Investigating the effect of staleness in Parameter Server

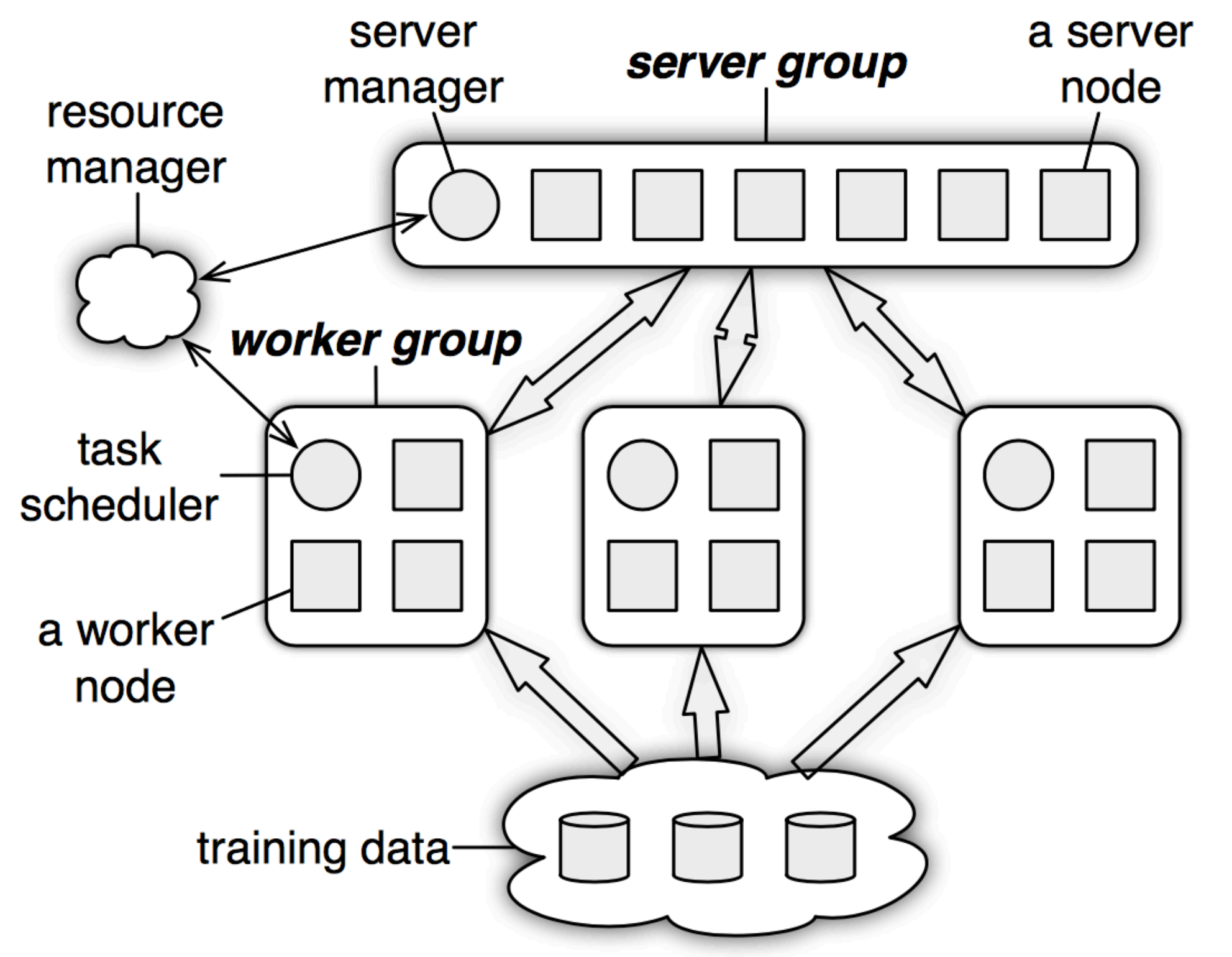
Wei Cui, Zhixun Tan

[1]Department of Computer Science, Carnegie Mellon University

Abstract

We implemented a light-version of the parameter server framework applied widely in large scale distributed machine learning. Using our implementation, we further investigated the effect of the staleness of different gradient-based optimization algorithms, with some common problems in machine learning, including linear regression (with and without regularization), logistic regression and SVM. Our experiment results indicate that gradient-based optimization algorithms are quite robust against staleness, as long as the learning rate is chosen properly.

Parameter Server



Architecture of parameter server.

Parameter Server is a powerful framework for solving engineering challenges common in large scale machine learning and optimization problems, such as efficient communication, flexible consistency models, scalability and fault tolerance. The framework is motivated by the observation that most problems have parameters in the form of vectors/matrices/tensors, and each worker node need access only part of the parameters. Shared parameters are stored in the server group, and the schedulers assigns tasks to each worker. Workers will do computation locally, mostly gradient-related calculations, and push the computation results to servers, and then pull the parameters from server to start the next iteration of computation. Server groups aggregate updates from workers and answer the pull requests from workers with the aggregated parameters. A sample usage of parameter server is listed as below.

Algorithm for distributed proximal gradient descent

```
**Worker $r$ at iteration $t$**
1: Wait until all iterations before $t - \tau$ are finished;
2: Compute first-order gradient $g_r^{(t)}$ using local parameter and data
3: Push $g_r^{(t)}$ to servers
4: Pull $w_r^{(t+1)}$ from servers
**Servers at iteration $t$**
1: Aggregate gradients to obtain $g_r$
2: Solve $w^{(t+1)} \leftarrow \underset{z}{\operatorname{argmin}} \frac{1}{2\eta_t} \| w^{(t)} - \eta_t g_r - z \|_2^2 + \lambda \| z \|_1$
```

Theoretical Results

Theorem: *Delayed Block Proximal Gradient Method Solving converges to a stationary point in expectation if the learning rate γ_t satisfies $\gamma_t < \frac{M_t}{L_{var} + \tau L_{cov} + \epsilon}$.*

L_{var}, L_{cov}: *the variations in training data across blocks and iterations.*

M_t: *minimal coordinate-specific learning rate at iteration t*

ϵ: *any positive value*

Additional assumption: the gradient of the target function is Lipschitz

The above theorem has a lot of strong assumptions and is problem specific. In general we do not have a nice theorem provides guaranteed convergence for parameter-server-like computation with stale data.

Methodology

Problem selected: linear regression, linear regression with lasso, logistic regression and support vector machine.

Dataset: self-generated data for linear regression, MNIST for logistic regression, dataset from convex optimization course for SVM.

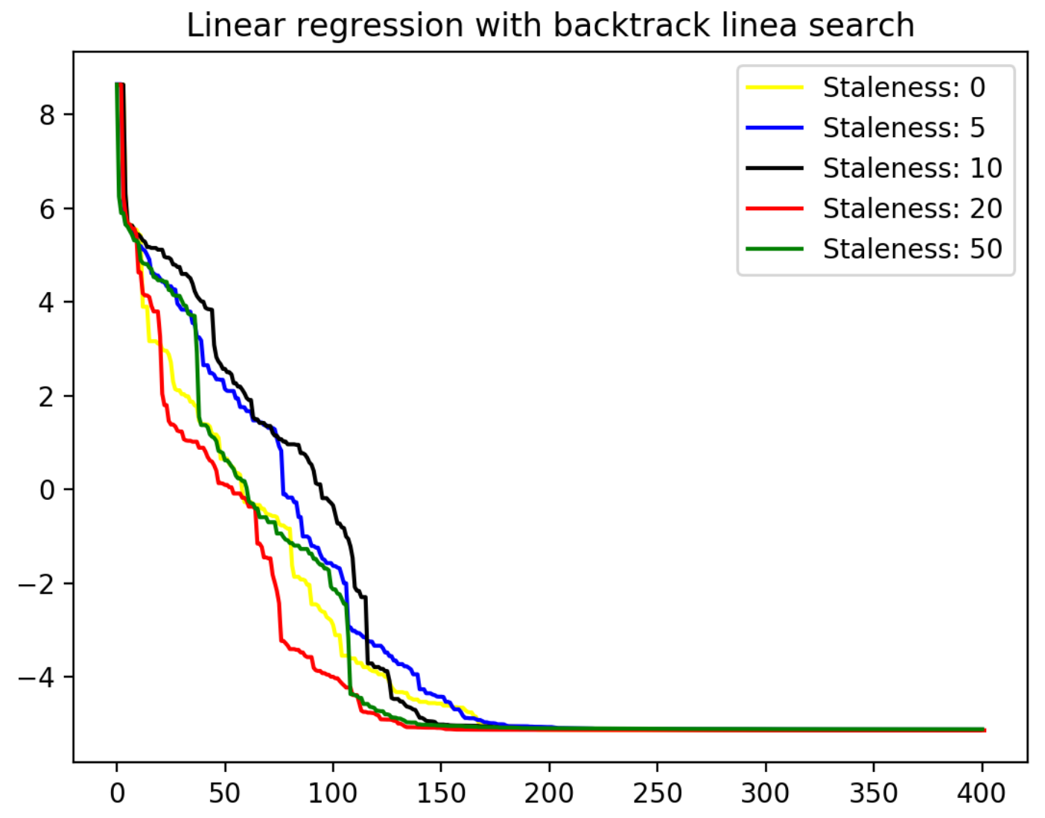
Reference solution: implemented single threaded version of (proximal) gradient descent to find the optimal solution.

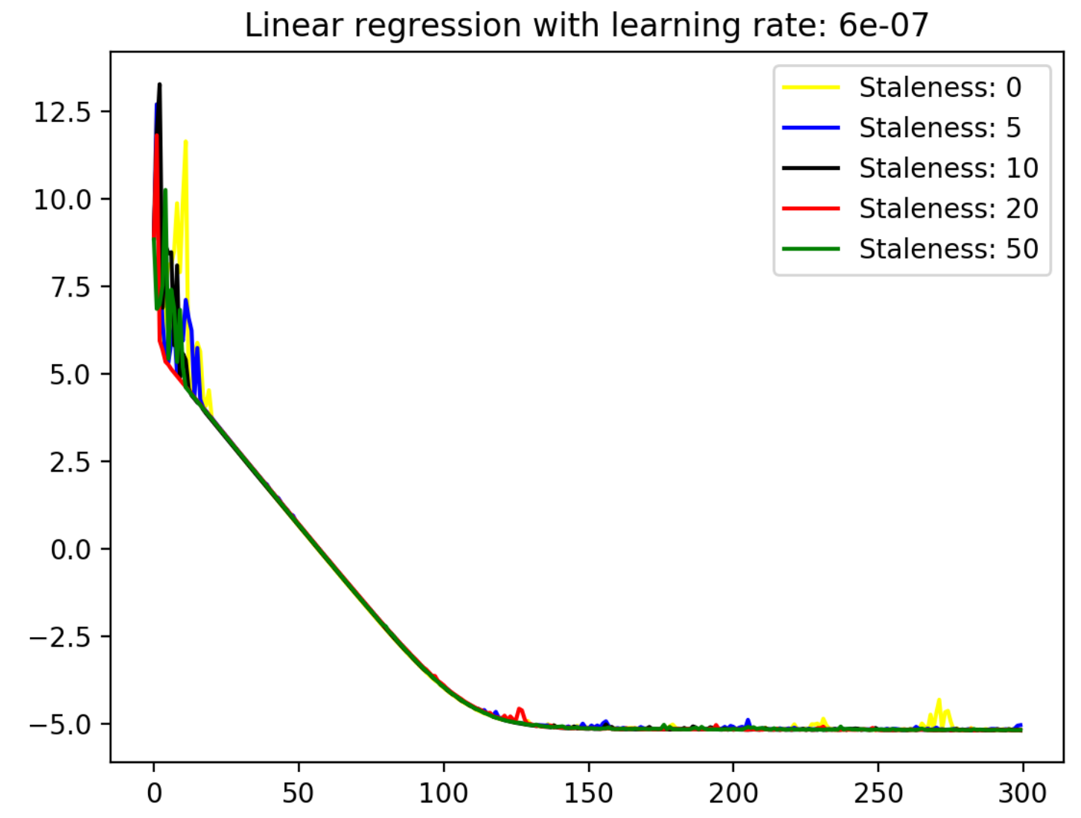
Investigating effect of staleness:

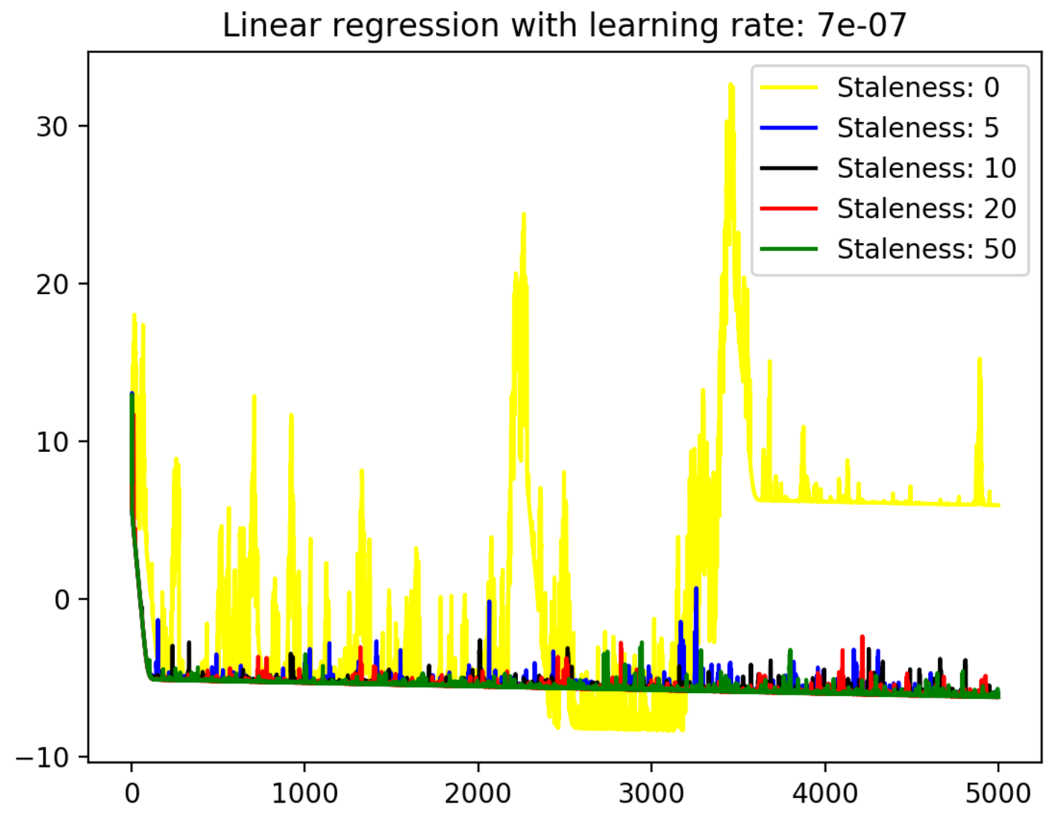
Method 1: We implemented a life-version of parameter server and run our code with it.

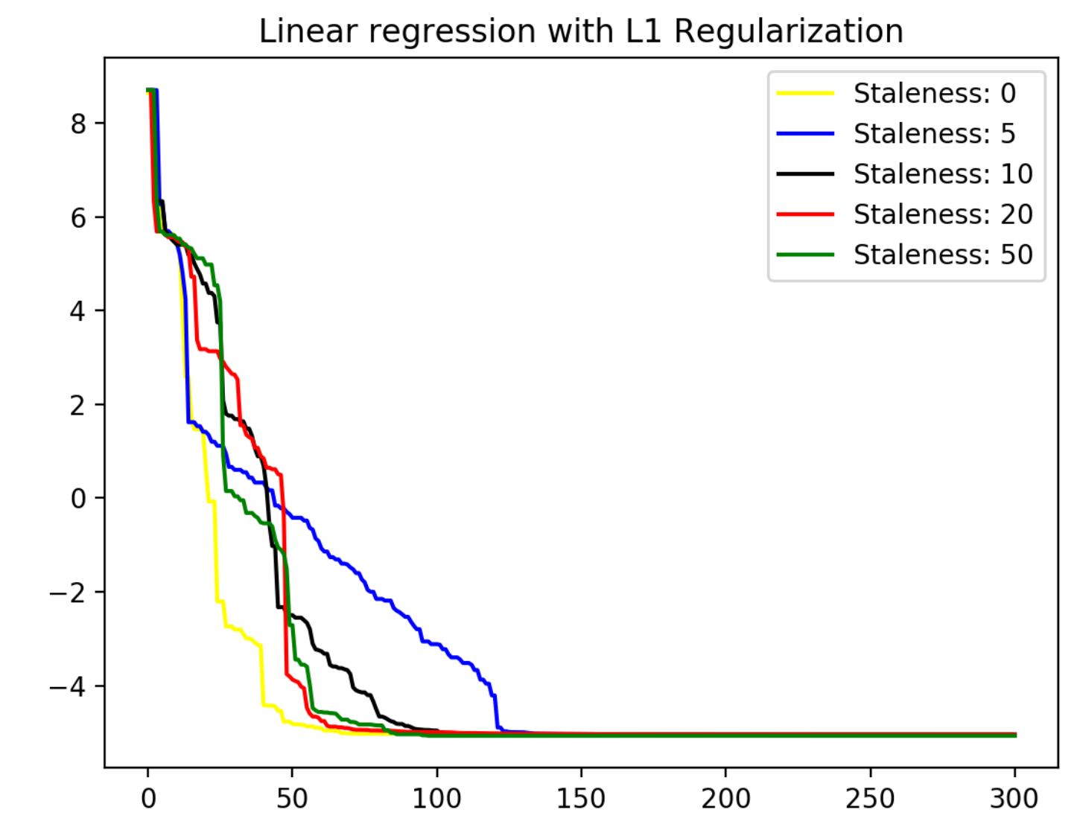
Method 2: Simulation. We observe that on the server side, every event is processed sequentially, and those events are push/pull requests from workers. We write a sequence generator that obeys the staleness requirement and write programs to process those requests as per problem. Both method produce similar results.

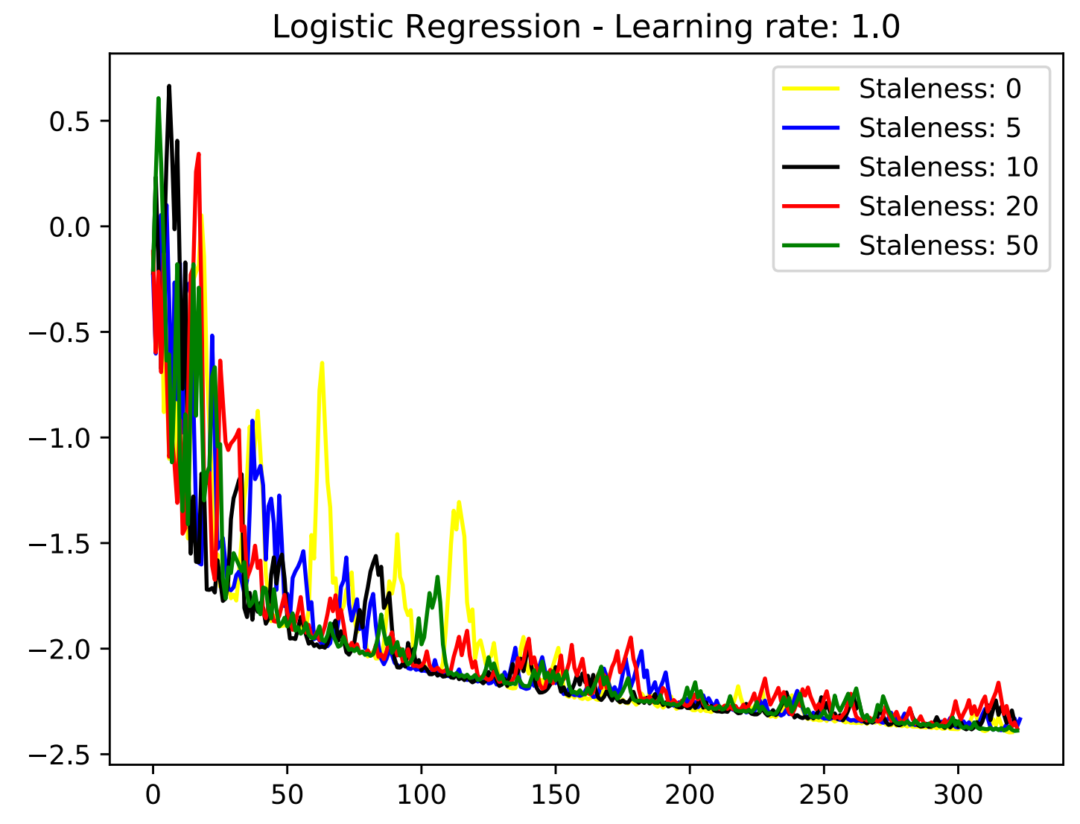
Experiment Results

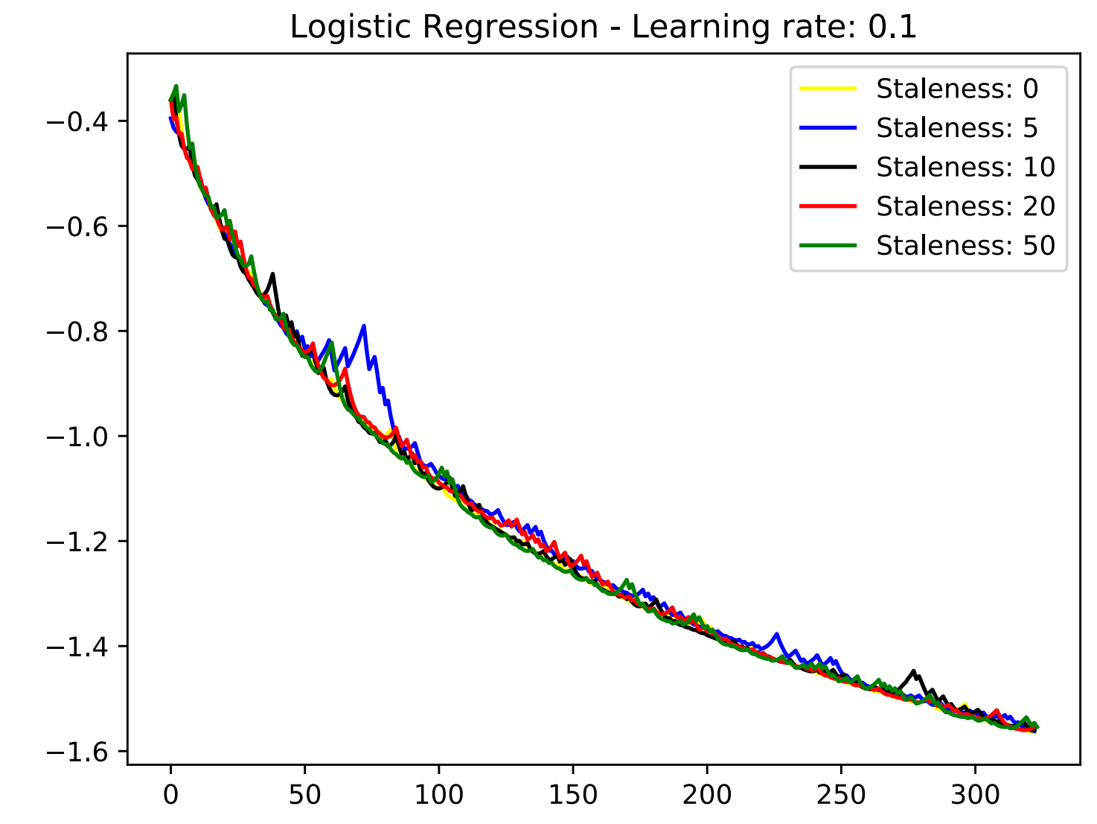


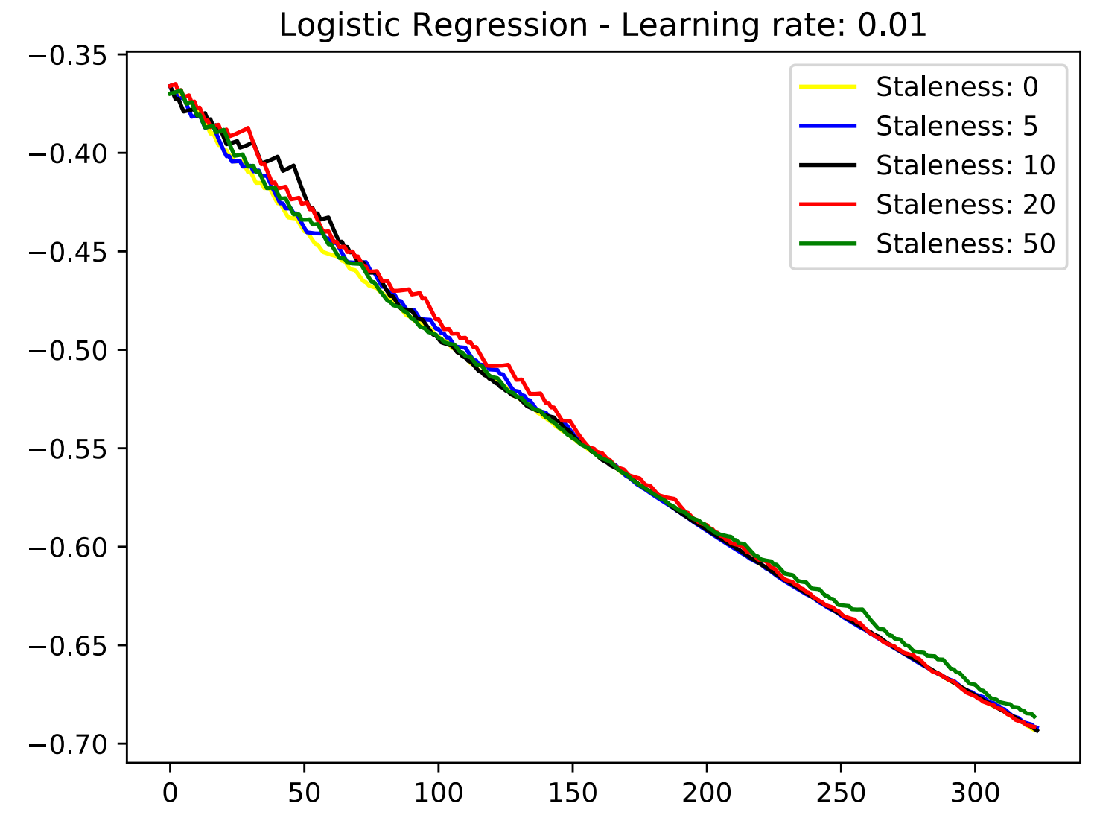


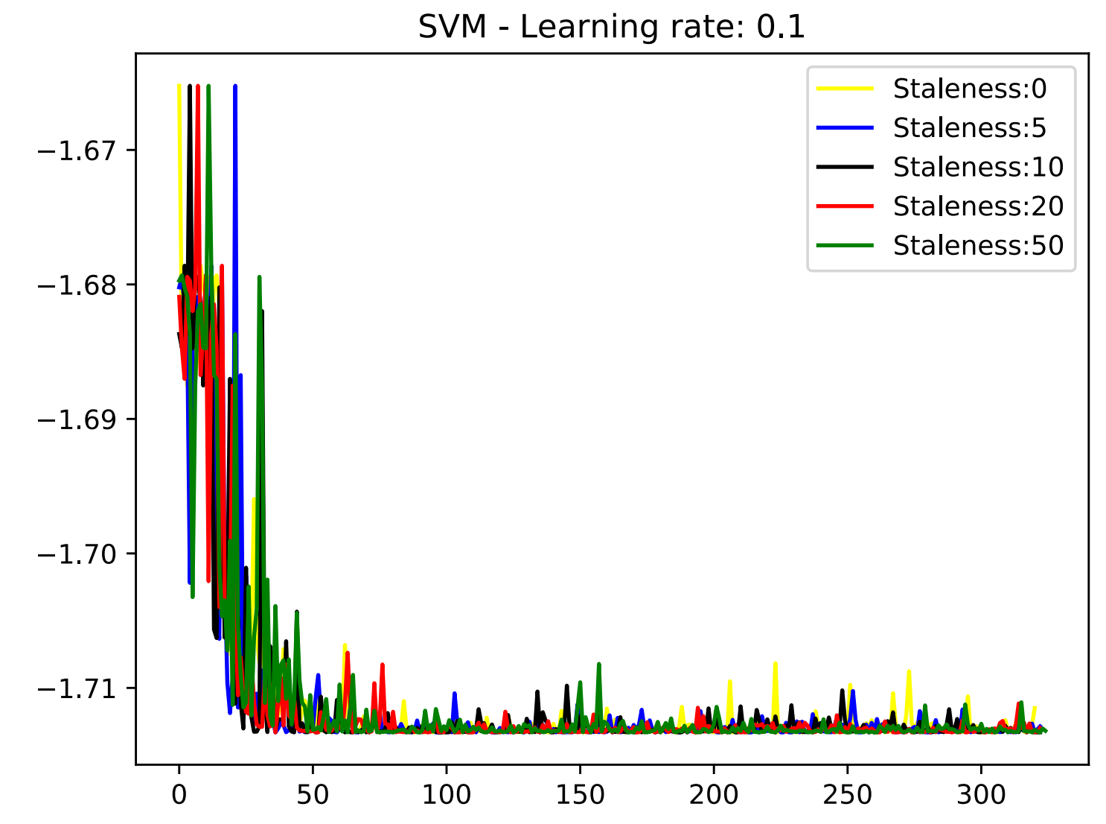


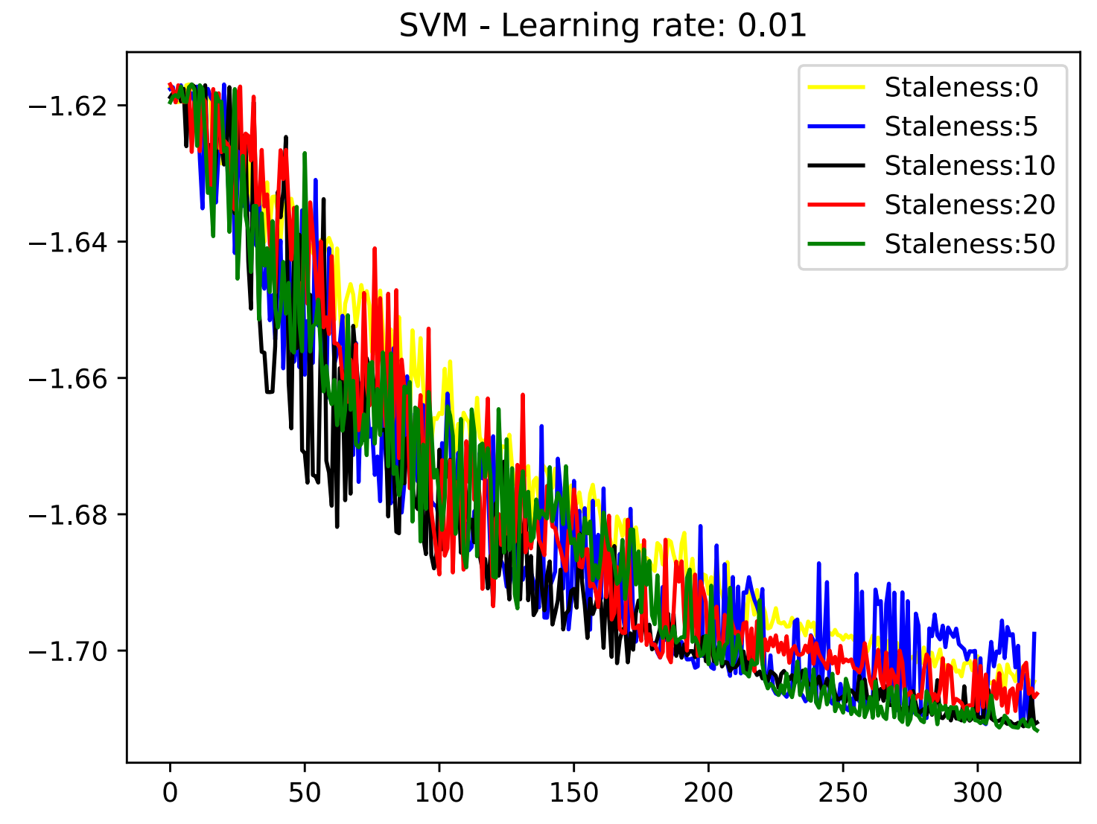


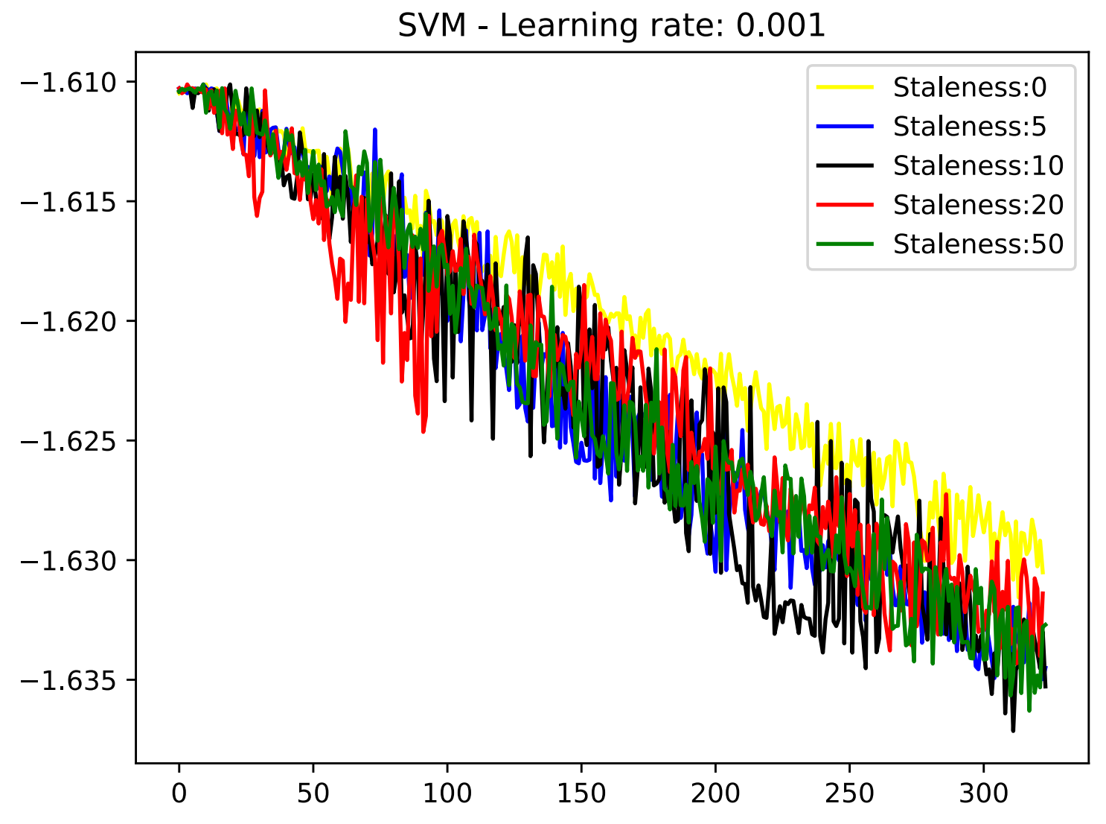












Discussion

From our experiment results, we discovered that

1. Staleness does not necessarily adversely affect convergence. In most ideal cases, the algorithm converges with the same rate with or without staleness.

2. Sometimes staleness can even accelerate convergence. We suspect that staleness effectively introduce momentum similar to accelerated gradient descent.

3. Learning rate is still the most crucial hyperparameter in iterative optimization algorithms.

Bibliography

[1] Mu Li, Li Zhou, Zichao Yang, Aaron Li, Fei Xia, David G Andersen, and Alexander Smola. Parameter server for distributed machine learning. In *Big Learning NIPS Workshop*, volume 6, page 2, 2013.

[2] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *OSDI*, volume 1, page 3, 2014.