An Empirical Study of Large-Batch Stochastic Gradient Descent with Structured Covariance Noise

Yeming Wen[*,1,2], **Kevin Luk**[*,3], **Maxime Gazeau**[*,3], **Guodong Zhang**[1,2], **Harris Chan**[1,2], **Jimmy Ba**[1,2]

[1] University of Toronto, [2] Vector Institute, [3] Borealis AI

Abstract

The choice of batch-size in a stochastic optimization algorithm plays a substantial role for both optimization and generalization. Increasing the batch-size used typically improves optimization but degrades generalization. To address the problem of improving generalization while maintaining optimal convergence in large-batch training, we propose to add covariance noise to the gradients. We demonstrate that the learning performance of our method is more accurately captured by the structure of the covariance matrix of the noise rather than by the variance of gradients. Moreover, over the convex-quadratic, we prove in theory that it can be characterized by the Frobenius norm of the noise matrix. Our empirical studies with standard deep learning model-architectures and datasets shows that our method not only improves generalization performance in large-batch training, but furthermore, does so in a way where the optimization performance remains desirable and the training duration is not elongated.
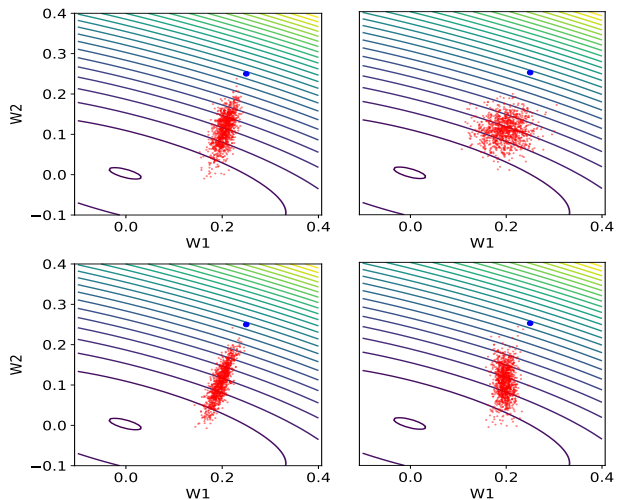
1 Introduction

From a strictly mathematical perspective, training neural networks is a high-dimensional non-convex optimization problem and the dynamics of the training process is incredibly complicated. Despite this, Stochastic Gradient Descent (SGD) and its variants have proven to be extremely effective for training neural networks in practice. Much of the recent successes of deep learning in application tasks such as image recognition (He et al., 2016), speech recognition (Amodei et al., 2016),

Figure 1: Noise structure in a simple two-dimensional regression problem. **Top-left**: One-step SGD update. **Top-right**: One-step SGD update with isotropic Gaussian ($\sigma = 0.1$) noise. **Bottom-left**: One-step SGD update with full Fisher noise. **Bottom-right**: One-step SGD update with diagonal Fisher noise. The full Fisher noise almost recovers the SGD noise. Observe that the full Fisher noise direction is perpendicular to the contours of the loss surface. Moreover, full Fisher exhibits slower convergence than diagonal Fisher; we refer to Section 4 for a more detailed analysis.

natural language processing (Wu et al., 2016) and game playing (Mnih et al., 2015) can be seen as testaments to the effectiveness of SGD.

The choice of a batch-size plays an important role in the learning behavior of SGD. Taking larger batch-sizes ensures better gradient estimation which typically leads to faster training convergence. However, there is a tradeoff from the viewpoint of generalization; the intrinsic noise stemming from mini-batch gradients provides regularization effects (Chaudhari and Soatto, 2017; Smith and Le, 2017) and by increasing batch-sizes, we lose such generalization benefits. It is then an interesting question to ask whether large-batch can be engineered in a way such that generalization significantly improves but at the same time not sacrificing too much the training convergence. This is exactly the

central objective of our paper.

To address this question, we propose to add a noise term whose covariance structure is given by the diagonal Fisher matrix to the large-batch gradient updates. We discuss the motivations underlying our approach. Under the standard log-likelihood loss assumption, the difference of large-batch gradients and small-batch gradients can be modeled as a Fisher noise. We can expect that adding this noise directly to large-batch gradients will yield small-batch performance. While this may resolve generalization issues associated with large-batch training (Keskar et al., 2016; Hoffer et al., 2017), the resulting convergence performance is undesirable. To attain our ultimate goal of designing a method which enjoys desirable optimization and generalization performance simultaneously, we reduce the noise level by changing the covariance structure from full Fisher to diagonal Fisher.

Variance is commonly regarded as a criteria of optimization performance. However, for our proposed method in this paper, studying the gradient variance is not sufficient in deducing any information on the optimization behavior. Rather, it is the structure of the covariance matrix of the noise which plays a critical role. For large-batch training with diagonal Fisher, we find that despite having a high gradient variance, it still attains an ideal optimization performance.

Outline and Summary of Main Contributions. We begin in Section 2 by introducing the basic framework and necessary definitions. We consider different covariance structures for large-batch training in Section 2.2 and then propose the choice of diagonal Fisher. Sections 3 and 4 constitute the central contributions of the paper. The primary takeaways are:

- Gradient variance is not an accurate indicator of optimization behavior. In Fig. 3, we find empirically that the convergence of large-batch with diagonal Fisher is much faster than that of large-batch with full Fisher and small-batch. However, in Fig. 2, we find that all these regimes share roughly the same average gradient variance.

- The main theoretical contribution is Theorem 4.1. We show over the convex quadratic setting, the convergence can be characterized by the Frobenius norm of the noise covariance matrix. In Fig. 5(a), we show empirically that this carries over to the non-convex deep learning context.

In Section 5, we apply our methodology to address the "generalization gap" problem. We show that within the same number of training epochs, large-batch with diagonal Fisher can attain generalization performance

roughly comparable to that of small-batch. Related works are discussed in Section 6 and we close the paper in Section 7.

2 Preliminaries and Approach

2.1 Preliminary Background

Excess Risk Decomposition. We work in the standard framework of supervised learning. Let $\mathcal{D}$ be the unknown joint probability distribution over the data domain $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the target space. We have a training set $\mathcal{S} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ of N input-target samples drawn i.i.d. from $\mathcal{D}$. The family of classifiers of interest to us are neural network outputs $f(x_i, \theta)$, where $\theta \in \mathbb{R}^d$ are parameters of the network. Let $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be the loss function measuring the disagreement between outputs $f(x_i, \theta)$ and targets y_i. For convenience, we use the notation $\ell_i(\theta)$ to denote $\ell(f(x_i, \theta), y_i)$. The expected risk and empirical risk functions are defined to be

$$\mathscr{L}(\theta) := \mathbb{E}_{(x,y) \sim \mathcal{D}}[\ell(f(x, \theta), y)], \ \mathcal{L}(\theta) := \frac{1}{N} \sum_{i=1}^{N} \ell_i(\theta).$$

The standard technique to analyze the interplay between optimization and generalization in statistical learning theory is through excess risk decomposition. The excess risk, after k iterations, is defined as:

$$\Delta_k := \mathscr{L}(\theta_k) - \inf_{\theta \in \mathbb{R}^d} \mathscr{L}(\theta).$$

From Bottou and Bousquet (2008); Chen et al. (2018), the expected excess risk can be upper-bounded by

$$\mathbb{E}_{\mathcal{S}}[\Delta_k] \leq \underbrace{\mathbb{E}_{\mathcal{S}}[\mathscr{L}(\theta_k) - \mathcal{L}(\theta_k)]}_{\mathcal{E}_{\mathrm{gen}}} + \underbrace{\mathbb{E}_{\mathcal{S}}[\mathcal{L}(\theta_k) - \mathcal{L}(\theta^*)]}_{\mathcal{E}_{\mathrm{opt}}}.$$

$$(1)$$

where $\theta^* = \mathrm{argmin}_\theta \mathcal{L}(\theta)$ here is the empirical risk minimizer. The terms $\mathcal{E}_{\mathrm{gen}}$ and $\mathcal{E}_{\mathrm{opt}}$ are the expected generalization and optimization errors respectively. It is often the case that optimization algorithms are studied from one perspective: either optimization or generalization. The decomposition in Eqn. 1 indicates that both aspects should be analyzed together (Bottou and Bousquet, 2008; Chen et al., 2018); since the goal of a good optimization-generalization algorithm in machine learning is to minimize the excess risk in the least amount of iterations.

2.2 Motivations and Approach

We begin by formalizing the setup. Let B_L denote large-batch and $M_L = |B_L|$ denote the size of the

large-batch. We consider the following modification of large-batch SGD updates

$$\theta_{k+1} = \theta_k - \alpha_k \nabla \mathcal{L}_{M_L}(\theta_k) + \alpha_k C(\theta_k)\xi_k. \quad (2)$$

where α_k is the learning rate, $\xi_k \sim \mathcal{N}(0, I_d)$ is the multivariate Gaussian distribution with mean zero and identity covariance, and $\nabla \mathcal{L}_{M_L}(\theta_k) = \frac{1}{M_L}\sum_{i \in B_L} \nabla \ell_i(\theta_k)$ is the large-batch gradient. We can interpret Eqn. 2 as modifying large-batch SGD by injecting Gaussian noise with mean zero and covariance $C(\theta_k)C(\theta_k)^\top$ to the gradients. The central goal of this paper is to determine a suitable matrix $C(\theta_k)$ such that the excess risk of the algorithm in Eqn. 2 is minimized; in more concrete terms, it achieves low optimization and generalization error simultaneously within a reasonable computational budget.

2.2.1 Intrinsic SGD Noise

Let $B \subset \mathcal{S}$ be a mini-batch drawn uniformly and without replacement from $\mathcal{S}$ and $M = |B|$ be the size of this chosen mini-batch. We can write the SGD update rule here as

$$\theta_{k+1} = \theta_k - \alpha_k \nabla \mathcal{L}_M(\theta_k)$$
$$= \theta_k - \alpha_k \nabla \mathcal{L}(\theta_k) + \alpha_k(\underbrace{\nabla \mathcal{L}(\theta_k) - \nabla \mathcal{L}_M(\theta_k)}_{\delta_k})$$

where $\nabla \mathcal{L}(\theta_k) = \frac{1}{N}\sum_{i=1}^{N} \nabla \ell_i(\theta_k)$ is the full-batch gradient. The difference $\delta_k = \nabla \mathcal{L}(\theta_k) - \nabla \mathcal{L}_M(\theta_k)$ is the intrinsic noise stemming from mini-batch gradients. The covariance of δ_k is given by

$$\frac{N-M}{M}\frac{1}{N}\sum_{i=1}^{N}\left(\nabla \mathcal{L}(\theta_k) - \nabla \ell_i(\theta_k)\right)\left(\nabla \mathcal{L}(\theta_k) - \nabla \ell_i(\theta_k)\right)^\top \quad (3)$$

This result can be found in Hu et al. (2017); Hoffer et al. (2017). Moreover, this type of noise has been studied in Zhu et al. (2018). For the purposes of this paper, we assume that the loss is taken to be negative log-likelihood, $\ell_i(\theta_k) = -\log p(y_i|x_i, \theta_k)$ where $p(y|x,\theta)$ is the density function for the model's predictive distribution. Moreover, we assume that the gradient covariance matrix above can be approximated by

$$\frac{N-M}{M}\underbrace{\frac{1}{N}\sum_{i=1}^{N} \nabla \log p(y_i|x_i, \theta_k)\nabla \log p(y_i|x_i, \theta_k)^\top}_{F(\theta_k)}, \quad (4)$$

where (x_i, y_i) are sampled from the empirical data distribution. In the literature, the matrix $F(\theta_k)$ above is often referred to as the empirical Fisher matrix (Martens, 2014). We make this approximation for two reasons. First, computing the full-batch gradient

$\nabla \mathcal{L}(\theta_k)$ at every iteration is not feasible computationally. Secondly, from Fig. 3, we find empirically that the training dynamics of a large-batch regime with empirical Fisher is very close to a small-batch regime (which by the above analysis should be captured by large-batch with empirical covariance in Eqn. 3); suggesting that it is a reasonable assumption to make.

For the remainder of this paper, unless otherwise specified, all mentions of "Fisher matrix" or $F(\theta)$ refers to the empirical Fisher. For completeness, we provide explicit expressions of diagonal Fisher for feed-forward and convolutional network architectures in Appendix A.3.

2.2.2 Naive Choices of Covariance Matrices

We begin by considering the choice of $C(\theta_k) = 0$. In this case, Eqn. 2 is just standard large-batch gradient descent. Since large-batches provide better gradient estimation, we can expect better training error per parameter update. However, from the perspective of generalization, it has been observed in LeCun et al. (1998); Keskar et al. (2016); Hoffer et al. (2017) that using larger batch-sizes can lead to a decay in generalization performance of the model.

Now, let B_S, B_L denote small-batch and large-batch, $M_S = |B_S|, M_L = |B_L|$ denote the size of small-batch and large-batch. Consider $C(\theta_k)$ to be

$$C(\theta_k) = \sqrt{\frac{M_L - M_S}{M_L M_S}}\sqrt{F(\theta_k)}. \quad (5)$$

Now, if the intrinsic SGD noise is reasonably approximated as a Gaussian distribution with mean zero and covariance given by $C(\theta_k)$ above, then Eqn. 2 with this choice of $C(\theta_k)$ should exhibit similar behavior as small-batch. If this is the case, then we can expect that Eqn. 2 exhibits poor convergence. Indeed, as shown on a 2D convex example in Fig. 1, choosing $C(\theta_k)$ as in Eqn. 5 essentially recovers SGD behavior. Furthermore, on the CIFAR-10 image classification task trained using ResNet-44 in Fig. 3, we find that adding this noise significantly worsens the training convergence. Thus, choosing $C(\theta_k)$ as in Eqn. 5 does not satisfy our objective of simultaneously attaining desirable convergence and generalization for large-batch training.

2.2.3 Using Diagonal Fisher

We now propose to take a "middle ground" and choose $C(\theta_k)$ to be

$$C(\theta_k) = \sqrt{\frac{M_L - M_S}{M_L M_S}}\sqrt{\text{diag}\left(F(\theta_k)\right)}. \quad (6)$$

A formal statement is given in Algorithm 1. Changing from full Fisher to diagonal Fisher has important

---

Algorithm 1 Adding diagonal Fisher noise to large-batch SGD. Differences from standard large-batch SGD are highlighted in blue

---

Require: Number of iterations K, initial step-size α_0, large-batch B_L of size M_L, small-batch B_S of size M_S, initial condition $\theta_0 \in \mathbb{R}^d$

    for $k = 1$ to K **do**

        $\xi_k \sim \mathcal{N}(0, I_d)$

        $\epsilon_k = \alpha_k \sqrt{\frac{M_L - M_S}{M_L M_S}} \sqrt{\text{diag}\left(F(\theta_k)\right)} \xi_k$

        $\theta_{k+1} = \theta_k - \alpha_k \nabla \mathcal{L}_{M_L}(\theta_k) + \epsilon_k$

    end for

Figure 2: Average variance of gradients for **LB**, **SB**, **LB** with full Fisher and **LB** with diagonal Fisher.

implications for both optimization and generalization behavior. In our empirical analysis in Sections 3 and 5, we show that Algorithm 1 can achieve both desirable convergence and generalization performance within an epoch training budget; which implies that the excess risk is minimized.

With regards to computational complexity, computing diagonal Fisher only introduces minor overhead. Goodfellow (2015) shows that it can be done at the cost of one forward pass.

3 Variance and Optimization

The objective of this section is to examine the optimization performance of the four regimes: large-batch with $C(\theta_k)$ equal to 0 (standard large-batch), large-batch with $C(\theta_k)$ equal to diagonal Fisher as in Eqn. 6, large-batch with $C(\theta_k)$ equal to full Fisher as in Eqn. 5 and small-batch. In the experimentation, we fix large-batch to be 4096 and small-batch to be 128. For conciseness, we set forth the notation **LB** and **SB** for large-batch and small-batch respectively.

In Fig. 3, we compare the training error (measured per parameter update) of ResNet44 (CIFAR-10) of the four regimes. The same learning rate is used across all four regimes (better training error can be obtained if we tune the learning rate further for the **LB** regimes). We find that **LB** with diagonal Fisher trains much faster than **LB** with full Fisher and **SB**. In contrast, **LB** with diagonal Fisher attains a convergence similar to **LB**, demonstrating that adding this particular form of noise does not hamper much the optimization performance.

We now analyze the gradient variance of each of the four regimes. We define gradient variance here to be the trace of the covariance matrix of the gradients. In the experiment depicted in Fig. 2, we provide an estimation of the variance of gradients of the four regimes. The experiment is performed as follows: we freeze a partially-trained network and compute Monte-Carlo estimates of gradient variance with respect to each pa-
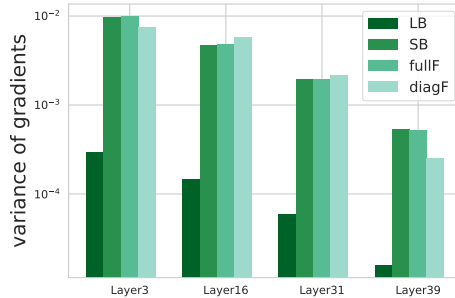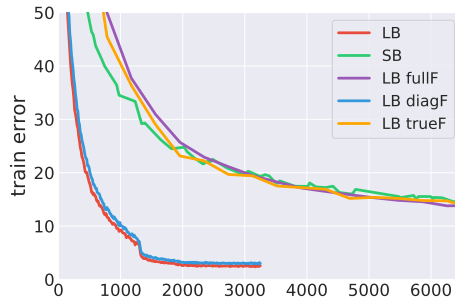

Figure 3: Training error per parameter update for **SB**, **LB**, **LB** with full Fisher and **LB** with diagonal Fisher.

rameter over different mini-batches. This variance is then averaged over the parameters within each layer.

In Fig. 2, we find that **LB** with diagonal Fisher, **LB** with full Fisher, and **SB** all share roughly the same gradient variance meanwhile **LB** has a much lower one.

However, the optimization behaviors are completely different. These experiments show that the number of iterations needed to reach a small optimization error is not purely determined by the gradient variance. Many recent works have suggested to add isotropic noise to the gradient dynamics to escape from saddle point or local minima (Ge et al., 2015; Jin et al., 2017). We believe that the number of iterations needed to escape saddle points is determined by the nature of the noise. A similar observation has been made in Zhu et al. (2018) where the authors studied the effects of how using Fisher covariance noise relates to the efficiency of escaping from local minima and compare it to isotropic noise.

In the next section, we analyze the efficiency of the algorithm in Eqn. 2 for a convex quadratic when the covariance matrix is given by the (exact true) Fisher matrix. We prove that it requires less iterations to reach the global optimum compared to isotropic noise. We iterate that this choice of diffusion matrix is completely specific to the convex quadratic example.

4 Convex Quadratic Example

4.1 Motivations

In this section, we analyze the optimization behavior of our proposed algorithm for the convex quadratic model. Analyzing the convex quadratic model serves as a good proxy to understand the complex dynamics of neural network training. Although the optimization landscape of neural networks is non-convex, using such a model marginalizes away inessential features and enables us to tractably analyze optimization phenomena of neural networks. There is ample evidence suggesting this: the recent work of Zhang et al. (2019) uses the convex quadratic model to accurately predict critical batch-sizes for commonly used optimizers in neural networks. Short-horizon bias phenomena of optimized learning rates were studied in Wu et al. (2018) for the convex quadratic and their theoretical insights were successfully translated to neural networks. Furthermore, a convex quadratic objective can always be obtained by first linearizing around a given parameter vector and then taking a second-order Taylor approximation. Recent empirical work (Lee et al., 2019) have demonstrated that linearized approximations do indeed match the training phenomena of large yet realistic networks.

Therefore, approximating the loss surface of a neural network with a convex quadratic has proven to be a fertile "testing ground" when introducing new methodologies in deep learning. Analyzing the toy quadratic problem has led to important insights; for example, in learning rate scheduling (Schaul et al., 2013) and formulating SGD as approximate Bayesian inference (Mandt et al., 2017).

4.2 Analysis

For strongly-convex objective functions and diminishing step-sizes, the expected optimality gap is bounded in terms of the second-order moment of the gradients (Bottou et al., 2018). However, in practice, different algorithms having the same gradient moments might not need the same number of iterations to converge to the minimum.

Consider the loss function

$$\mathcal{L}(\theta) = \frac{1}{2}\theta^\top A\theta,$$

where A is a symmetric, positive-definite matrix. We focus on the algorithm in Eqn. 2 and consider a constant $d \times d$ covariance matrix C. The following theorem, adapted from Bottou et al. (2018), analyzes the convergence of this optimization method. The proof is relegated to Section A.1 of Appendix.
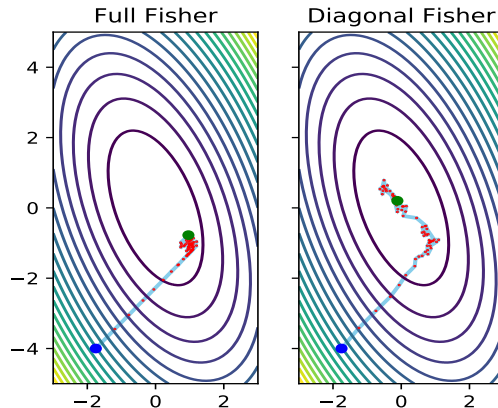

Figure 4: Trajectory using full Fisher versus diagonal Fisher noise for the algorithm in Eqn. 2 used to minimize a two-dimensional quadratic function. Blue dot indicates the initial parameter value and the green dot shows the final parameter value. We used a learning rate of 0.1 for 500 iterations (plotting every 10 iterations). Observe that adding diagonal Fisher to the gradient achieves faster convergence than full Fisher.

Theorem 4.1. *Let $\lambda_{\max}$ and $\lambda_{\min}$ denote the maximum and minimum eigenvalue of A respectively. For a chosen $\alpha_0 \leq \lambda_{\max}^{-1}$, suppose that we run the algorithm in Eqn. 2 according to the decaying step-size sequence*

$$\alpha_k = \frac{2}{(k+\gamma)\lambda_{\min}},$$

for all $k \in \mathbb{N}_{>0}$ and where γ is chosen such that $\alpha_k \leq \alpha_0$. Then for all $k \in \mathbb{N}$,

$$\mathbb{E}[\mathcal{L}(\theta_k)] \leq \frac{\nu}{k+\gamma}$$

where

$$\nu = \max\left(\frac{2\operatorname{Tr}(C^\top AC)}{\lambda_{\min}^2}, \gamma\mathcal{L}(\theta_0)\right).$$

We make a couple of observations concerning this bound. First, the convergence rate is optimal when $C = 0$ which is expected. In this case, there is no noise and hence we obtain no regularization benefits which leads to poor generalization. A more formal discussion is given at the end of Section A.2 in the appendix where if we employ a scaling factor $C_\lambda := \lambda C$; as $\lambda \to 0$, the expected generalization error becomes worse.

The second observation is that the term of importance in this theorem is $\operatorname{Tr}(C^\top AC)$. While the overall convergence rate of the algorithm is $O(1/k)$, the discrepancy in convergence performance for different choices of the matrix C rests entirely on this term. The number of iterations for the algorithm in Eqn. 2 to reach the unique minimum depends entirely on $\operatorname{Tr}(C^\top AC)$ and not on the second-order moment.

Table 1: Number of iterations needed for the algorithm in Eqn. 2 to reach an ϵ error for a strongly-convex quadratic loss function. Since $\|\text{diag}(A)\|_{\text{Frob}}^2$ is smaller than $\|A\|_{\text{Frob}}^2$, less iterations are required when choosing $C = \sqrt{\text{diag}(A)}$.

COVARIANCE MATRIX C	$\sqrt{A}$	$\sqrt{\text{diag}(A)}$
STEPS k TO REACH ϵ ERROR	$\|A\|_{\text{Frob}}^2 / \epsilon$	$\|\text{diag}(A)\|_{\text{Frob}}^2 / \epsilon$

We analyze two specific cases which are relevant for us: the first case where C is square-root of A, $C = \sqrt{A}$, and the second case where C is the square-root of the diagonal of A, $C = \sqrt{\text{diag}(A)}$. The second-order moment of the noise perturbation is the same for both and is given by

$$\mathbb{E}_\xi[\|C\xi_k\|^2] = \text{Tr}(C^\top C) = \text{Tr}(A). \qquad (7)$$

However, it is different for $\text{Tr}(C^\top A C)$; in the case of $C = \sqrt{A}$, we get

$$\text{Tr}(C^\top A C) = \text{Tr}(A^2) = \|A\|_{\text{Frob}}^2,$$

and for the case of $C = \sqrt{\text{diag}(A)}$,

$$\text{Tr}(C^\top A C) = \text{Tr}(\text{diag}(A)^2) = \|\text{diag}(A)\|_{\text{Frob}}^2.$$

Thus, the difference in training performance between the two cases can be measured by the difference of their respective Frobenius norms and less number of iterations are needed with the choice of $\sqrt{\text{diag}(A)}$. This suggests that the off-diagonal elements of A play a role in the optimization performance. In Fig. 4, we provide a visualization of the difference between $C = \sqrt{A}$ and $C = \sqrt{\text{diag}(A)}$ over a two-dimensional quadratic function.

We summarize our observation in Table 1: different choices of covariance matrix C impacts the number of iterations required to reach an ϵ error.

Limitations: We mention a couple of limitations of the current analysis. In the previous analysis, the matrix A above is the Hessian which is also, in this specific setup, equal to the (exact true) Fisher defined as the expectation of the outer product of log-likelihood gradients,

$$\mathbb{E}_{P_x, P_{y|x}}[\nabla \log p(y|x, \theta) \nabla \log p(y|x, \theta)^\top] \qquad (8)$$

The expectation here is taken with respect to the data distribution P_x for inputs x and the model's predictive distribution $P_{y|x}$ for targets y. For much of this paper, we have been working with the empirical Fisher instead. We believe this to be a reasonable approximation; in Fig. 3, we find that the training curves for **LB** + full Fisher and **LB** + true Fisher are almost identical. Furthermore, if we assume that the implicit conditional distribution over the network's output is close to the conditional distribution of targets from

the training distribution, then the covariance of the gradients closely matches the Hessian (Martens, 2014). The recent work of Zhang et al. (2019) shows that this relationship indeed holds tightly in empirical settings.

Secondly, we have focused solely on the optimization performance of our proposed method. While it would be desirable to accompany this with a complete theoretical analysis of generalization performance, this is beyond the current scope of the paper. However, in Appendix A.2, we use the framework of uniform stability to provide some theoretical insights on how different choices of the covariance matrix impact generalization.

5 Experiments

5.1 Experimentation Details

Batch Normalization. For all experiments involving **LB**, we adopt Ghost Batch Normalization (GBN) (Hoffer et al., 2017) and hence **LB** throughout stands for **LB** with Ghost Batch Normalization. This allows a fair comparison between **LB** and **SB**, as it ensures that batch normalization statistics are computed on the same number of training examples. Using standard batch normalization for large batches can lead to degradation in model quality (Hoffer et al., 2017).

Models and Datasets. The network architectures we use are fully-connected networks, shallow convolutional networks (LeNet (LeCun et al., 1998), AlexNet (Krizhevsky et al., 2012)), and deep convolutional networks (VGG16 (Simonyan and Zisserman, 2014), ResNet44 (He et al., 2016), ResNet44x2 (the number of filters are doubled)). These models are evaluated on the standard deep-learning datasets: MNIST, Fashion-MNIST (LeCun et al., 1998; Xiao et al., 2017), CIFAR-10 and CIFAR-100 (Krizhevsky and Hinton, 2009).

5.2 Frobenius Norm

Over the convex quadratic setting in Section 4, Theorem 4.1 tells us that the number of iterations to reach optimum is characterized by the Frobenius norm. Hence, the optimization difference between using large-batch with diagonal Fisher versus full Fisher lies in the difference of their respective Frobenius norms.

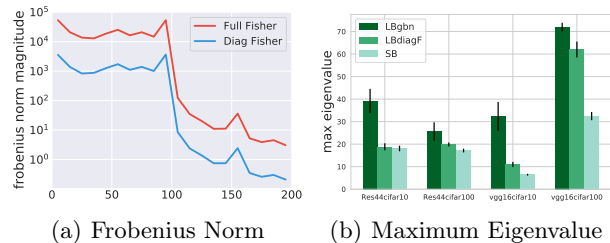We now give an empirical verification of this phenomena

(a) Frobenius Norm (b) Maximum Eigenvalue

Figure 5: a) Frobenius norms of full Fisher and diagonal Fisher along the training trajectory. The model is trained on ResNet44 with CIFAR-10. **b)** Maximum eigenvalue of the Hessian matrix at the end of training for **LB** with Ghost Batch Normalization, **LB** with Ghost Batch Normalization + diagonal Fisher and **SB**. Error bar is computed over 3 random seeds.

in the non-convex setting of deep neural networks. We compute the Frobenius norms during the training of the ResNet44 network on CIFAR-10. Fig. 5(a) shows that the full Fisher matrix has much larger Frobenius norm than the diagonal Fisher matrix, which suggests that using diagonal Fisher noise should have faster convergence than full Fisher noise in the deep neural network setting. Indeed, Fig. 3 shows that **LB** with full Fisher converges at the same rate as **SB** whereas **LB** with diagonal Fisher converges much faster; and in fact, roughly the same as **LB**. This indicates that adding diagonal Fisher noise to **LB** does not degrade the optimization performance of **LB**.

5.3 Maximum Eigenvalue of Hessian

While the relationship between the curvature of the loss surface landscape and generalization is not completely explicit, numerous works have suggested that the maximum eigenvalue of the Hessian is possibly correlated with generalization performance (Keskar et al., 2016; Chaudhari et al., 2016; Chaudhari and Soatto, 2017; Yoshida and Miyato, 2017; Xing et al., 2018). In this line of research, the magnitudes of eigenvalues of the Hessian may be interpreted as a heuristic measure for generalization; the smaller the magnitude the better the model generalizes. To situate our method with this philosophy, we compute the maximum eigenvalue of the Hessian of the final model for the following three regimes: **SB**, **LB**, and **LB** with diagonal Fisher.

We provide the details of this experiment. Computing maximum eigenvalue without any modification to the model gives inconsistent estimates even between different runs of the same training configuration. To make the maximum eigenvalue of the Hessian comparable over different training trajectories, the Hessian needs to be invariant under typical weight reparameterizations; for example, affine transformations (Liao et al., 2018). To achieve this, we make the following modification

to the trained model: (1) For the layers with batch normalization, we can just push the batch-norm layer parameters and running statistics into the layerwise parameter space so that the the layer is invariant under affine transformations; and (2) For the layers without batch normalization, reparameterization changes the prediction confidence while the prediction remains the same. Hence, we train a temperature parameter on the cross-entropy loss on the test data set, this encourages the model to make a calibrated prediction (prevent it from being over-confident). In Fig. 5(b), we give the error bar of the maximum eigenvalue of the Hessian over different runs, which indicates the modification gives a roughly consistent estimate.

In Fig. 5(b), we give the error bar of the maximum eigenvalue of the Hessian over different runs for the regimes we experiment with. The central takeaway here is that across all models with which we experiment, we consistently find that the maximum eigenvalue of the Hessian for **LB** with diagonal Fisher is lower than that of **LB** and in some cases, comparable to **SB**.

5.4 Generalization Gap

In this last part of our empirical studies, we apply our methodology to address the "generalization gap" problem in stochastic optimization. Here, we experiment with four regimes: **SB**, **LB**, **LB** with Fisher Trace and **LB** with diagonal Fisher. All regimes are trained for the same number for epochs. **LB** with Fisher Trace refers to **LB** with the following noise $\sqrt{\mathrm{Tr}(F(\theta_k))/d} \cdot \xi_k, \xi_k \sim \mathcal{N}(0, I_d)$ injected at every iteration. The purpose of this is to provide an experimental comparison which adds isotropic noise with the same variance as **LB** with diagonal Fisher.

We also point out that we do not experiment with **LB** with full Fisher due to its exceeding long training time. This can be seen from Fig. 3 where for ResNet44 trained on CIFAR-10, **LB** with full Fisher does not achieve good convergence even after 8000 parameter updates. There is typically no optimal learning-rate scaling rule for large-batch training across different models and datasets (Shallue et al., 2018); hence, we tune the learning rate schedule to obtain optimal results for each method.

The final validation accuracy numbers are reported in Table 2. While it is true that using **LB** with diagonal Fisher cannot completely close the "generalization gap" in some cases, it yields definite improvements over **SB** within an epoch-training budget. Such a training regime typically favors small-batch training as they perform more parameter updates (Shallue et al., 2018). This highlights that our approach is a data-efficient way to improve generalization for large-batch training.

Table 2: Validation accuracy results on classification tasks for **SB**, **LB**, **LB** + Fisher Trace, and **LB** + diagonal Fisher. The results are averaged over 3 random seeds. All methods in each row are trained with the same number of epochs. While it is indeed not feasible to experiment **LB** with full Fisher for all the models below, we note that this reaches roughly the same validation accuracy (93.22) as **SB** in the case of ResNet44 (CIFAR-10).

DATASET	MODEL	SB	LB	LB+FISHER TRACE	**LB+Diag**
MNIST	MLP	98.10	97.95	98.08	**98.10**
MNIST	LENET	99.10	98.88	99.02	**99.11**
FASHION	LENET	91.15	88.89	90.29	**90.79**
CIFAR-10	ALEXNET	87.80	86.42	N/A	**87.61**
CIFAR-100	ALEXNET	59.21	56.79	N/A	**59.10**
CIFAR-10	VGG16	93.25	91.81	92.91	**93.19**
CIFAR-100	VGG16	72.83	69.45	71.35	**72.11**
CIFAR-10	RESNET44	93.42	91.93	92.33	**92.88**
CIFAR-100	RESNET44x2	75.55	73.13	73.77	**74.26**

In addition, we experimented with other regimes such as injecting multiplicative Gaussian noise with constant covariance as in Hoffer et al. (2017) and replacing diagonal Fisher with the block-diagonal Kronecker-Factored Approximate Curvature (K-FAC) (Martens and Grosse, 2015) [1]. We delegate these results to Section A.8 of Appendix.

6 Related Works

Variance and Optimization. In the context of large-scale learning, stochastic algorithms are very popular compared to full-batch methods due to lower computational overhead (Bottou et al., 2018; Bottou, 1991). The tradeoff is that stochastic algorithms exhibit slower convergence asymptotically due to the inherent noise present in their gradients (Moulines and Bach, 2011; Bottou et al., 2018; Wen et al., 2018). For smooth and strongly-convex functions, variance reduction is a common technique to improve convergence rates (Johnson and Zhang, 2013; Defazio et al., 2014).

In contrast, for non-convex optimization, increasing the variance by adding noise is often times beneficial. Unlike the convex setting, the loss surface is much more complicated and there is an abundance of global minima (Choromanska et al., 2015). Adding noise can significantly improve training since it enables the dynamics to escape saddle points or shallow local minima (Ge et al., 2015; Jin et al., 2017). More specifically for deep learning, injecting annealed gradient noise has been shown to speed up training of very deep neural networks (Neelakantan et al., 2015).

Variance and Generalization. The inherent noise in stochastic optimization methods is also conducive to generalization performance. There are vast bodies of literature devoted to this in deep learning; for example, scaling the learning rate or batch-size (Smith and Le, 2017; Goyal et al., 2017; Hoffer et al., 2017) to augment gradient noise in order to encourage better generalization. More direct approaches of studying the covariance structure of mini-batch gradients have also been explored (Jastrzebski et al., 2017; Xing et al., 2018; Zhu et al., 2018; Li et al., 2015). A closely-related approach to ours is the Stochastic Gradient Langevin Dynamics (SGLD) (Gelfand et al., 1992; Welling and Teh, 2011); a modification of SGD where an annealed isotropic Gaussian noise is injected to the gradients. The recent systematic empirical study of Shallue et al. (2018) demonstrates that links between optimization, generalization, and the choice of batch-size in SGD is extremely complex. It underscores the necessity of a more foundational understanding of the interaction between batch-sizes, model architectures, and other optimization metaparameters.

7 Conclusion

In this paper, we explored using covariance noise in designing optimization algorithms for deep neural networks that could potentially exhibit ideal learning behavior. We proposed to add diagonal Fisher noise to large-batch gradient updates. Our empirical studies showed that this yield significant improvements in generalization while retaining desirable convergence performance. Furthermore, we demonstrated that the structure of the noise covariance matrix encodes much more information about optimization than the variance of gradients. An immediate question which arises is to better understand how the mathematical structure of the noise covariance matrix ties to generalization. For example, in the special cases of diagonal Fisher and full Fisher, our experiments appear to indicate that the generalization performance is somewhat comparable. This seems to suggest that the diagonal elements contribute much more to the generalization performance than the off-diagonal elements. It is an interesting theoretical question as to why this may be the case.

---

[1]This corresponds to the matrix-variate Gaussian noise in Zhang et al. (2018)

References

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

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

Léon Bottou. Stochastic gradient learning in neural networks. In *In Proceedings of Neuro-Nîmes. EC2*, 1991.

Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *arXiv preprint arXiv:1710.11029*, 2017.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*, 2016.

Y. Chen, C. Jin, and B. Yu. Stability and Convergence Trade-off of Iterative Optimization Algorithms. *ArXiv e-prints*, April 2018.

A. Choromanska, M. Henaff, M. Mathieu, G.B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. *Journal of Machine Learning Research*, 38: 192–204, 2015.

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.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping From Saddle Points — Online Stochastic Gradient for Tensor Decomposition. *arXiv e-prints*, art. arXiv:1503.02101, Mar 2015.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

Saul B. Gelfand, Peter C. Doerschuk, and Mohamed Nahhas-Mohandes. Theory and application of annealing algorithms for continuous optimization. In *Proceedings of the 24th Conference on Winter Simulation*, WSC '92, pages 494–499, New York, NY,

USA, 1992. ACM. ISBN 0-7803-0798-4. doi: 10. 1145/167293.167407.

Ian J. Goodfellow. Efficient per-example gradient computations. *CoRR*, abs/1510.01799, 2015.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741, 2017.

Wenqing Hu, Chris Junchi Li, Lei Li, and Jian-Guo Liu. On the diffusion approximation of nonconvex stochastic gradient descent. *arXiv preprint arXiv:1705.07562*, 2017.

Stanislaw Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to Escape Saddle Points Efficiently. *arXiv e-prints*, art. arXiv:1703.00887, Mar 2017.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pages 1724–1732, 2017.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *ArXiv*, abs/1902.06720, 2019.

Qianxiao Li, Cheng Tai, et al. Stochastic modified equations and adaptive stochastic gradient algorithms. *arXiv preprint arXiv:1511.06251*, 2015.

Qianli Liao, Brando Miranda, Andrzej Banburski, Jack Hidary, and Tomaso Poggio. A surprising linear relationship predicts test performance in deep networks. *arXiv preprint arXiv:1807.09659*, 2018.

Stephan Mandt, Matthew D Hoffman, and David M Blei. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.

James Martens. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2014.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

Arvind Neelakantan, Luke Vilnis, Quoc V. Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach, and James Martens. Adding Gradient Noise Improves Learning for Very Deep Networks. *arXiv e-prints*, art. arXiv:1511.06807, November 2015.

Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *International Conference on Machine Learning*, pages 343–351, 2013.

Christopher J Shallue, Jaehoon Lee, Joe Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *arXiv preprint arXiv:1811.03600*, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

S. L. Smith and Q. V. Le. A Bayesian Perspective on Generalization and Stochastic Gradient Descent. *ArXiv e-prints*, October 2017.

Samuel L Smith and Quoc V Le. Understanding generalization and stochastic gradient descent. *arXiv preprint arXiv:1710.06451*, 2017.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, 2018.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Yuhuai Wu, Mengye Ren, Renjie Liao, and Roger B. Grosse. Understanding short-horizon bias in stochastic meta-optimization. *ArXiv*, abs/1803.02021, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Chen Xing, Devansh Arpit, Christos Tsirigotis, and Yoshua Bengio. A walk with sgd. *arXiv preprint arXiv:1802.08770*, 2018.

Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5852–5861, 2018.

Guodong Zhang, Lala Li, Zachary Nado, James Martens, Sushant Sachdeva, George E Dahl, Christopher J Shallue, and Roger Grosse. Which algorithmic choices matter at which batch sizes? insights from a noisy quadratic model. In *Advances in neural information processing systems*, 2019.

Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma. The Anisotropic Noise in Stochastic Gradient Descent: Its Behavior of Escaping from Minima and Regularization Effects. *ArXiv e-prints*, 2018.