Variance-Reduction Methods: SGD(+SWA) vs Nesterov vs SVRG

Author: Shtykov Pavel

Problem: SGD does not converge to the minimum, but instead oscillates around it.

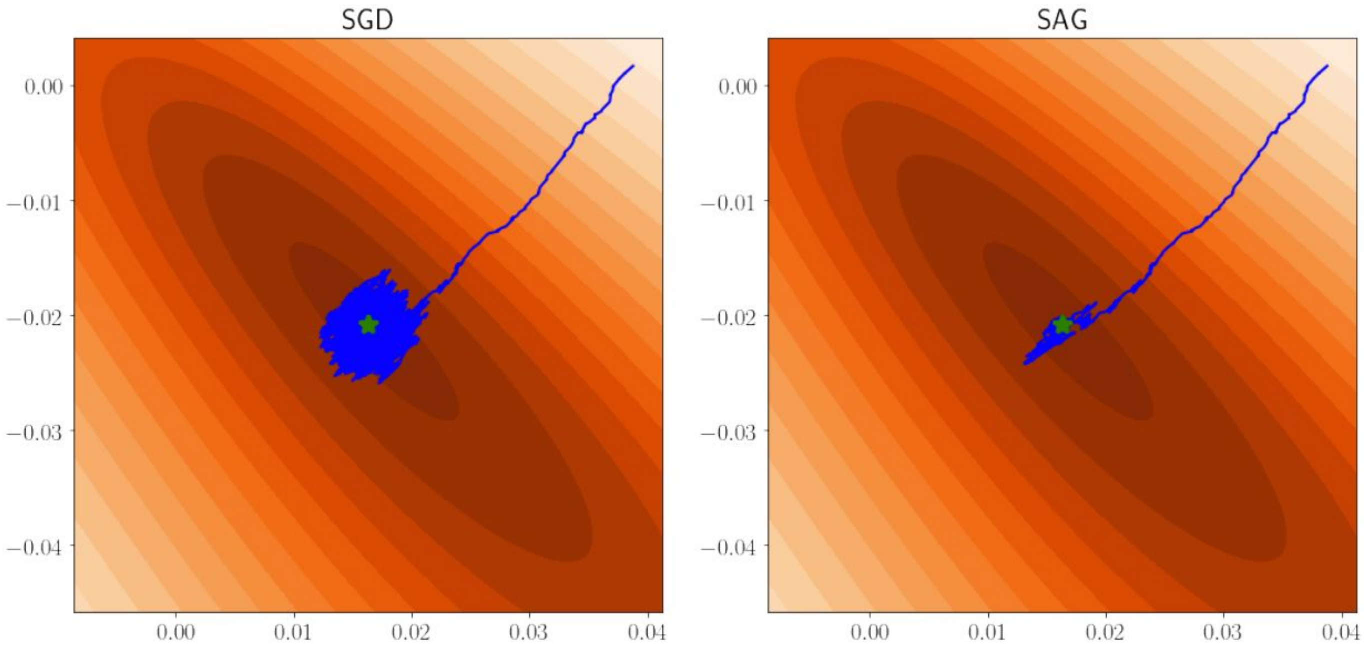


Fig. 2. Level set plot of 2D logistic regression with the iterates of SGD (left) and SAG (right) with constant stepsize. The green star is the x_* solution.

Typical Solutions to This Problem and their disadvantages (according to authors*)

- *Scheduling LR* – but it is difficult to tune
- *Momentum* – but it does not converge to the *full gradient* $\nabla f(x_k)$ whatever
- Mini-batching – the cost of this iteration increases proportionally to the mini-batch size.

*Gower, Robert Mansel et al. "Variance-Reduced Methods for Machine Learning." *Proceedings of the IEEE* 108 (2020): 1968-1983.

Authors' Solution: **Variance Reduction Methods**

Let's use estimate $g_k \in \mathbb{R}^d$ gradient such that $g_k \approx \nabla f(x_k)$.

Then iteration step looks like: $x_{k+1} = x_k - \gamma g_k,$

To make such algorithm converge with a *constant step size*, we need to ensure that the variance of our gradient estimate g_k converges to zero (VR-property):

$$\mathbf{E}\left[\|g_k - \nabla f(x_k)\|^2\right] \underset{k \to \infty}{\longrightarrow} 0,$$

Ideal (unreal) **VR-method**: $\mathtt{SGD}_\star$

Algorithm: $x_{k+1} = x_k - \gamma \left(\nabla f_{i_k}(x_k) - \nabla f_{i_k}(x_\star) \right),$

This algorithm is unreal because we don't know $\nabla f_i(x_\star)$, but we can think that real VR-methods is "approximation" of $\mathtt{SGD}_\star$.

Of course it satisfies main VR-property:

$$\begin{aligned} \mathbf{E}\left[\|g_k - \nabla f(x_k)\|^2\right] &= \mathbf{E}\left[\|\nabla f_i(x_k) - \nabla f_i(x_\star) - \nabla f(x_k)\|^2\right] \\ &\le \mathbf{E}\left[\|\nabla f_i(x_k) - \nabla f_i(x_\star)\|^2\right], \end{aligned}$$

SVRG: Stochastic Variance-Reduced Gradient method

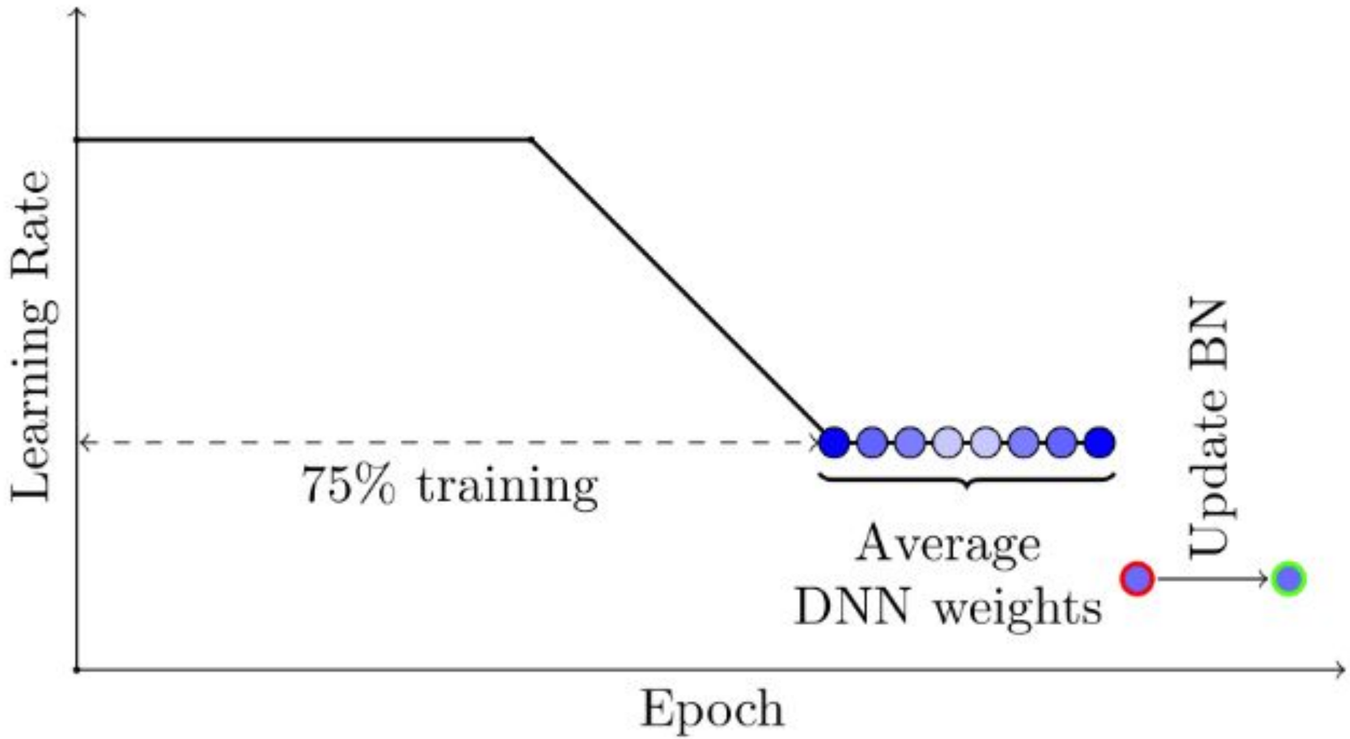
Parameters stepsize $\gamma > 0$
Initialization $\bar{x}_0 = x_0 \in \mathbb{R}^d$
for $s = 1, 2, \ldots$ **do**
    Compute and store $\nabla f(\bar{x}_{s-1})$
    $x_0 = \bar{x}_{s-1}$
    Choose the number of inner-loop iterations t
    for $k = 0, 1, \ldots, t-1$ **do**
        Sample $i_k \in \{1, \ldots, n\}$
        $g_k = \nabla f_{i_k}(x_k) - \nabla f_{i_k}(\bar{x}_{s-1}) + \nabla f(\bar{x}_{s-1})$
        $x_{k+1} = x_k - \gamma g_k$
    $\bar{x}_s = x_t$.

Properties of SVRG:

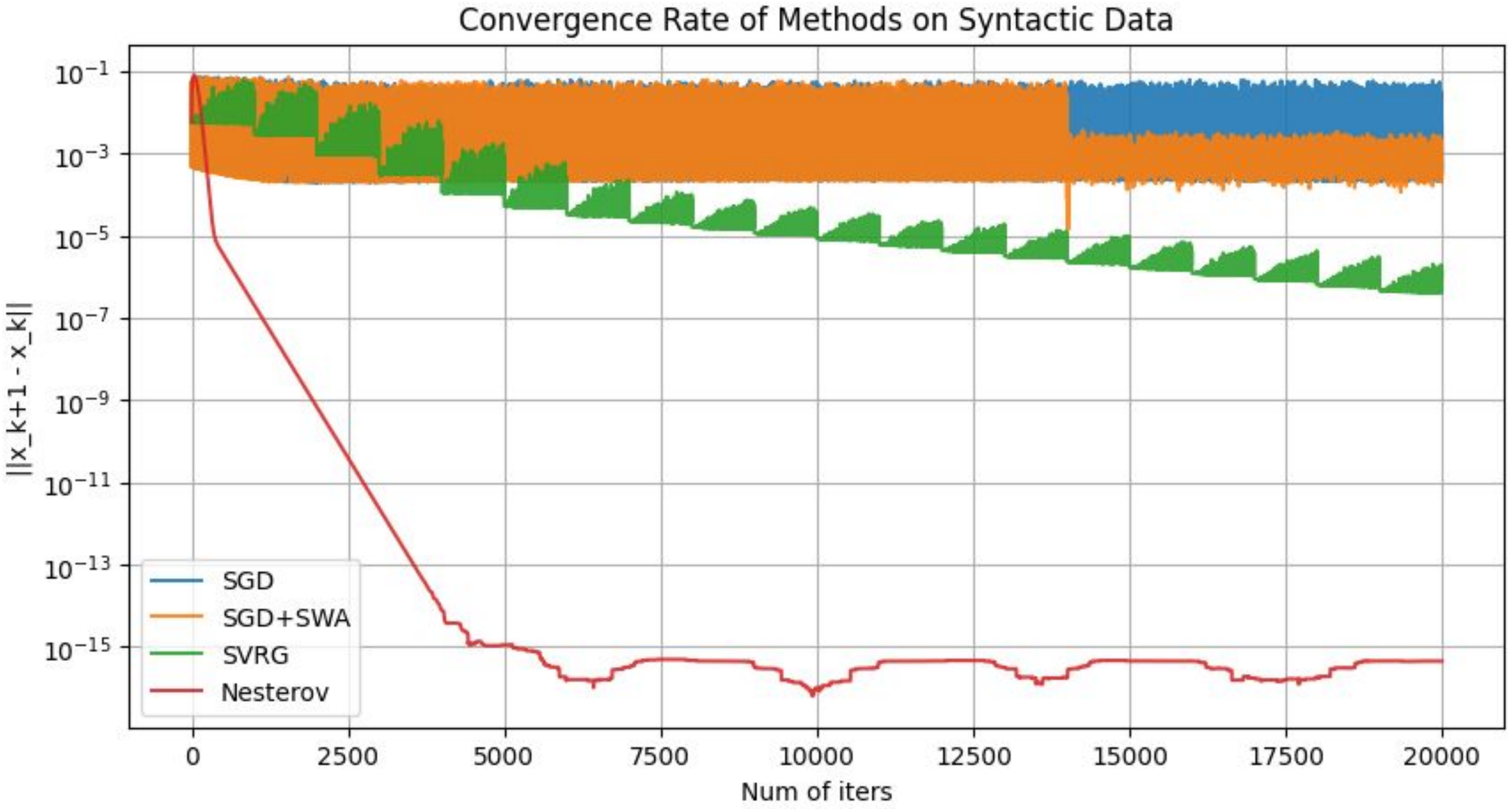
- Requires only $\mathcal{O}(d)$ memory, less that other VR methods
- Has iteration complexity $\mathcal{O}((\kappa_{\max} + n)\log(1/\varepsilon))$, similar to other VR methods
- Gradient estimate g_k is bounded:

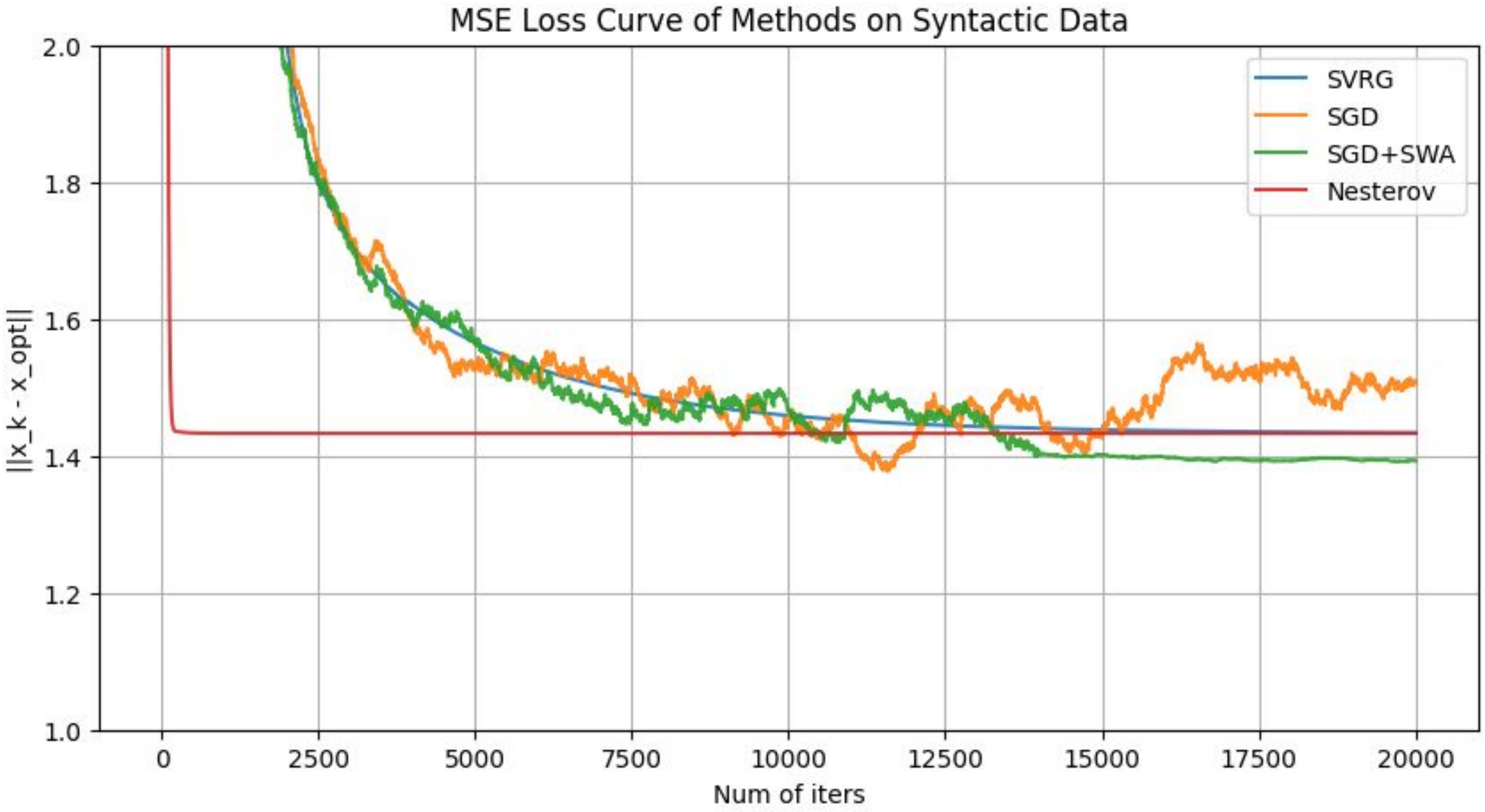
$$\begin{aligned}\mathbf{E}\left[\|g_k - \nabla f(x_k)\|^2\right] &\leq \mathbf{E}\left[\|\nabla f_i(x_k) - \nabla f_i(\bar{x})\|^2\right] \\ &\leq L_{\max}^2 \|x_k - \bar{x}\|^2,\end{aligned}$$

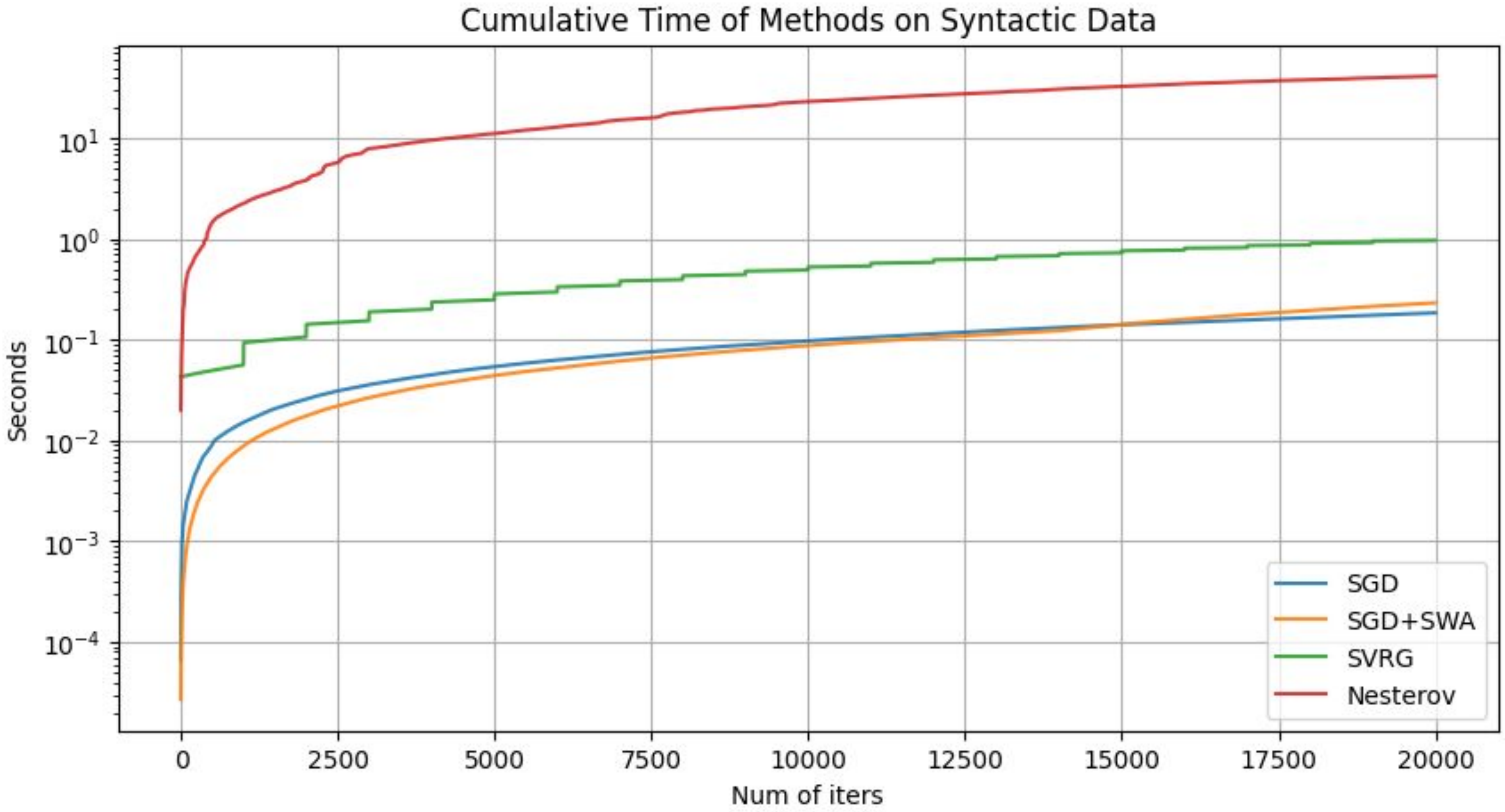
For a more interesting baseline, I try to used **SWA** for **SGD**



*SWA - [Stochastic Weight Averaging]


Convergence Rate of Methods on Syntactic Data
10^{-1}
10^{-3}
10^{-5}
10^{-7}
10^{-9}
10^{-11}
10^{-13}
10^{-15}
||x_k+1 - x_k||
SGD
SGD+SWA
SVRG
Nesterov
0
2500
5000
7500
10000
12500
15000
17500
20000
Num of iters


MSE Loss Curve of Methods on Syntactic Data
||x_k - x_opt||
Num of iters
SVRG
SGD
SGD+SWA
Nesterov
2.0
1.8
1.6
1.4
1.2
1.0
0
2500
5000
7500
10000
12500
15000
17500
20000


Cumulative Time of Methods on Syntactic Data
Seconds
Num of iters
0
2500
5000
7500
10000
12500
15000
17500
20000
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
SGD
SGD+SWA
SVRG
Nesterov

Real Data: **Student Depression Dataset**

- **Binary classification**, 27k samples, 18 features (categorical & numerical)
- **Basic preprocessing**: drop NaNs, One-Hot encoded, standard scaled
- Set **same LR** and **number of iterations** for each method

ROC-AUC Score on test set for methods:

SGD	SGD + SWA	Nesterov	SVRG
0.731	0.900	0.920	0.917

Conclusion

- **SVRG** has clear idea and fast iterations, and it produces good results. However, **Nesterov Momentum** has slightly better results quality and faster convergence, although its iterations are much slower.
- **SWA** can significantly improve SGD performance on real data.