1}\sum_{i=0}^{n} v_s^i\right\|^2 \leq \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=1}^{n}\|w_s^i - w_s\|^2.$$

And with Lemma 5

$$P(w_{s+1}) \leq P(w_s) - \frac{\eta(n+1)}{2}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=1}^{n}\|w_s^i - w_s\|^2.$$

Then we will work with $\sum\limits_{i=1}^{n}\|w_s^i - w_s\|^2$. By Lemma 3 from [17] (see the proof) we get that $\|v_s^i\|^2 \leq \|v_s^{i-1}\|^2$. Then

$$\begin{aligned}\sum_{i=1}^{n}\|w_s^i - w_s\|^2 = \eta^2\sum_{i=1}^{n}\left\|\sum_{k=0}^{i-1} v_s^k\right\|^2 &\leq \eta^2\sum_{i=1}^{n} i\sum_{k=0}^{i-1}\left\|v_s^k\right\|^2 \leq \eta^2\sum_{i=1}^{n} i\sum_{k=0}^{i-1}\|v_s\|^2 \\ &\leq \eta^2\left\|v_s\right\|^2\sum_{i=1}^{n} i\sum_{k=0}^{i-1} 1 \\ &\leq \eta^2 n^3\left\|v_s\right\|^2 = \eta^2 n^3\left\|\nabla P(w_s)\right\|^2.\end{aligned}$$

Hence

$$\begin{aligned}P(w_{s+1}) &\leq P(w_s) - \frac{\eta(n+1)}{2}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\left\|\nabla P(w_s)\right\|^2 \\ &\leq P(w_s) - \frac{\eta(n+1)}{2}\left(1 - \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\right)\|\nabla P(w_s)\|^2.\end{aligned}$$

With $\gamma \le \frac{1}{8nL}; \frac{1}{8n^2\delta}$ we get

$$P(w_{s+1}) - P^* \le P(w_s) - P^* - \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2.$$

Strong-convexity of P end the proof:

$$P(w_{s+1}) - P^* \le \left(1 - \frac{\eta(n+1)\mu}{2}\right)\left(P(w_s) - P^*\right).$$

□

Proof of Theorem 1. For RR-SARAH $v_s = \frac{1}{n}\sum_{i=1}^{n} f_{\pi_{s-1}^i}(w_{s-1}^i)$, then

$$\begin{aligned}\left\|\nabla P(w_s) - \frac{1}{n}\sum_{i=1}^{n} v_s^i\right\|^2 &\le \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=1}^{n}\|w_s^i - w_s\|^2 + 2\left\|\frac{1}{n}\sum_{i=1}^{n} f_{\pi_{s-1}^i}(w_s) - f_{\pi_{s-1}^i}(w_{s-1}^i)\right\|^2 \\ &\le \left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\sum_{i=1}^{n}\|w_s^i - w_s\|^2 + \frac{2L^2}{n}\sum_{i=1}^{n}\left\|w_{s-1}^i - w_s\right\|^2.\end{aligned}$$

With $\sum_{i=1}^{n}\|w_t^i - w_t\|^2$ we work in the same way as in proof of Theorem 3. And with $\sum_{i=1}^{n}\left\|w_{s-1}^i - w_s\right\|^2$

$$\begin{aligned}\sum_{i=1}^{n}\|w_{s-1}^i - w_s\|^2 = \eta^2\sum_{i=1}^{n}\left\|\sum_{k=1}^{n+1-i} v_{s-1}^{n+1-k}\right\|^2 &\le \eta^2\sum_{i=1}^{n}(n+1-i)\sum_{k=1}^{n+1-i}\left\|v_{s-1}^{n+1-k}\right\|^2 \\ &\le \eta^2\sum_{i=1}^{n}(n+1-i)\sum_{k=1}^{n+1-i}\|v_{s-1}\|^2 \\ &\le \eta^2\|v_{s-1}\|^2\sum_{i=1}^{n}(n+1-i)\sum_{k=1}^{n+1-i} 1 \\ &\le \eta^2 n^3\|v_{s-1}\|^2. \end{aligned} \tag{8}$$

With Lemma 5

$$\begin{aligned}P(w_{s+1}) \le P(w_s) &- \frac{\eta(n+1)}{2}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left[\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\|v_s\|^2 + \frac{2L^2}{n}\cdot\eta^2 n^3\|v_{s-1}\|^2\right] \\ = P(w_s) &- \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left[\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\|v_s\|^2 + \frac{2L^2}{n}\cdot\eta^2 n^3\|v_{s-1}\|^2\right] \\ &- \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 \\ \le P(w_s) &- \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left[\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\|v_s\|^2 + \frac{2L^2}{n}\cdot\eta^2 n^3\|v_{s-1}\|^2\right] \\ &- \frac{\eta(n+1)}{8}\|v_s\|^2 + \frac{\eta(n+1)}{4}\|v_s - \nabla P(w_s)\|^2 \\ \le P(w_s) &- \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{2}\left[\left(\frac{4L^2}{n+1} + 4\delta^2 n\right)\cdot\eta^2 n^3\|v_s\|^2 + \frac{2L^2}{n}\cdot\eta^2 n^3\|v_{s-1}\|^2\right] \\ &- \frac{\eta(n+1)}{8}\|v_s\|^2 + \frac{\eta(n+1)}{4}\cdot\frac{2L^2}{n}\cdot\eta^2 n^3\|v_{s-1}\|^2.\end{aligned}$$

The last is deduced the same way as (8). Small rearrangement gives

$$\begin{aligned}P(w_{s+1}) - P^* &\le P(w_s) - P^* - \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 \\ &\quad - \frac{\eta(n+1)}{8}\left(1 - \left(\frac{16L^2}{n+1} + 16\delta^2 n\right)\cdot \eta^2 n^3\right)\|v_s\|^2 + \eta(n+1)\cdot \frac{2L^2}{n}\cdot \eta^2 n^3 \|v_{s-1}\|^2 .\end{aligned}$$

$\eta \le \min\{\frac{1}{8nL}; \frac{1}{8n^2\delta}\}$ gives

$$P(w_{s+1}) - P^* + \frac{\eta(n+1)}{16}\|v_s\|^2 \le P(w_s) - P^* - \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 + \frac{\eta(n+1)}{16}\cdot\frac{32L^2}{n}\cdot \eta^2 n^3 \|v_{s-1}\|^2 .$$

With $\eta \le \frac{1}{8Ln}$, we get $32L^2\eta^2 n^2 \le \left(1 - \frac{\eta(n+1)\mu}{2}\right)$ and

$$P(w_{s+1}) - P^* + \frac{\eta(n+1)}{16}\|v_s\|^2 \le P(w_s) - P^* - \frac{\eta(n+1)}{4}\|\nabla P(w_s)\|^2 + \left(1 - \frac{\eta(n+1)\mu}{2}\right)\cdot\frac{\eta(n+1)}{16}\|v_{s-1}\|^2 .$$

Strong-convexity of P ends the proof.

$\square$