Painless Stochastic Gradient
Interpolation, Line-Search, and Convergence Rates

NeurIPS 2019






Sharan Vaswani (**Mila**)

Aaron Mishkin (**UBC**)

Issam Laradji (**UBC, EAI**)

Mark Schmidt (**UBC**)

Gauthier Gidel **(Mila, EAI)**

Simon Lacoste-Julien (**Mila**)

SGD is awesome, but...

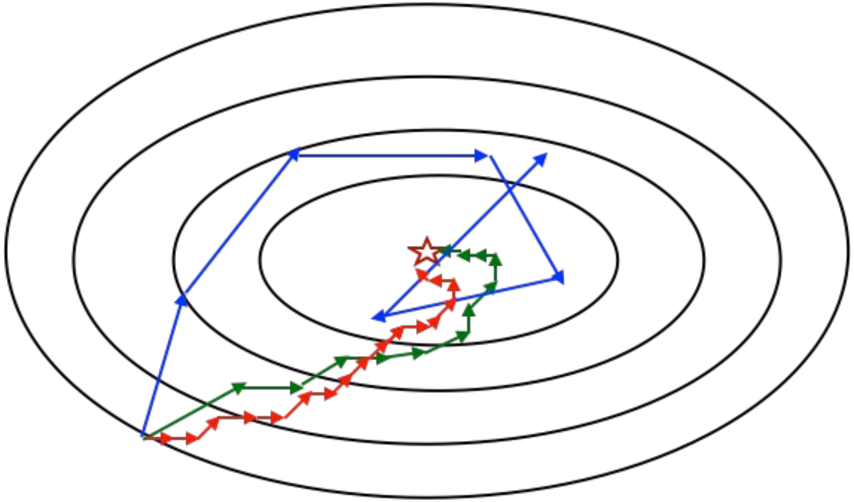
Objective $$\min f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$

Stochastic Gradient Descent (SGD) $$w_{k+1} = w_k - \eta_k \nabla f_{i_k}(w_k)$$

- \+ Simple to implement and use.
- \+ Good generalization properties.

- \- Slow convergence.
- \- Need to carefully tune the step-size.

Painful ! :(



Making SGD painless

Ingredient 1: Interpolation

- Satisfied by large *over-parametrized* models including typical neural networks and expressive kernel mappings.

$$||\nabla f(w^*)|| = 0 \implies \forall i, ||\nabla f_i(w^*)|| = 0$$

- Results in fast convergence of constant step-size SGD.

Ingredient 2: Stochastic Line Search (SLS)

- Don't manually tune the step-size, automatically *search* for it.

Strong theoretical results and good empirical performance!

SLS is simply SGD + Line search

In iteration k,

1. Compute the gradients $\nabla f_{ik}(w_k)$ for a given training batch

2. Search for a step-size η_k that satisfies the following *stochastic Armijo condition,*

$$f_{ik}\left(w_k - \eta_k \nabla f_{ik}(w_k)\right) \leq f_{ik}(w_k) - c\, \eta_k \, ||\nabla f_{ik}(w_k)||^2$$

3. Use the step-size and update the model parameters with SGD,

$$w_{k+1} = w_k - \eta_k \nabla f_{ik}(w_k)$$

SLS is theoretically sound

Interpolation enables SLS

- Match *full-batch* convergence rates for smooth convex functions and strongly-convex functions.
- Achieve fast convergence rates for non-convex functions*.

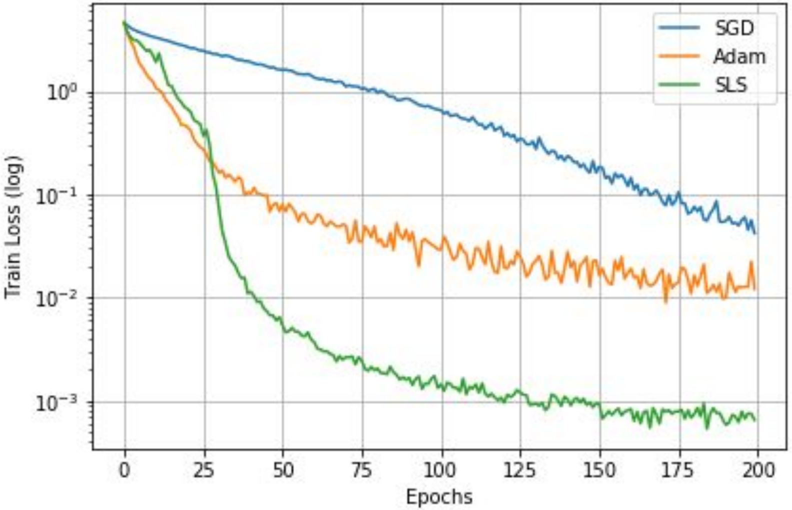
Other results:

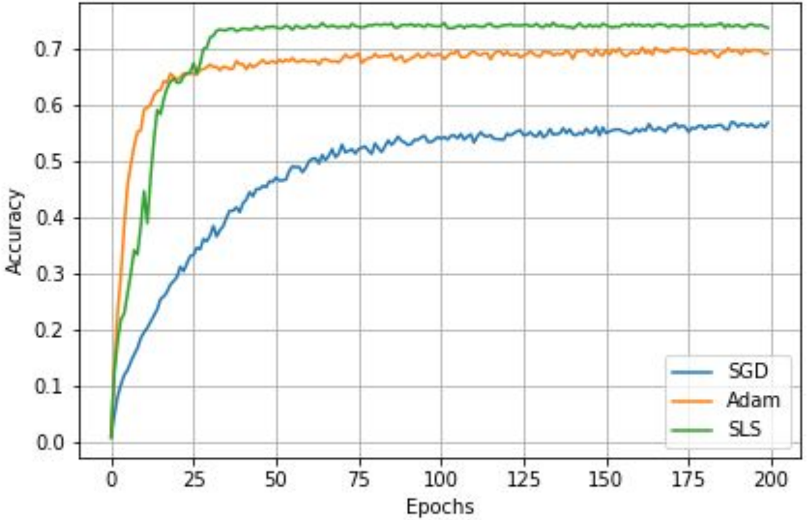
- *Lipschitz line search* for Stochastic Extra-gradient (SEG) and its fast convergence for a subset of non-convex functions.

- Fast convergence of SEG + Lipschitz line search for a class of saddle point problems.

* Additional assumptions apply

SLS optimizes faster and better





ResNet-34 on CIFAR100

Other results: Deep matrix factorization, Kernels, Min-max games

Try SLS in your project

1. Install our optimizer

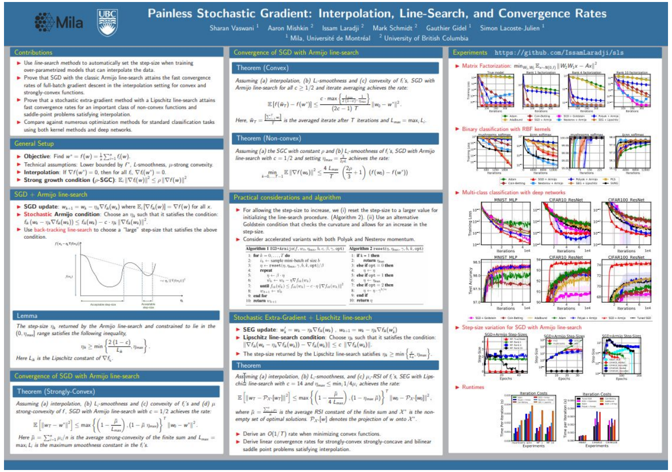
```
pip install --upgrade git+https://github.com/IssamLaradji/sls.git
```

2. Check out our Github code:

https://github.com/IssamLaradji/sls

3. Read the paper:

https://arxiv.org/abs/1905.09997



Come to our poster
Hall B + C #123
East Exhibition
Tue Dec 10th
5:30 PM