

# A Stochastic Quasi-Newton Optimizer for TensorFlow

Jason Chen<sup>1</sup>, David Kraemer<sup>1</sup>

<sup>1</sup>CSE 592: Convex Optimization  
Stony Brook University

May 1, 2018

# Problem overview

- TensorFlow is an open source framework for building neural networks.
- It provides an interface for creating new `Optimizer` objects which perform different minimization algorithms.
- Typical existing implementations are first order descent algorithms, but it lacks<sup>1</sup>any implementation of a BFGS-type algorithm.

---

<sup>1</sup>See: <https://github.com/tensorflow/tensorflow/issues/446>

# Our contribution

- We implemented stochastic dampened (Sd) L-BFGS in TensorFlow following the work of Wang, Ma, Goldfarb, and Liu (2017).
- The approach is based on the iteration step:

$$x_{k+1} = x_k - \alpha_k H_k g_k,$$

- $\alpha_k$  is the step size,
- $H_k$  is the L-BFGS approximate Hessian,
- $g_k$  is a batch gradient defined as

$$g_k = \frac{1}{m_k} \sum_{i=1}^{m_k} g(x_k, \xi_{k,i}),$$

- $m_k$  is the batch size,
- $\xi_{k,i}$  is randomly sampled training data.

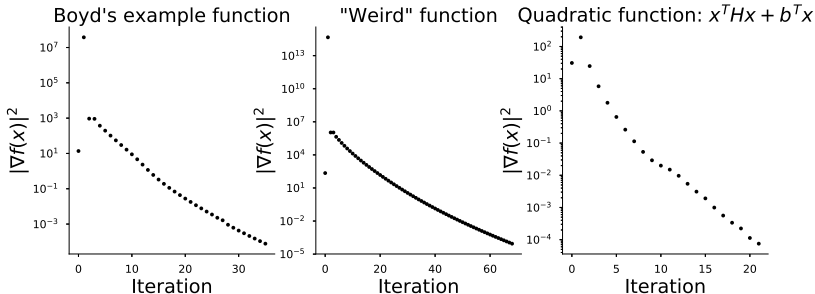
# Our contribution

- The specific approach to approximating the Hessian differs from traditional L-BFGS implementations.
- Wang, et al showed that convergence (in expectation) is guaranteed by applying dampening at specific places in the approximation.
- Crucially, this holds even if the problem is **non-convex**.
- Their update, SdLBFGS, preserves the positive-definiteness of  $H_k$  even in non-convex situations.

# Implementation

- Implementing SdLBFGS proved challenging, and we discovered why it wasn't part of the library already!
- Following the advice from the discussion of Issue 446, we implemented SdLBFGS as an `ExternalOptimizerInterface`.
- This allowed us to write NumPy code but “plug in” to TensorFlow.

# Results: sanity checks



- Since our code is in NumPy, we could test it by running familiar homework functions.
- Here all gradients are deterministic.

# Results: Higgs dataset

Higgs dataset SVM (with hinge loss)

