G-TRACER: Expected Sharpness Optimization

Anonymous authors
Paper under double-blind review

Abstract

We propose a new regularization scheme for the optimization of deep learning architectures, G-TRACER ("Geometric TRACE Ratio"), which promotes generalization by seeking minima with low mean curvature, and which has a sound theoretical basis as an approximation to a natural gradient-descent based optimization of a generalized Bayes objective. By augmenting the loss function with a G-TRACER penalty, which can be interpreted as the metric trace of the Hessian (the Laplace-Beltrami operator) with respect to the Fisher information metric, curvature-regularized optimizers (e.g. SGD-TRACER and Adam-TRACER) are simple to implement as modifications to existing optimizers and don't require extensive tuning. We show that the method can be interpreted as penalizing, in the neighborhood a minimum, the difference between the mean value of the loss and the value at the minimum, in a way that adjusts for the natural geometry of the parameter space induced by the KL divergence. We show that the method converges to a neighborhood (depending on the regularization strength) of a local minimum of the unregularized objective, and demonstrate promising performance on a number of benchmark computer vision and NLP datasets, with a particular focus on challenging problems characterized by low a signal-to-noise ratio, or an absence of natural data augmentations and other regularization schemes.

1 Introduction

Contemporary neural network architectures (e.g. Llama 2: 70B parameters, GPT-4: 1.7T parameters) are typically overparameterized, with more parameters than constraints (Liu et al., 2022). The fact that interpolating solutions with no explicit regularization can generalize well to unseen data (Zhang et al., 2016) (Belkin et al., 2019) is surprising from a classical statistical learning perspective, and there is an emerging consensus that this phenomenon is due to implicit regularization in which, in very high dimensional settings, among all interpolating solutions, well-behaved minimum-norm solutions are preferred[1] (Curth et al., 2023). Whether implicit regularization alone suffices is problem-dependent, and is influenced by, among many other factors, the signal-to-noise ratio (Hastie et al., 2022). In practical settings, non-zero weight decay is typically applied, and explicit regularization is key to obtaining SOTA performance.[2]

Deep neural networks possess discrete and continuous symmetries (transformations which leave the underlying function invariant (Kristiadi et al., 2023)) as well as reparameterization invariance with respect to many common coordinate changes (e.g. BatchNorm (Ioffe & Szegedy, 2015), WeightNorm (Salimans & Kingma, 2016)). There is a large literature relating these characteristics of deep neural networks to the geometry of the loss surface[3] (Liu et al., 2022). Li et al. (2018) show, under mild assumptions, that sufficiently wide networks have no set-wise local minima that aren't global minima. Moreover, solutions to overparameterized neural networks typically form a high-dimensional manifold (Cooper, 2018) and are characterized by degenerate Hessians (Sagun et al., 2017), where the bulk of the eigenvalue spectrum is clustered around 0.

The connection between the geometry of the loss surface and generalization has long been the subject of interest and speculation, dating back to the MDL-based arguments of Hinton & van Camp (1993) and

---

[1] Indeed this effect can be observed even in certain linear models in the overparameterized regime $p > n$ (Hastie et al., 2022)

[2] Weight decay is equivalent ridge regularization, and it can be shown that minimum l2-norm regression is a limiting case of ridge regression as the ridge penalty goes to 0.

[3] Viewing the loss, for example, as a hypersurface: $\{(\boldsymbol{w}, L(\boldsymbol{w})), \boldsymbol{w} \in \mathbb{R}^{p+1} : \boldsymbol{w} \in \Theta\}$

Hochreiter & Schmidhuber (1997). In particular, the connection between sharpness and generalization is an intuitively appealing one, in that the sharp local minima of the highly nonlinear, non-convex optimization problems associated with modern large-scale deep learning architectures are more likely to be brittle and sensitive to perturbations in the parameters and training data, and thus lead to worse performance on unseen data. The recent success of the SAM algorithm, which measures sharpness as $\max_{\|\boldsymbol{\Delta w}\|_2 \leq \epsilon} L(\boldsymbol{w}+\boldsymbol{\Delta w})-L(\boldsymbol{w})$ (Foret et al., 2020) has reignited interest in geometrically-motivated regularization schemes. We propose a novel regularization scheme, which implicitly measures Sharpness as $\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H})$, where $\boldsymbol{H}$ is the Hessian of the loss and $\boldsymbol{G}$ can be interpreted as a Fisher Information Matrix, and show that the resulting scheme penalizes curvature in a principled geometric (intrinsic and approximately coordinate-free) way, admits an interpretation as a kernel smoothing of the loss surface, and performs competitively on benchmark vision and NLP datasets.

1.1 Problem setting

Our problem setting is as follows: we are given a dataset $\mathcal{D} = \{(\boldsymbol{x_i}, \boldsymbol{y_i})_{i=1}^n\}$ consisting of n independent input variables $\boldsymbol{x_i} \in \mathbb{R}^{d_x}$ with distribution $p(\boldsymbol{x})$, and corresponding targets (or labels) $\boldsymbol{y_i} \in \mathbb{R}^{d_y}$ with distribution $p(\boldsymbol{y}|\boldsymbol{x})$ and treat the parameters $\boldsymbol{w} \in \Theta \subseteq \mathbb{R}^p$ of a deep neural network (DNN) $f(\cdot, \boldsymbol{w}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ as a random variable. Given a loss function $l(\boldsymbol{y_i}, f(\boldsymbol{x_i}, \boldsymbol{w}))$, our goal is to find a $\boldsymbol{w}^*$ that minimizes the expected loss: $\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[l(\boldsymbol{y}, f(\boldsymbol{x}, \boldsymbol{w}))]$. Writing the finite-sample version of this expected loss as $L(\boldsymbol{w}) = \sum_{i=1}^n l(\boldsymbol{y_i}, f(\boldsymbol{x_i}, \boldsymbol{w}))$, we can form a generalized posterior distribution (Bissiri et al., 2016) $p(\boldsymbol{w}|\mathcal{D}) = p(\boldsymbol{w})\frac{1}{Z}\exp\{-L(\boldsymbol{w})\}$ (with normalizer Z and prior $p(\boldsymbol{w})$) over the weights, which coincides with the Bayesian posterior in the special case that the loss is the negative log-likelihood $L(\boldsymbol{w}) = -\frac{1}{n}\sum_{i=1}^n \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})$.

1.2 What is sharpness and why does it hurt generalization?

We first examine various notions of sharpness in the literature and then introduce the sharpness measure used in this paper.

1.2.1 The sharpness puzzle

Much of the literature on loss surface flatness and generalization has been concerned with non-geometric measures of sharpness and flatness, which is to say that they depend on the particular choice of coordinate system. Notable examples include Keskar et al. (2016), who define ϵ-sharpness as the maximum relative change in loss over a Euclidean norm-ball:

$$\max_{\|\boldsymbol{\Delta w}\|_2 \leq \epsilon} \frac{L(\boldsymbol{w}+\boldsymbol{\Delta w})-L(\boldsymbol{w})}{1+L(\boldsymbol{w})} \tag{1}$$

and the SAM algorithm (Foret et al., 2020), which uses a similar notion:

$$\max_{\|\boldsymbol{\Delta w}\|_2 \leq \epsilon} L(\boldsymbol{w}+\boldsymbol{\Delta w})-L(\boldsymbol{w}) \tag{2}$$

Since, at such a local minimum of the loss, for a perturbation $\boldsymbol{\Delta w}$, we have:

$$L(\boldsymbol{w}+\boldsymbol{\Delta w})-L(\boldsymbol{w}) = \boldsymbol{\Delta w}^T \boldsymbol{\nabla^2} L(\boldsymbol{w}) \boldsymbol{\Delta w} + O(\|\boldsymbol{\Delta w}\|^3) \tag{3}$$

both these measures are essentially equivalent to the spectral norm of the Hessian[4], which is not an invariant quantity. In particular, at any critical point which is a minimum with non-zero Hessian, there exists a reparameterization that leaves the underlying function of the data unchanged and which makes the spectral norm arbitrarily large (Dinh et al., 2017). More generally, there has been an extensive literature (Hochreiter & Schmidhuber, 1997) (Hinton & van Camp, 1993) attempting to characterize the loss-surface Hessian $\boldsymbol{\nabla^2} L(\boldsymbol{w})$ and to relate these characteristics to generalization. In many practically relevant cases, multiple minima are associated with zero (or close to zero) training error, and explicit or implicit regularization is needed to find solutions with the best generalization error.

---

[4] To second order, $\max_{\|\boldsymbol{\Delta w}\|_2 \leq \epsilon} \frac{L(\boldsymbol{w}+\boldsymbol{\Delta w})-L(\boldsymbol{w})}{1+L(\boldsymbol{w})} \approx \epsilon \frac{||\boldsymbol{\nabla^2} L(\boldsymbol{w})||_2}{2(1+L(\boldsymbol{w}))}$

Wei & Schwab (2020) show that given a degenerate valley in the loss surface, stochastic gradient descent (SGD) on average decreases the trace of the Hessian, which is suggestive of a connection between locally flat minima, overparameterization, and generalization. The parallel works of Sagun et al. (2017) and Chaudhari et al. (2016) examine the spectrum of the loss-function Hessian to characterize the landscape of the loss before and after optimization, and find that overparameterization is associated with the bulk of the Hessian spectrum lying close to zero and thus to highly degenerate minima. Observing that the clustering of eigenvalues around 0 corresponds to wide valleys in the loss surface, Chaudhari et al. (2016) propose an algorithm, Entropy-SGD, which has parallels with this work, and which explicitly introduces an "entropic term" which explicitly captures the width of the valley in the objective, resulting in a modified objective (to be maximized) (Dziugaite & Roy, 2017):

$$\log \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})}[\mathrm{e}^{-\tau L(\boldsymbol{w}+\boldsymbol{\epsilon})}] \tag{4}$$

Thus, the loss landscape is smoothed by applying a Gaussian convolution and the resulting minima are expected to be less sharp.

Although these arguments have intuitive appeal, these notions of flatness are dependent on arbitrary choices of parameterization, which can in general be used to arbitrarily change the flatness of the loss surface without changing the underlying function of the input data (Dinh et al., 2017).

1.2.2 Our approach to sharpness

In this work we will implicitly measure sharpness as $\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H})$ where $\boldsymbol{H}$ is the Hessian, and $\boldsymbol{G}$ can be interpreted as a Fisher Information Matrix, which, as a Riemannian metric tensor, defines a metric on the parameter manifold. At a critical point, $\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H})$ is the Laplace-Beltrami operator[5] which generalizes the Laplacian to Riemannian manifolds (Lee, 2019) (Kristiadi et al., 2023), and defines an invariant, geometric quantity which, by analogy with $\mathrm{Tr}(\boldsymbol{H})$ in Euclidean space, measures the average deviation from flatness, adjusting for the curvature of the manifold. Crucially, this notion is not an assumption, but rather emerges naturally from the variational optimization of a generalized Bayes objective using the KL-metric. In particular, for a multivariate Gaussian variational approximation, the trace penalty corresponds to a smoothing of the loss surface using a kernel estimated online.

1.3 Penalizing sharpness: Sharpness-Aware Minimization

We first give an overview of SAM and its derivation, then highlight its strengths and weaknesses, and the relevant recent literature.

1.3.1 SAM overview

Despite the intuitive appeal and plausible justifications for flat solutions to be a goal of DNN optimization algorithms, there have been few practical unqualified successes in exploiting this connection to improve generalization performance. A notable exception is a recent algorithm, Sharpness Aware Minimization (SAM) (Foret et al., 2020), which seeks to improve generalization by optimizing a saddle-point problem of the form:

$$\min_{\boldsymbol{w}} \max_{\|\boldsymbol{\Delta w}\| \leq \rho} L(\boldsymbol{w} + \boldsymbol{\Delta w}) \tag{5}$$

An approximate solution to this problem is obtained by differentiating through the inner maximization, so that, given an approximate solution $\boldsymbol{\Delta w}^* := \rho \frac{\boldsymbol{\nabla} L(\boldsymbol{w}^k)}{\|\boldsymbol{\nabla} L(\boldsymbol{w}^k)\|_2}$ to the inner maximization (dual norm) problem:

$$\arg \max_{\|\boldsymbol{\Delta w}\| \leq \rho} L(\boldsymbol{w} + \boldsymbol{\Delta w}) \tag{6}$$

the gradient of the SAM objective is approximated as follows:

$$\boldsymbol{\nabla}_{\boldsymbol{w}} \left(\max_{\|\boldsymbol{\Delta w}\|_{FR} \leq \epsilon} L(\boldsymbol{w} + \boldsymbol{\Delta w}) \right) \approx \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w} + \boldsymbol{\Delta w}^*) \approx \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}+\boldsymbol{\Delta w}^*} \tag{7}$$

---

[5] Also known as the manifold Laplacian

While the method has gained widespread attention, and state-of-the-art performance has been demonstrated on several benchmark datasets, it remains relatively poorly understood, and the motivation and connection to sharpness is questionable given that the Euclidean norm-ball isn't invariant to changes in coordinates. Given a 1-1 mapping $g : \Theta' \to \Theta$ we can reparameterize our DNN $f(\cdot, \boldsymbol{w})$ using the "pullback" $g^*(f)(\cdot, \boldsymbol{\nu}) := f(\cdot, g(\boldsymbol{\nu}))$ under which, crucially, the underlying prediction function $f(\cdot, \boldsymbol{w}) : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ (and therefore the loss) itself is invariant, since, for $\boldsymbol{\nu} = g^{-1}(\boldsymbol{w})$, we have $f(\cdot, \boldsymbol{w}) = f(\cdot, g(\boldsymbol{\nu}))$. Under this coordinate transformation, however, the Hessian at a critical point transforms as (Dinh et al., 2017):

$$\boldsymbol{\nabla^2} L(\boldsymbol{\nu}) = \boldsymbol{\nabla} g(\boldsymbol{\nu})^T \boldsymbol{\nabla^2} L \boldsymbol{\nabla} g(\boldsymbol{\nu}) \tag{8}$$

In particular, Dinh et al. (2017) explicitly show, using layer-wise transformations $T_\alpha : (\boldsymbol{w}_1, \boldsymbol{w}_2) \to (\alpha \boldsymbol{w}_1, \alpha^{-1} \boldsymbol{w}_2)$, that deep rectifier feedforward networks possess large numbers of symmetries which can be exploited to control sharpness without changing the network output. The existence of these symmetries in the loss function, under which the geometry of the local loss can be substantially modified (and in particular, the spectral norm and trace of the Hessian) means that the relationship between the local flatness of the loss landscape and generalization is a subtle one.

It is instructive to consider the PAC Bayes generalization bound that motivates SAM, the derivation of which starts from a PAC-Bayesian generalization bound (McAllester, 1999; Dziugaite & Roy, 2017):

Theorem 1. *For any distribution $\mathcal{D}$ and prior p over the parameters $\boldsymbol{w}$, with probability $1 - \delta$ over the choice of the training set $\mathcal{S} \sim \mathcal{D}$, and for any posterior q over the parameters:*

$$\mathbb{E}_q[L_{\mathcal{D}}(\boldsymbol{w})] \leq \mathbb{E}_q[L_{\mathcal{S}}(\boldsymbol{w})] + \sqrt{\frac{\mathbb{D}_{KL}[q, p] + \log \frac{n}{\delta}}{2(n-1)}} \tag{9}$$

where the KL divergence:

$$\mathbb{D}_{KL}[q, p] = \mathbb{E}_{p(\boldsymbol{w})} \left[\log \left(\frac{q(\boldsymbol{w})}{p(\boldsymbol{w})} \right) \right] \tag{10}$$

defines a statistical distance $\mathbb{D}_{KL}[q, p]$ (though not a metric, as it's symmetric only to second order) on the space of probability distributions. Assuming an isotropic prior $p = N(\boldsymbol{0}, \sigma_p^2 \boldsymbol{I})$ for some σ_p, an isotropic posterior $q = N(\boldsymbol{w}, \sigma_q^2 \boldsymbol{I})$, so that $\mathbb{E}_q[L_{\mathcal{D}}(\boldsymbol{w})] = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_q^2 \boldsymbol{I})}[L_{\mathcal{D}}(\boldsymbol{w} + \boldsymbol{\epsilon})]$, applying the covering approach of Langford & Caruana (2001) to select the best (closest to q in the sense of KL divergence) from a set of pre-defined data-independent prior distributions satisfying the PAC generalization bound, Foret et al. (2020) show that the bound in Theorem 1 can be written in the following form:

$$\mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_q^2 \boldsymbol{I})}[L_{\mathcal{D}}(\boldsymbol{w} + \boldsymbol{\epsilon})] \leq \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_q^2 \boldsymbol{I})}[L_{\mathcal{S}}(\boldsymbol{w} + \boldsymbol{\epsilon})] + g\left(\frac{\|\boldsymbol{w}\|_2^2)}{\rho^2} \right) \tag{11}$$

(for a monotone function g). Then, crucially, one may apply a well-known tail-bound for a chi-square random variable to bound $\|\boldsymbol{\epsilon}\|_2$, thus bounding the expectation over q (with probability $1 - 1/\sqrt{n}$) by the maximum value over a Euclidean norm-ball ball. This provides the following generalization bound:

Theorem 2. *For any $\rho > 0$ and any distribution $\mathcal{D}$, with probability $1 - \delta$ over the choice of the training set $\mathcal{S} \sim \mathcal{D}$,*

$$L_{\mathcal{D}}(\boldsymbol{w}) \leq \max_{\|\boldsymbol{\epsilon}\|_2 \leq \rho} L_{\mathcal{S}}(\boldsymbol{w} + \boldsymbol{\epsilon}) + g\left(\frac{\|\boldsymbol{w}\|_2^2)}{\rho^2} \right) \tag{12}$$

where $\rho = \sigma\sqrt{k}\left(1 + \sqrt{\frac{\ln(n)}{k}}\right)$, $n = |\mathcal{S}|$, and k is the number of parameters.

This bound justifies and motivates the SAM objective:

$$\max_{\|\boldsymbol{\Delta w}\| \leq \rho} L(\boldsymbol{w} + \boldsymbol{\Delta w}) + \lambda \|\boldsymbol{w}\|_2^2 \tag{13}$$

and resulting algorithm. Although the bound in Theorem 2 suggests that the ridge penalty should vary with the radius of the perturbation, in practice (Foret et al., 2020) the penalty term is fixed (or simply set to zero) even when different perturbation radii are searched over. Subsequent refinements of SAM (Kim et al., 2022b) ignore the ridge penalty term altogether, and the choice of an optimal perturbation radius is what drives the success of the method. It is not clear, however, why this adversarial parameter-space perturbation should help generalization more than evaluating (and approximating) the expectation in the very bound which motivates the SAM procedure in the first place, which would lead instead to an objective (ignoring, for now, the ridge penalty term) of the following form:

$$\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})}[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})] \tag{14}$$

Moreover, the worst-case adversarial perturbation used by SAM is likely to be noisier and is also naturally a significantly looser bound than the expectation-based bound.

1.3.2 SAM Strengths and weakness and related literature

SAM has shown great promise in some applications, particularly in its robustness to noise, where the performance gains are sometimes dramatic (Baek et al., 2024) (Foret et al., 2020). The method has also reinvigorated research on flatness-promoting regularizations. There are, however, numerous weaknesses and open questions, some of which have been addressed in the literature.

1. The Euclidean norm-ball based bound is not invariant to coordinate transformations, so that scale changes (such as occur, for example, when applying batch-normalization or weight-normalization) which have no effect on the output of the learned probability distribution, can nevertheless still result in arbitrary changes to the penalty. More generally, any geometric notion of loss surface flatness must be independent of arbitrary rescaling of the network parameters.
2. SAM performs poorly for large batch-sizes and the practical benefits of SAM are typically only seen for very small batch sizes (even though there is nothing in the theory or deviation to suggest this) (Andriushchenko & Flammarion, 2022).
3. SAM optimizes a loose upper-bound on an expectation in the generalization bound that motivates the method.

Several attempts have been made to address some of these issues. Kwon et al. (2021) focus on the inner maximization problem and propose an ad-hoc linear node-wise rescaling to mitigate the scale dependence of the method. Kim et al. (2022a) address the Euclidean norm-ball limitation by preconditioning the inner gradient step with an empirical inverse diagonal Fisher information matrix and demonstrate modest improvements over SAM on CIFAR-10 and CIFAR-100 datasets. Möllenhoff & Khan (2022) make the connection between SAM and Bayesian methods and show that SAM can be derived as an optimal relaxation of the Bayes objective, also demonstrating increased accuracy by improving variance estimates and using those in the inner gradient step.

2 G-TRACER

Motivated by these considerations, our starting point is the optimization of the following generalized variational objective (Knoblauch et al., 2019) (Bissiri et al., 2016) (related to, but more general than the objective from which SAM is derived) over the space of probability measures $\mathcal{P}(\Theta)$ on the parameter space Θ:

$$q^*(\boldsymbol{w}) = \arg\min_{q\in\mathcal{P}(\Theta)}\left\{\mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \rho\mathbb{D}_{KL}[q,p]\right\} \tag{15}$$

where ρ is a positive-valued parameter which controls the strength of regularization. By assuming that the posterior $q(\boldsymbol{w})$ and prior $p(\boldsymbol{w})$ belong to multivariate Gaussian parametric families, we derive a general purpose regularization scheme which, at a high level, consists of the following simple steps:

1. Augment the loss function $L(\boldsymbol{w})$ with a G-TRACER ("Geometric TRACE Regularizer") of the form $\rho\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H}^{\dagger})$, where $\boldsymbol{H}^{\dagger}$ is a positive definite approximation to the Hessian $\boldsymbol{\nabla}^2_{\boldsymbol{w}}L(\boldsymbol{w})$.

2. Estimate $\boldsymbol{G}$ online using an exponential smoothing of $\boldsymbol{H}^\dagger$

The effect is to modify the gradient from a pure descent direction $\boldsymbol{\nabla_w} L(\boldsymbol{w})$ by a direction given by $\rho \boldsymbol{\nabla_w} \mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H}^\dagger)$, which encourages a reduction in the preconditioned Hessian trace, in a way that, at critical points, doesn't depend on the parameterization of f, and, more generally, is invariant to affine coordinate changes, such as the layer-wise scale transformations $T_\alpha : (\boldsymbol{w_1}, \boldsymbol{w_2}) \to (\alpha \boldsymbol{w_1}, \alpha^{-1}\boldsymbol{w_2})$ of Dinh et al. (2017). The preconditioner[6] $\boldsymbol{G}^{-1}$ can be viewed as an approximate inverse metric tensor that captures the geometry of the variational parameter space and the penalty term can be interpreted, in the neighbourhood of a critical point, as an approximate metric trace of the Hessian, or Laplace-Beltrami operator, which measures the difference between the mean value of the loss over a geodesic ball (Lee, 2019) and the value at a minimum, and which thus encourages flatness.

Starting with SGD as the base optimizer (for example), using stochastic gradients $\boldsymbol{\nabla_w} L_{\mathcal{B}}(\boldsymbol{w})$ computed on minibatches $\mathcal{B}$ of the data, and choosing as a positive definite approximation to the Hessian the log-likelihood Fisher Information Matrix:[7]

$$\boldsymbol{F_w} = \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}), \boldsymbol{x} \sim p(\boldsymbol{x})}[-\boldsymbol{\nabla}^2_{\boldsymbol{w}} \log p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})] \tag{16}$$

which is the expectation, under the model's output distribution, of the log-likelihood Hessian, leads to the following general update equations:

$$\boxed{\begin{aligned} \boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \boldsymbol{\nabla_w}[L_{\mathcal{B}}(\boldsymbol{w}) + \rho \mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{F_w})] \\ \boldsymbol{G} \leftarrow (1-\beta)\boldsymbol{G} + \beta \boldsymbol{F_w} \end{aligned}} \tag{17}$$

Next, we show how penalizing the objective with a G-TRACER penalizes sharpness, how to add a simple G-TRACER in practice, and how the penalty can be derived from the generalized variational objective (15).

2.1 How does G-Tracer penalize sharpness?

We first examine, in a simplified setting, the interplay between the penalty parameter ρ and the variance of the perturbation over which the loss is smoothed by convolution with a Gaussian kernel. We then show that, when the expectation (or convolution) is approximated to second order, the result has a direct correspondence with the Laplace-Beltrami operator, which establishes a rigorous link to flatness.

2.1.1 ρ determines generalized variance of the Gaussian kernel smoothing

Ignoring for simplicity the contribution from the prior term (which would correspond to a ridge-regularization term under the assumption $p(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{0}, \sigma_p \boldsymbol{I})$), leads to following objective, which we seek to minimize over $\boldsymbol{w}$:

$$\mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] - \rho \mathcal{H}(q) \tag{18}$$

where $\mathcal{H}(q) = \mathbb{E}_{q(\boldsymbol{w})}[-\log q(\boldsymbol{w})]$ is the entropy of q. For the choice $q(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{w}, \sigma^2 \boldsymbol{I})$, the optimization problem associated with the variational objective becomes (absorbing some constants into ρ):

$$\arg\min_q \mathbb{E}_q[L(\boldsymbol{w})] - \rho \mathcal{H}(q) = \arg\min_{\boldsymbol{w}, \sigma^2} \mathbb{E}_q[L(\boldsymbol{w})] + \rho \log \frac{1}{\sigma^2} \tag{19}$$

so that we can see that ρ determines the variance of Gaussian perturbation over which the loss is averaged. More generally, choosing $q \sim \mathcal{N}(\boldsymbol{w}, \boldsymbol{\Sigma})$ leads to the following variational objective:

$$\arg\min_q \mathbb{E}_q[L(\boldsymbol{w})] - \rho \mathcal{H}(q) = \arg\min_{\boldsymbol{w}, \boldsymbol{\Sigma}} \mathbb{E}_q[L(\boldsymbol{w})] + \rho \log \frac{1}{\det(\boldsymbol{\Sigma})} \tag{20}$$

so that large values of ρ will correspond to distributions with larger volume, since for $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$, $\boldsymbol{x}$ lies within the ellipsoid $\boldsymbol{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{x} = \chi^2(\alpha)$ with probability $1-\alpha$, with the volume of the ellipsoid proportional to $\det(\boldsymbol{\Sigma})^{\frac{1}{2}}$ (Anderson, 2003). The regularization parameter ρ thus controls the generalized variance $\det(\boldsymbol{\Sigma})$ of the Gaussian kernel which is used to smooth the loss when calculating the expectation $\mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})]$.

---

[6] Note that $\boldsymbol{G}^{-1}$ is a constant in the update equation for $\boldsymbol{w}$, whereas $\boldsymbol{F} \equiv \boldsymbol{F}(\boldsymbol{w})$

[7] We will see that this choice is a natural one in section 2.3

2.1.2 $\mathrm{Tr}(\mathbf{G}^{-1}\mathbf{H})$ regularization is curvature regularization and thus promotes flatness

We first examine curvature regularization from the perspective of the variational parameter space. The geometric trace penalty arises by modeling the posterior over the parameters as a multivariate Gaussian: $q(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and then performing natural gradient descent on the variational objective (39) and forming the following second-order approximation:

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathbb{E}_q[L(\boldsymbol{w})] \approx \boldsymbol{\nabla}_{\boldsymbol{\mu}}[L(\boldsymbol{\mu}) + \frac{1}{2}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{H})] \tag{21}$$

The Fisher Information Matrix $\boldsymbol{F}_{\boldsymbol{\mu}}$ of q, viewed as a function of the variational mean parameter[8] $\boldsymbol{\mu}$, can be computed exactly, and is given by:

$$\boldsymbol{F}_{\boldsymbol{\mu}} = \mathbb{E}_q\left[-\boldsymbol{\nabla}^2_{\boldsymbol{\mu}} \log q\right] = \boldsymbol{\Sigma}^{-1} \tag{22}$$

(see Appendix A.1.2 for details). We thus have:

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathbb{E}_q[L(\boldsymbol{w})] \approx \boldsymbol{\nabla}_{\boldsymbol{\mu}}[L(\boldsymbol{\mu}) + \frac{1}{2}\mathrm{Tr}(\boldsymbol{F}_{\boldsymbol{\mu}}^{-1}\boldsymbol{H})] \tag{23}$$

Comparing with the G-TRACER penalty term $\rho\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{F}_{\boldsymbol{w}})$ we see that $\boldsymbol{G}^{-1} \propto \boldsymbol{F}_{\boldsymbol{\mu}}^{-1}$. Thus $\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{F}_{\boldsymbol{w}}) \propto \mathrm{Tr}(\boldsymbol{F}_{\boldsymbol{\mu}}^{-1}\boldsymbol{F}_{\boldsymbol{w}}) \approx \mathrm{Tr}(\boldsymbol{F}_{\boldsymbol{\mu}}^{-1}\boldsymbol{H}) = \mathrm{Tr}_{\boldsymbol{F}_{\boldsymbol{\mu}}}(\boldsymbol{H})$, where $\mathrm{Tr}_{\boldsymbol{F}_{\boldsymbol{\mu}}}$ is the metric trace[9] (Lee, 2019) of the Hessian, and which, at a critical point, is exactly the Laplacian $\boldsymbol{\Delta}$ (also known as the Laplace-Beltrami operator (Lee, 2019) Kristiadi et al. (2023)), a fundamental operator in differential geometry and analysis, which measures curvature in an intrinsic, coordinate independent way, correcting for the underlying geometry of the manifold[10]. In the neighborhood of a local minimum $\boldsymbol{\mu}^*$ of L (in particular), it can be interpreted as the difference between the mean value of L over a (geodesic) ball centered at $\boldsymbol{\mu}^*$ and $L(\boldsymbol{\mu}^*)$, due to the following mean-value property for smooth functions over geodesic balls $B_r(\boldsymbol{\mu})$ (Gray & Willmore, 1982) (Loustau, 2015):

$$\frac{1}{\mathrm{vol}(B_r(\boldsymbol{\mu}^*))}\int_{B_r(\boldsymbol{\mu}^*)} L(\boldsymbol{\mu})\boldsymbol{dV} - L(\boldsymbol{\mu}^*) = \frac{\boldsymbol{\Delta}L(\boldsymbol{\mu}^*)}{2(n+2)}r^2 + \mathcal{O}(r^4) \tag{24}$$

so that we have the following asymptotic expression for the value of the Laplacian at $\boldsymbol{\mu}^*$:

$$\boldsymbol{\Delta}L(\boldsymbol{\mu}^*) = \lim_{r\to 0}\frac{2(n+2)}{r^2}\frac{1}{\mathrm{vol}(B_r(\boldsymbol{\mu}^*))}\int_{B_r(\boldsymbol{\mu}^*)} L(\boldsymbol{\mu}) - u(\boldsymbol{\mu}^*)\boldsymbol{dV} \tag{25}$$

Since, at a minimum $\boldsymbol{\mu}^*$ of $L(\boldsymbol{\mu})$, $\mathrm{Tr}(\boldsymbol{F}_{\boldsymbol{\mu}^*}^{-1}\boldsymbol{H}) = \boldsymbol{\Delta}L(\boldsymbol{\mu}^*) \geq 0$, penalizing $\mathrm{Tr}(\boldsymbol{F}_{\boldsymbol{\mu}}^{-1}\boldsymbol{H})$ has the effect of forcing the values of $L(\boldsymbol{\mu})$ in a neighborhood of a minimum to be closer to the value at the minimum.

From the perspective of the parameter space Θ (the weight space), the final general form of the update equations (equation 17) consists of the G-TRACER penalty $\rho\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{F}_{\boldsymbol{w}})$ (which is affine invariant, as well as invariant to all diffeomorphic coordinate transformations at critical points), where $\boldsymbol{G}$ is an exponentially smoothed (over minibatches) estimate of the log-likelihood FIM. $\boldsymbol{G}$ captures the local geometry of the loss surface $L(\boldsymbol{w})$ and, in the neighborhood of critical points, the G-TRACER penalty, as a metric trace w.r.t. $\boldsymbol{G}$, can be interpreted as a measure of sharpness on the weight space.

This formulation therefore highlights the deep connection between:

- Intrinsic curvature as measured by the manifold Laplacian, both in the space of variational parameters, and in the parameter space of the weights Θ
- Convolving the loss with a multivariate Gaussian kernel which captures the local geometry in the neighbourhood of $\boldsymbol{\mu}$

---

[8] $\boldsymbol{\Sigma}$ is a constant in the update equation for $\boldsymbol{\mu}$

[9] The metric induces canonical or musical isomorphisms $\sharp$ and $\flat$ between the tangent and cotangent bundles. The metric trace of a symmetric 2-tensor $\boldsymbol{H}$ is $\mathrm{Tr}_{\boldsymbol{F}_{\boldsymbol{\mu}}}\boldsymbol{H} = \mathrm{Tr}\boldsymbol{H}^{\sharp}$

[10] An alternative viewpoint, complementary to the one we take here, is an extrinsic one, where we consider the loss surface in graph coordinates as a hypersurface embedded in ambient Euclidean space $\mathbb{R}^{p+1}$: $\{(\boldsymbol{w}, L(\boldsymbol{w})), \boldsymbol{w} \in \mathbb{R}^{p+1} : \boldsymbol{w} \in \Theta\}$. In Euclidean space, the Hessian is the matrix of the shape operator (or the second fundamental form (Lee, 2019)). The eigenvalues of the Hessian correspond to the principal curvatures, and the mean curvature is the mean of the principal curvatures (or equivalently, the Hessian trace). The Euclidean metric on $\mathbb{R}^{p+1}$ then induces a pullback metric on the embedded submanifold, $\boldsymbol{G}$, and at a critical point, the matrix of the shape operator in a Riemannian manifold is given by $\boldsymbol{G}^{-1}\boldsymbol{H}$ with corresponding mean curvature $\mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{H})$ (Kristiadi et al., 2023).

Algorithm 1 SGD-TRACER

```
Require: $\alpha_t$: Stepsize
Require: $\beta$: Exponential smoothing constant for the online Fisher estimate
Require: $\rho$ : flatness inducing penalty term
Require: $\delta$: small positive constant
  Initialize $\mathbf{w}_0$, $\mathbf{f}_0$, $t = 0$
  while not converged do
    Sample batch $\mathcal{B} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), ...(\boldsymbol{x}_b, \boldsymbol{y}_b)\}$
    $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t \boldsymbol{\nabla}_{\mathbf{w}} \left[ L_{\mathcal{B}}(\mathbf{w}_t) + \rho \left(\boldsymbol{\nabla}_{\mathbf{w}} L_{\mathcal{B}}(\mathbf{w}_t)\right)^2 / (\mathbf{f}_t + \delta) \right]$
    $\mathbf{f}_{t+1} = (1-\beta)\mathbf{f}_t + \beta \left(\boldsymbol{\nabla}_w L_{\mathcal{B}}(\mathbf{w}_t)\right)^2$
  end while
```

2.2 How is G-Tracer evaluated in practice?

The inverse of the diagonal empirical Fisher, as an approximation to the inverse Fisher Information Matrix, is used as a gradient preconditioner by the Adam (Kingma & Ba, 2014) and Adagrad (Duchi et al., 2011) optimizers. While many other approaches are possible here, of varying degrees of sophistication and complexity (see KFAC (Martens & Grosse, 2015) for example), we find that this simple, cheap, and scalable approach works extremely well. This regularization scheme can in principle be combined with any optimizer by simply augmenting the loss, and maintaining an estimate of the smoothed squared gradients (as is done by Adam). Modifying SGD to use a G-TRACER with the diagonal empirical Fisher yields algorithm 1. In our experiments, we freely add momentum to this simple SGD formulation. In experiments with transformer architectures, we modify Adam in the same way, by simply adding a G-TRACER penalty to the objective and reusing the squared gradients already needed for the Adam update.

2.3 Derivation sketch

We first sketch the derivation of the general update equations (17) and then show how these lead to SGD-TRACER (1).

2.3.1 General update equations

Since we are optimizing in the space of probability distributions whose parameterizations can be changed without changing the underlying probability distribution, it is natural to perform gradient descent on our variational objective using Riemannian gradients corresponding to the Fisher-Rao metric[11] (also known as natural gradient descent (Amari, 1998)). We assume that the posterior has the form $q(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{w}, \boldsymbol{\Sigma})$, so that our optimization problem becomes:

$$\arg\min_{\boldsymbol{\mu}, \boldsymbol{\Sigma}} \mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \rho \mathbb{D}_{KL}[q, p] \tag{26}$$

where ρ is a positive real-valued parameter.

We show in Appendix A.1.2 that, assuming an isotropic Gaussian prior, $p(\boldsymbol{w}) \sim \mathcal{N}(\mathbf{0}, \eta \boldsymbol{I})$, performing gradient descent w.r.t. the natural gradient leads to the following iterative update equations for the variational parameters(Khan & Rue, 2021) (Zhang et al., 2017):

$$\begin{aligned} \boldsymbol{\mu} &\leftarrow \boldsymbol{\mu} - \alpha \boldsymbol{\Lambda}^{\mathbf{-1}} \left(\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})] + \frac{\rho}{\eta} \boldsymbol{w} \right) \\ \boldsymbol{\Lambda} &\leftarrow (1-\beta)\boldsymbol{\Lambda} + \beta \left(\frac{\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})]}{\rho} + \eta^{-1} \boldsymbol{I} \right) \end{aligned} \tag{27}$$

where α and β are the learning rates for the mean and precision updates, respectively, and $\boldsymbol{\Lambda} := \boldsymbol{\Sigma}^{-1}$ is the precision matrix. Approximating the expectations to second order, approximating gradients using mini-

---

[11] This can be shown to be steepest descent in the KL-metric (Martens, 2020)

batches, and further simplifying leads to the following update equations for the parameter[12] (see Appendix A.1.2 for a detailed derivation):

$$\begin{aligned} \boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \left(\boldsymbol{G}^{-1} \boldsymbol{\nabla}_{\boldsymbol{w}} [L_{\mathcal{B}}(\boldsymbol{w}) + \rho \mathrm{Tr}(\boldsymbol{G}^{-1} \boldsymbol{H})] \right) \\ \boldsymbol{G} \leftarrow (1-\beta)\boldsymbol{G} + \beta \boldsymbol{H} \end{aligned} \tag{28}$$

where $\boldsymbol{H} = \boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})$ is the Hessian.

As we will see in the full derivation of the method in Appendix A.1.2, the recursion for $\boldsymbol{G}$ involving the loss Hessian is an update equation for the precision[13], which is exactly the Fisher Information Matrix $\boldsymbol{F}_{\boldsymbol{\mu}}$ of the variational distribution q viewed as a function of $\boldsymbol{\mu}$:

$$\boldsymbol{F}_{\boldsymbol{\mu}} = \mathbb{E}_{\boldsymbol{w}\sim q}[\boldsymbol{\nabla}_{\boldsymbol{\mu}} \log q(\boldsymbol{w};\boldsymbol{\mu})^T \boldsymbol{\nabla}_{\boldsymbol{\mu}} \log q(\boldsymbol{w};\boldsymbol{\mu})] = \mathbb{E}_{\boldsymbol{w}\sim q}[-\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\mu}} \log q(\boldsymbol{w};\boldsymbol{\mu})^T] \tag{29}$$

and which defines a Riemannian metric on $\boldsymbol{\mu}$.

The Hessian $\boldsymbol{H}$ in the update equations (28) arises from a second-order approximation to the loss $L(\boldsymbol{w})$. When the loss is given by the log-loss $l(\boldsymbol{y}, f(\boldsymbol{x},\boldsymbol{w})) = -\log p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$[14], the log-likelihood Fisher Information Matrix (FIM),

$$\boldsymbol{F}_{\boldsymbol{w}} = \mathbb{E}_{\boldsymbol{y}\sim p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}),\boldsymbol{x}\sim p(\boldsymbol{x})}[-\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} \log p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})] \tag{30}$$

as the expectation of the Hessian under the model's output distribution[15] (Amari, 1998), is known to provide a better local quadratic approximation to $L(\boldsymbol{w})$ than the Hessian H (Martens & Sutskever, 2012) when optimizing non-convex objectives. In particular:

$$M(\boldsymbol{\Delta}_{\boldsymbol{w}}) = \frac{1}{2} \boldsymbol{\Delta}_{\boldsymbol{w}}^T \boldsymbol{F}_{\boldsymbol{w}} \boldsymbol{\Delta}_{\boldsymbol{w}} + \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})^T \boldsymbol{\Delta}_{\boldsymbol{w}} + L(\boldsymbol{w}) \tag{31}$$

can be viewed as a convex approximation to the second-order Taylor series of $L(\boldsymbol{w} + \boldsymbol{\Delta}_{\boldsymbol{w}})$ for which the minimizer is the negative natural gradient $-\boldsymbol{F}_{\boldsymbol{w}}^{-1} \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})$ (Martens, 2020), and moreover, $\boldsymbol{F}_{\boldsymbol{w}}$ can be shown to converge to the Hessian in the limit[16], as the training error goes to zero (Kunstner et al., 2019).

Crucially, the log-likelihood FIM is also the Hessian of the KL divergence, which measures the dissimilarity between probability distributions in an intrinsic way, independently of parameterization. As an infinitesimal form of the KL divergence, it defines a metric tensor on the parameter manifold corresponding to the Fisher-Rao metric(Amari, 1998).

We therefore make the substitution $\boldsymbol{H} \leftrightarrow \boldsymbol{F}_{\boldsymbol{w}}$ and, in addition to the advantages of positive definiteness and strong empirical performance, this is a natural identification for the log-loss, where the log-likelihood FIM converges to the Hessian as the residual r goes to zero (Kunstner et al., 2019), since then we have the relations. Furthermore, this connects in a natural way the Riemannian metric on the space of variational parameters $\boldsymbol{F}_{\boldsymbol{\mu}}$ and the approximate Riemannian metric $\boldsymbol{F}_{\boldsymbol{w}}$ on the weight space Θ.

We show in Appendix A.1.2 that the penalty term $\rho \mathrm{Tr}(\boldsymbol{G}^{-1} \boldsymbol{F}_{\boldsymbol{w}})$ is invariant to affine coordinate transformations. Moreover, as $\boldsymbol{G}^{-1}$ is an approximation to the inverse metric on the parameter space Θ and $\boldsymbol{F}$ is an approximation to the Hessian (which is exact when the model fits the data and the residuals are zero (Kunstner et al., 2019)), the penalty can be interpreted, in the neighborhood of local minima, as a coordinate-independent measure of curvature which correctly accounts for the geometry of the underlying manifold $L(\boldsymbol{w})$.

Using a stochastic update based on minibatches, substituting-in the Empirical Fisher approximation of the Hessian (see Appendix A.1.2 for details, and a discussion of alternatives) and dropping the preconditioner (for

---

[12] Having approximated the expectations, we can identify $\boldsymbol{w}$ and $\boldsymbol{\mu}$.

[13] We explain later why this is a natural choice

[14] And for the most practically relevant losses (which are the ones we consider here): cross-entropy (classification), and squared error (regression), corresponding to exponential family output distributions with natural parameters given by the output function $f(\boldsymbol{x},\boldsymbol{w})$ (Kunstner et al., 2019) (Martens, 2020)

[15] Whereas H is the expected Hessian under the data distribution

[16] Assuming exponential family distributions

simplicity of exposition, and since our focus here is on the regularizer), we arrive at the modified SGD-type update (17):

$$\begin{aligned} \boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \left(\boldsymbol{\nabla_w}[L_{\mathcal{B}}(\boldsymbol{w}) + \rho \mathrm{Tr}(\boldsymbol{G}^{-1}\boldsymbol{F_w})]\right) \\ \boldsymbol{G} \leftarrow (1-\beta)\boldsymbol{G} + \beta \boldsymbol{F_w} \end{aligned} \tag{32}$$

2.3.2 SGD-TRACER

We now make two simplifications. First, we use a mean-field approximation, representing the FIM by its diagonal, as is done in Adam (Kingma & Ba, 2014) and Adagrad (Duchi et al., 2011), thus:

$$\boldsymbol{F_w} \approx \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})^2 \tag{33}$$

Secondly, as most current deep learning frameworks don't straightforwardly support access to per-example gradients, which can in principle be achieved with negligible additional cost (see, for example, BackPACK Dangel et al. (2020) second-order Pytorch extensions), for simplicity and efficiency, we use the gradient magnitude (GM) approximation (Bottou et al., 2016), as used in standard optimizers Adam and RMSprop, replacing the sum of squared gradients with the square of summed gradients:

$$\frac{1}{n}\sum_{i=1}^{n} [\boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})]^2 \approx \left[\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})\right]^2 \tag{34}$$

and we write the resulting FIM diagonal as $\boldsymbol{\nabla_w} L(\boldsymbol{w})^2$. Finally, it is standard practice to (Martens, 2020) to add Tikhonov regularization or damping via a small positive real constant δ when using 2nd-order optimization methods, so that we end up with Algorithm 1:

$$\begin{aligned} \mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t \boldsymbol{\nabla}_{\mathbf{w}} \left[L_{\mathcal{B}}(\mathbf{w}_t) + \rho \left(\boldsymbol{\nabla}_{\mathbf{w}} L_{\mathcal{B}}(\mathbf{w}_t)\right)^2 / (\mathbf{f}_t + \delta) \right] \\ \mathbf{f}_{t+1} = (1-\beta)\mathbf{f}_t + \beta \left(\boldsymbol{\nabla_w} L_{\mathcal{B}}(\mathbf{w}_t)\right)^2 \end{aligned} \tag{35}$$

in which the usual stochastic gradient update is modified with a term which penalizes the trace of the ratio between the diagonal of the Empirical Fisher Information Matrix (FIM) and an exponentially weighted average the of the Empirical FIM diagonal. By augmenting the loss with a TRACER term and maintaining a smoothed squared-gradient estimate, in principle, any optimization scheme can be modified in the same way. In our experiments, we use SGD with momentum for vision tasks and Adam-TRACER (Adam with a G-TRACER penalty) for NLP tasks, based on standard practice in each problem domain.

3 Results

While the original SAM paper (Foret et al., 2020) and subsequent papers Kwon et al. (2021) (Kim et al., 2022a) largely focus on standard benchmark problems and show marginal improvements in many settings, the most striking and practically relevant improvements concern the performance gains in the more challenging noisy-label settings. The CIFAR-10/100 benchmarks are extremely well understood, and good training schedules, data augmentations, and architecture choices have all been found over an extremely large number of trials run by the community over many years. The effect size of augmentations is often large (10% accuracy gains, or more) compared to post-augmentation gains exhibited by SAM (typically on the order of 1%).

Given our focus on delivering material performance gains in challenging settings, we examine the performance of our algorithm on especially challenging variants of standard benchmarks as they are a good model for the kinds of real-world applications that most require general-purpose regularizations. For example, we apply our method to noisy variants (with up to 50% label noise) of CIFAR-100, with and without data augmentations, first using a standard ResNet architecture and then on a vision transformer architecture (ViT). The ViT is trained from scratch (no pre-training), which is extremely challenging, since the convolution's inductive bias of spatial locality is lost in moving to a transformer architecture. We are thus using variants on standard benchmarks as a model for general settings, such as financial time-series forecasting, in which low effective sample size, extremely low signal-noise ratio, extreme nonstationarity, and a general lack of symmetries (financial time series exhibit neither up-down symmetry nor time-reversal symmetry) which give rise to data augmentations, all contrive to make generalization very difficult.

Finally, we show encouraging results on challenging subtasks from one of the most challenging NLP benchmarks, SuperGlue, using the BERT transformer architecture.

3.1 Vision: CIFAR-100, challenging variants

We first examine a variant on a standard benchmark in computer vision, CIFAR-100. We compare SGD, SAM and SGD-Tracer using none of the standard regularizations (no data augmentation, no weight-decay) and a standard training protocol (200 epochs, initial learning rate set to 0.1, cosine learning-rate decay). Furthermore, we randomly flip 50% of the labels so that 50% of the examples are incorrectly labeled.

The noisy label case is of primary interest and relevance in our setting, and relates to both SAM and G-TRACER type schemes which provide robustness w.r.t. weight perturbation since noise can be transferred to the weights. To see this, consider the single-layer case with weight matrix W and input x: we have $L(W + \Delta W, x) = L(W, x + \Delta x)$ for the choice $\Delta W = \frac{W \Delta x}{||x||_2^2} x^T$ (indeed, there are infinitely many solutions to the underlying matrix equation), so that input noise robustness corresponds to robustness in weight-space. This type of construction can be generalized to deeper architectures (Seong et al., 2018). Methods like G-Tracer and SAM which are robust to weight perturbations are for this reason expected to be robust to noise. Indeed, the most convincing and striking results in the original SAM paper concern robustness to label noise.

3.1.1 CIFAR-100 baseline with augmentation, consistency check

As a baseline and to establish consistency with other results in the literature and in order to demonstrate empirically that our training procedure is such that our models are well-trained, we apply the basic standard data-augmentations (rescaling, random cropping and flipping) together with a ResNet-20 architecture to the CIFAR-100 benchmark.

Table 1: CIFAR-100: ResNet20, accuracy (standard error)

	with aug
SGD	70.02% (0.36)
SAM	70.33% (0.22)
SGD-TRACER	**70.71%** (0.36)

The exact experimental setting for the vision tasks (unless otherwise indicated) follows standard practice and is as follows: SGD with weight decay/ridge penalty 5×10^{-4}, momentum 0.9, initial learning rate 0.1, 200 episodes, cosine learning rate decay to 0, batch size 128, global clip-norm= 1.0. The search spaces for the SAM and G-Tracer penalties ρ were chosen by first running on a logarithmic grid of 10 values $[1 \times 10^{-5}, \ldots, 1 \times 10^{4}]$ and then refining the range based on in-sample convergence, in order the span the space of plausible regularization strengths. These results are in line with (in fact, competitive with) the results in (Möllenhoff & Khan, 2022) (Kwon et al., 2021). [17]

3.1.2 CIFAR-100 50% noise, no regularization

Having established the baseline, we now consider the challenging setting where we randomly flip 50% of the labels and drop all augmentations (we simply rescale the inputs), and use no weight decay. The results in Table 2 show that GTRACER significantly improves upon SAM in this challenging setting. In Figure 1 we highlight the results for the same problem over different values of the regularization parameter ρ. In Figure

---

[17] As a further consistency check with practice, follow the training protocol (stepwise learning rate decay over 200 episodes, with learning rates $[.1, .02, .004, .0008]$ at $[0, 60, 120, 160]$) in `https://github.com/weiaicunzai/pytorch-cifar100/tree/master?tab=readme-ov-file` with larger architectures, eg ResNet-18 (11M parameters), and match the expected results for SGD, and see similar improvements vs SGD (75.8% accuracy vs 75.1% accuracy) and SAM (75.3% accuracy, $\rho = .05$).

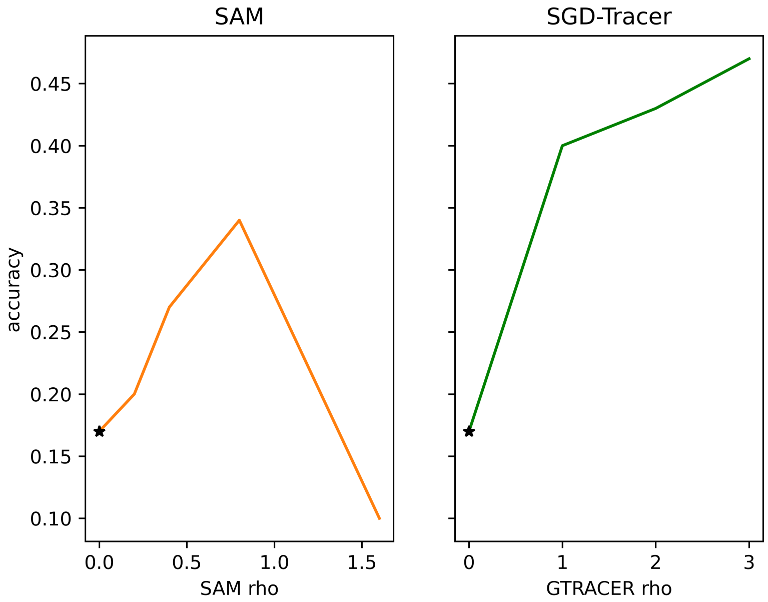

Figure 1: CIFAR 100: ResNet20, no weight-decay, 50% noise, accuracy vs regularization strength. GTRACER dominates the baseline and SAM across a wide range of regularization strengths.

2 we compare the training curves on this problem.

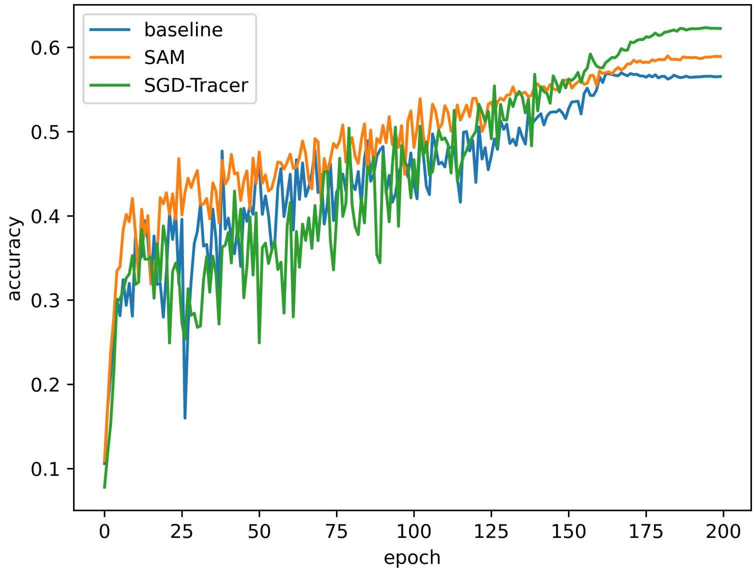

Figure 2: CIFAR 100: ResNet20, 50% noise, test-accuracy training curves. On a standard 200 epoch training protocol with cosine learning-rate decay, SGD-Tracer converges to a solution that generalizes materially better than SGD and SAM

Table 2: CIFAR 100: ResNet20, no weight-decay, 50% noise, accuracy (standard error)

	no aug
SGD	17.5% (2.41)
SAM	34.63% (1.85)
SGD-TRACER	**47.55%** (1.51)

3.1.3 CIFAR-100 results with weight decay

We next run SGD-Tracer on CIFAR-100 with and without label noise, with and without augmentation, with random label flipping and with a standard ridge penalty of 5×10^{-4}. The results in Table 3 show that SGD-TRACER performs consistently well, with a particularly strong advantage in the presence of noise and/or without additional regularization in the form of data augmentation.

Table 3: CIFAR-100: ResNet20, accuracy (standard error)

	no aug	**with aug**	**50% noise & no aug**
SGD	51.43 % (0.41)	70.02% (0.36)	21.96% (0.36)
SAM	58.98 % (0.52)	70.33% (0.22)	49.89% (0.32)
SGD-TRACER	**63.47%** (0.32)	**70.71%** (0.36)	**51.62%** (0.18)

3.1.4 ViT, no pretaining

We now turn to transformer architectures and use the Keras ViTTiny16 vision transformer architecture from the KerasCV library (Wood et al., 2022). We apply the standard augmentations as above with initial learning rate 1×10^{-4}, and batch size 256. We use this task to investigate the potential for further boosting the performance of G-TRACER by mitigating the gradient magnitude approximation, by splitting each batch into 4 sub-batches, computing squared gradients on each sub-batch and aggregating. [18] For fairness we also compute the SAM gradient on 4 sub-batches and average (as explored in Foret et al. (2020)). We see that this batch splitting delivers strong results for G-Tracer and suggests that moving to per-example gradients could significantly strengthen empirical results. We see that, despite the considerable challenge in losing the inductive bias of locality which drives the success of CNNs on vision tasks, SGD-TRACER is able to deliver a 10% performance boost to the naive SGD solution.

Table 4: CIFAR-100: ViT, accuracy (standard error)

	with aug
SGD	37.7 % (0.71)
SAM	38.2 % (0.52)
SAM batch-split	38.7 % (0.44)
SGD-TRACER	39.1 % (0.32)
SGD-TRACER batch-split	**41.6** % (0.28)

3.2 NLP

For NLP tasks we use the Huggingface Bert-base-uncased (Devlin et al., 2018) checkpoint together with Adam-TRACER with max sequence length 256 and batch size 8. We fine-tune using Adam-Tracer, using a standard protocol of 5 epochs with initial learning rate 2×10^{-5} and decay the learning rate using a linear schedule, with final learning rate 1×10^{-5}. The search ranges for the SAM and G-Tracer penalty parameters

---

[18] Using tools such as (Dangel et al., 2020) would allow per-example squared gradients to be calculated.

are chosen as in the vision experiments. Each run is repeated 20 times. We choose 3 distinct fine-tuning tasks[19]:

- BoolQ: boolean question answering (Clark et al., 2019)
- WiC: Words in Context (Pilehvar & Camacho-Collados, 2018)
- RTE: Recognizing Textual Entailment (Wang et al., 2020)

and we see that Adam with a G-TRACER performs competitively, and has the additional property of producing more stable results across runs (as reflected in the standard errors). Note that in this setting, the extensive pre-training (Devlin et al., 2018) combined with fine-tuning acts as a strong regularizer and that the relative performance gains we see from using a G-TRACER are smaller than those we observe in the ViT setting without pretraining.

Table 5: NLP tasks BERT base-uncased results, accuracy (standard error)

	BOOLQ	**WIC**	**RTE**
Adam	73.84% (0.14)	69.36% (0.08)	69.18% (0.33)
SAM	73.95% (0.13)	69.06% (0.07)	69.54% (0.28)
Adam-TRACER	**75.09%** (0.04)	**70.01%** (0.06)	**70.13%** (0.18)

4 Conclusion

Motivated by the notable empirical success of SAM, a prior that flat (in expectation, and in an intrinsic, geometric sense) minima should generalize better than sharp minima, and noting the connections between the generalized Bayes objective and SAM, we have derived a new algorithm that is simple to implement and understand, cheap to evaluate, provably convergent, naturally scale-independent (and approximately coordinate-free) and which shows promising performance on standard benchmarks in vision and NLP, and across transformer and convolutional architectures. Performance is notably strong for challenging low signal-to-noise ratio and large batch problems, and in settings where other regularizations (data augmentations, tailored learning rate schedules, weight decay) are not used. Crucially, the algorithm is straightforwardly derived from an approximate natural gradient optimization of an ELBO-type objective and does not rely on the use of small batch sizes (or "m-sharpness" (Foret et al., 2020)) or other poorly understood (and frequently expensive to compute) heuristics.

A Appendix

A.1 G-Tracer detailed derivation

We begin our exposition with a background on generalized variational posteriors and then derive the G-Tracer regularizer by performing natural gradient variational inference.

A.1.1 Generalized variational posterior

Our starting point is similar to that of SAM, but uses a more general objective, which arises in the variational optimization of a generalized posterior distribution, q, over the space of probability measures $\mathcal{P}(\Theta)$ on the parameter space Θ given by (Bissiri et al., 2016):

$$q^*(\boldsymbol{w}) = \arg\min_{q \in \mathcal{P}(\Theta)} \left\{ \mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \mathbb{D}_{KL}[q, p] \right\} \tag{36}$$

---

[19] All of which have been included in the challenging SuperGlue benchmark

for which, when $Z = \int_{\Theta} \exp\left\{-\sum_{i=1}^{n} l(\boldsymbol{w}, \boldsymbol{x_i})\right\} \pi(\boldsymbol{\theta}) d\boldsymbol{\theta} < \infty$, the solution is given by the generalized posterior:

$$q^*(\boldsymbol{w}) \propto p(\boldsymbol{w}) \prod_{i=1}^{N} \exp\{-l(\boldsymbol{w}, \boldsymbol{x_i})\} \tag{37}$$

The terms $\exp\{-l(\boldsymbol{w}, \boldsymbol{x_i})\}$ are to be interpreted as quasi-likelihoods, and for the particular choice $l(\boldsymbol{w}, \boldsymbol{x_i}) = -\log p(\boldsymbol{x_i}|\boldsymbol{w})$, we recover the standard Bayesian posterior. As this infinite dimensional optimization is in general intractable, it is usual to assume that the posterior belongs to a parametric family $\mathcal{Q} \subset \mathcal{P}$:

$$q^*(\boldsymbol{w}) = \arg\min_{q \in \mathcal{Q}(\Theta)} \left\{\mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \mathbb{D}_{KL}[q, p]\right\} \tag{38}$$

which, for the choice $l(\boldsymbol{w}, \boldsymbol{x_i}) = -\log p(\boldsymbol{x_i}|\boldsymbol{w})$, is the same objective (up to a constant factor) as the evidence lower bound (ELBO) used in variational Bayes.

In practice, it is often found that tempering the KL divergence term by a positive factor $\rho < 1$ produces optimal performance, giving rise to:

$$q^*(\boldsymbol{w}) = \arg\min_{q \in \mathcal{Q}(\Theta)} \left\{\mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \rho \mathbb{D}_{KL}[q, p]\right\} \tag{39}$$

A.1.2 Derivation of the TRACER flatness-inducing regularizer

Following Khan & Rue (2021) and Zhang et al. (2017), we make the assumption $q(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and seek to optimize the variational objective in equation 39 w.r.t. the variational parameters $\boldsymbol{\phi} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ [20] using natural gradient descent. This allows us to derive an algorithm that respects the intrinsic geometry of the parameter space, and thus derive an algorithm that seeks sharp minima in an approximately coordinate-independent way.

Thus, we aim to minimize:

$$\mathcal{L}(\boldsymbol{\phi}) := \mathbb{E}_{q(\boldsymbol{w})}[L(\boldsymbol{w})] + \rho \mathbb{D}_{KL}[q, p] \tag{40}$$

w.r.t. $\boldsymbol{\phi}$ where ρ is a positive real-valued regularization parameter. The negative gradient corresponds to the steepest descent direction in the Euclidean metric:

$$\frac{-\boldsymbol{\nabla}_{\boldsymbol{\phi}} \mathcal{L}}{\|\boldsymbol{\nabla}_{\boldsymbol{\phi}} \mathcal{L}\|} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \operatorname*{argmin}_{\boldsymbol{\Delta\phi}: \|\boldsymbol{\Delta\phi}\|_2 < \epsilon} \mathcal{L}(\boldsymbol{\phi} + \boldsymbol{\Delta\phi}) \tag{41}$$

and thus depends on the chosen coordinates $\boldsymbol{\phi}$. In contrast, the so-called natural gradient update corresponds to steepest descent in the KL-divergence metric:

$$\frac{-F^{-1} \boldsymbol{\nabla}_{\boldsymbol{\phi}} \mathcal{L}}{\|\boldsymbol{\nabla}_{\boldsymbol{\phi}} \mathcal{L}\|} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \operatorname*{argmin}_{\boldsymbol{\Delta\phi}: \mathbb{D}_{KL}[q_{\boldsymbol{\phi}}, q_{\boldsymbol{\phi}+\boldsymbol{\Delta\phi}}] < \epsilon} \mathcal{L}(\boldsymbol{\phi} + \boldsymbol{\Delta\phi}) \tag{42}$$

where F is the FIM:

$$F := \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})} \left[\boldsymbol{\nabla}_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w})^T \boldsymbol{\nabla}_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w})\right] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})} \left[-\boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w})\right] \tag{43}$$

which defines a Riemannian metric on the variational parameter manifold. Expanding to second order in a small neighborhood of $\boldsymbol{\phi}$ we have:

$$\mathbb{D}_{KL}[q_{\boldsymbol{\phi}}, q_{\boldsymbol{\phi}+\boldsymbol{\Delta\phi}}] = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})} \left[-\boldsymbol{\Delta\phi}^T \boldsymbol{\nabla}_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w}) - \frac{1}{2} \boldsymbol{\Delta\phi}^T \boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w}) \boldsymbol{\Delta\phi}\right] + O(||\boldsymbol{\Delta\phi}||^3) \tag{44}$$

and since:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})} \boldsymbol{\nabla}_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})} \left[\frac{\boldsymbol{\nabla}_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\boldsymbol{w})}{q_{\boldsymbol{\phi}}(\boldsymbol{w})}\right] = \boldsymbol{\nabla}_{\boldsymbol{\phi}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})}[1] = 0 \tag{45}$$

---

[20] We will write, to keep the notation as light as possible, the set of variational parameters as $\boldsymbol{\phi} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$. Depending on the context, e.g. when we write the gradient $\boldsymbol{\nabla}_{\boldsymbol{\phi}}$, we will take this to mean $\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\mu} \\ \text{vec}(\boldsymbol{\Sigma}) \end{bmatrix}$

the FIM (under certain regularity conditions) can be seen to be the Hessian (or curvature) of the K-L divergence:

$$\mathbb{D}_{KL}[q_{\boldsymbol{\phi}}, q_{\boldsymbol{\phi}+\boldsymbol{\Delta\phi}}] = -\frac{1}{2}\boldsymbol{\Delta\phi}^T \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{w})}\left[\boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{w})\right]\boldsymbol{\Delta\phi} + O(||\boldsymbol{\Delta\phi}||^3) = \frac{1}{2}\boldsymbol{\Delta\phi}^T F \boldsymbol{\Delta\phi} + O(||\boldsymbol{\Delta\phi}||^3) \tag{46}$$

It turns out that the update equations have a particularly simple form when $q_{\boldsymbol{\phi}}(\boldsymbol{w})$ is parameterized as $\boldsymbol{\phi} = (\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$. The following proposition gives expressions for the natural gradient vectors w.r.t. the mean and precision (for proof see Appendix A.5):

Proposition 1. *For a probability distribution with pdf $q_{\boldsymbol{\phi}}(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$ with the parameterization $\boldsymbol{\phi} = (\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$, the natural gradients of $\mathcal{L}$ w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$ of are given by:*

$$\begin{aligned} \tilde{\boldsymbol{\nabla}}_{\boldsymbol{\mu}}\mathcal{L} &= \boldsymbol{\Sigma}\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}) + \rho \boldsymbol{\nabla}_{\boldsymbol{w}} p(\boldsymbol{w})] \\ \tilde{\boldsymbol{\nabla}}_{\boldsymbol{\Lambda}}\mathcal{L} &= -\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho \boldsymbol{\nabla}^2_{\boldsymbol{w}} p(\boldsymbol{w})] + \rho \boldsymbol{\Sigma}^{-1} \end{aligned} \tag{47}$$

Assuming an isotropic Gaussian prior, $p(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{0}, \eta \boldsymbol{I})$, performing gradient descent w.r.t. this natural gradient then leads to the following iterative update equations:

$$\begin{aligned} \boldsymbol{\mu} &\leftarrow \boldsymbol{\mu} - \alpha \boldsymbol{\Lambda}^{-1}\left(\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})] + \frac{\rho}{\eta}\boldsymbol{w}\right) \\ \boldsymbol{\Lambda} &\leftarrow (1-\beta)\boldsymbol{\Lambda} + \beta\left(\frac{\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})]}{\rho} + \eta^{-1}\boldsymbol{I}\right) \end{aligned} \tag{48}$$

where α and β are the learning rates for the mean and precision updates, respectively. We work with each of these update equations in turn. Starting with the update equation for the mean $\boldsymbol{\mu}$, the key observation is that the expectation $\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})]$ is taken with respect to the distribution $q(\boldsymbol{w})$, which is an exponential moving average of the expected Hessian $\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})]$. This updating happens naturally as a consequence of taking natural gradient steps, and leads to an approximately coordinate-free algorithm in the sequel. Applying Bonnet's theorem (Khan & Rue, 2021) and forming the second-order approximation to the loss we obtain:

$$\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})] = \boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathbb{E}_q[L(\boldsymbol{w})] \approx \boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathbb{E}_q[L(\boldsymbol{\mu}) + \frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^T \boldsymbol{\nabla}^{\boldsymbol{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}(\boldsymbol{w}-\boldsymbol{\mu})] \tag{49}$$

We also have:

$$\mathbb{E}_q[(\boldsymbol{w}-\boldsymbol{\mu})^T \boldsymbol{\nabla}^{\boldsymbol{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}(\boldsymbol{w}-\boldsymbol{\mu})] = \mathbb{E}_q[\mathrm{Tr}\left((\boldsymbol{w}-\boldsymbol{\mu})^T \boldsymbol{\nabla}^{\boldsymbol{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}(\boldsymbol{w}-\boldsymbol{\mu})\right)] = \mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{H}) \tag{50}$$

where $\boldsymbol{H}$ is the Hessian $\boldsymbol{\nabla}^{\boldsymbol{2}}_{\boldsymbol{w}} L(\boldsymbol{w})$. We therefore have that:

$$\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})] \approx \boldsymbol{\nabla}_{\boldsymbol{\mu}}[L(\boldsymbol{\mu}) + \frac{1}{2}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{H})] \tag{51}$$

Choosing the prior variance η to be infinite and thus ignoring terms involving η in both update equations (corresponding to an improper prior, and so consistent with the discussion above), leads to the following update for the mean:

$$\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \alpha \boldsymbol{\Lambda}\left(\boldsymbol{\nabla}_{\boldsymbol{\mu}}[L(\boldsymbol{\mu}) + \frac{1}{2}\mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{H})]\right) \tag{52}$$

Thus, in order to blur the loss with multivariate Gaussian noise in a way that aligns with the intrinsic geometry of the parameter space, we can (to second order) augment the loss with a term involving the Trace of the Hessian. Considering now the update equation for the precision, we can use Price's theorem (Khan & Rue, 2021) together with a Taylor expansion to get, to second order $\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})] \approx \boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}$ (see Appendix A.3 for details), which leads to

$$\boldsymbol{\Lambda} \leftarrow (1-\beta)\boldsymbol{\Lambda} + \beta\left(\frac{\boldsymbol{\nabla}^{\boldsymbol{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}}{\rho}\right) \tag{53}$$

We next substitute, as is common in the literature using approximate second order approximation (Martens, 2020), the log-likelihood Fisher Information Matrix $\boldsymbol{F_w}$ for the Hessian, where:

$$\boldsymbol{F_w} = \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w}), \boldsymbol{x} \sim p(\boldsymbol{x})}[-\boldsymbol{\nabla}^2_{\boldsymbol{w}} \log p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})] \tag{54}$$

Substituting $\boldsymbol{F_w}$ for the Hessian in the update equation for the precision 53, and rewriting the update in terms of $\boldsymbol{G} := \rho\boldsymbol{\Lambda}$, absorbing constants into ρ and α, and writing the iteration in terms of the parameter $\boldsymbol{w}$, we obtain the general update equations:

$$\begin{aligned} \boldsymbol{w} &\leftarrow \boldsymbol{w} - \alpha\left(\boldsymbol{\nabla_w}[L(\boldsymbol{w}) + \rho\mathrm{Tr}(\boldsymbol{G^{-1}F_w})]\right) \\ \boldsymbol{G} &\leftarrow (1-\beta)\boldsymbol{G} + \beta\boldsymbol{F_w} \end{aligned} \tag{55}$$

Crucially, the penalty term $\rho\mathrm{Tr}(\boldsymbol{G^{-1}F})$ can be seen to be invariant to affine coordinate transformations, since it is the trace of the ratio of two (0,2) tensors which transform in the same way. Indeed, under an affine coordinate transformation with Jacobian $\boldsymbol{J}$, we have $\boldsymbol{F_w} \to \boldsymbol{J^T F'_w J}$ and $\boldsymbol{G} \to \boldsymbol{J^T G' J}$ so that:

$$\mathrm{Tr}\left(\boldsymbol{G^{-1}F_w}\right) = \mathrm{Tr}\left(\boldsymbol{J^{-1}G'^{-1}J^{T^{-1}}J^T F'_w J}\right) = \mathrm{Tr}\left(\boldsymbol{J^{-1}G'^{-1}F'_w J})\right) = \mathrm{Tr}\left(\boldsymbol{G'^{-1}F'_w}\right) \tag{56}$$

By using the ratio of the (squared) gradients and the exponentially smoothed gradients, the trace ratio in effect penalizes the change in (squared) gradient, in a coordinate-free way. More generally, given a smooth coordinate change defined by a diffemorphism $\Phi : \mathbb{R}^p \to \mathbb{R}^p$ and Jacobian $J(\boldsymbol{w})$, then given sufficiently rapid exponential decay in the update equation for the Fisher, subject to Φ having sufficient regularity, the penalty term is readily seen to be approximately coordinate free.

Although the evaluation of the GGN matrix, in particular the matrix multiplications involving the Jacobians J_f, can be relatively costly, the FIM can be expressed as an expectation of outer products of gradients w.r.t. the output distribution $p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$:

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x_i},\boldsymbol{w})}\left[\boldsymbol{\nabla_w}\log p(\boldsymbol{y}|\boldsymbol{x_i},\boldsymbol{w})^T\boldsymbol{\nabla_w}\log p(\boldsymbol{y}|\boldsymbol{x_i},\boldsymbol{w})\right] \approx \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\nabla_w}\log p(\tilde{\boldsymbol{x_i}}|\boldsymbol{x_i},\boldsymbol{w})^T\boldsymbol{\nabla_w}\log p(\tilde{\boldsymbol{y_i}}|\boldsymbol{x_i},\boldsymbol{w}) \tag{57}$$

which, following Martens (2020), can be estimated using a single Monte Carlo sample from the output distribution: $\tilde{\boldsymbol{y}} \sim p(\boldsymbol{y}|\boldsymbol{x_i},\boldsymbol{w})$. Using this (biased) Fisher approximation in our setting thus requires gradients to be calculated through an expectation $\boldsymbol{\nabla_w}\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})}[L(\boldsymbol{w};\boldsymbol{y})]$, approximated using a Monte Carlo sample from the model's output distribution. Since the expectation is taken w.r.t. a distribution which depends on $\boldsymbol{w}$, it is necessary to reparameterize so that the discrete Monte Carlo sample is expressed as the deterministic transformation of a $g_{\boldsymbol{w}}(\boldsymbol{z})$ (depending on $\boldsymbol{w}$) of a sample $\boldsymbol{z} \sim h_{\boldsymbol{\theta}}(\boldsymbol{z})$ from a distribution not depending on $\boldsymbol{w}$, so that $\mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})}[L(\boldsymbol{w};\boldsymbol{y})] = \mathbb{E}_{\boldsymbol{z}\sim h_{\boldsymbol{\theta}}(\boldsymbol{z})}[L(\boldsymbol{w};g_{\boldsymbol{w}}(\boldsymbol{z})]$. In the discrete case (corresponding to classification), since the argmax function is non-differentiable, the standard approach is the Gumbel-Softmax reparameterization (Jang et al., 2016), which uses the softmax function as a continuous relaxation of the argmax function together with i.i.d. samples distributed as Gumbel(0,1).

It is important to note that this approach is different from simply evaluating $\log p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})$ on the training labels, a widely used approximation known as the empirical Fisher $\boldsymbol{F}_{\mathrm{emp}}$:

$$\boldsymbol{F_{\mathrm{emp}}} := \sum_{i=1}^{n}\boldsymbol{\nabla_w}\log p(\boldsymbol{y_i}|\boldsymbol{x_i},\boldsymbol{w})^T\boldsymbol{\nabla_w}\log p(\boldsymbol{y_i}|\boldsymbol{x_i},\boldsymbol{w}) \tag{58}$$

This, despite lacking the same convergence guarantees, performs competitively in many settings (Kunstner et al., 2019). We find in our experiments that the empirical Fisher performs competitively with the MC approximation to the GGN (Khan et al., 2018; Kingma & Ba, 2014) and has the advantage of being straightforward and cheap to compute from already computed gradients (in the case of Adam-TRACER, the smooth squared gradients are already computed and maintained for use as a preconditioner). Given the conceptual and computational simplicity of this approach we substitute the empirical Fisher for the Hessian. Recent advances in approximate second-order methods in optimization, notably Yao et al. (2020), suggest avenues for improvement, and we leave investigations of alternatives, such as the smoothed (Hessian-free) Hessian diagonal sketch used in AdaHessian, for future work.

We now make two simplifications. First, we use a mean-field approximation, representing the FIM by its diagonal, as is done in Adam (Kingma & Ba, 2014), Adagrad (Duchi et al., 2011) and RMSProp thus:

$$\boldsymbol{F_{\text{diag}}} \approx \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})^2 \tag{59}$$

Secondly, it is standard practice to (Martens, 2020) to add Tikhonov regularization or damping via a small positive real constant δ when using 2nd-order optimization methods, giving in this case the preconditioner: $(\boldsymbol{F_e} + \delta \boldsymbol{I})^{-1}$. In fact this would arise naturally in our setup by choosing η to be non-zero, in which case we would simply have $\delta := \frac{\rho}{\eta}$. From an optimization perspective, it is justified by recognizing that the local quadratic model from which the second-order update is ultimately derived is a second-order approximation to the KL divergence and is thus only valid locally. For directions corresponding to small eigenvalues, parameter updates can lie outside the region where the approximation is reasonable (Martens, 2020). This is true, a fortiori, when diagonal approximations are used, as is the case here. As our emphasis here is on geometric regularization, we drop the preconditioner entirely by choosing δ to be sufficiently large that the preconditioner is equal to the identity (up to a constant, which is absorbed into the learning rate).

Finally, as most current deep learning frameworks don't straightforwardly support access to per-example gradients, which can in principle be achieved with negligible additional cost (see, for example, BackPACK Dangel et al. (2020) second-order Pytorch extensions), for simplicity and efficiency, we use the gradient magnitude (GM) approximation (Bottou et al., 2016), as used in standard optimizers such as Adam, Adagrad, and RMSprop, replacing the sum of squared gradients with the square of summed gradients:

$$\frac{1}{n}\sum_{i=1}^{n} \left[\boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})\right]^2 \approx \left[\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\nabla_w} \log p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{w})\right]^2 \tag{60}$$

Writing the resulting FIM diagonal as $(\boldsymbol{\nabla_w} L(\boldsymbol{w}))^2$, and using stochastic gradient updates computed on on minibatches $\mathcal{B}$ of the data, $\boldsymbol{\nabla_w} L_{\mathcal{B}}(\boldsymbol{w})$, we finally end up with the following update (Algorithm 1):

$$\begin{aligned} \mathbf{w}_{t+1} &= \mathbf{w}_t - \alpha_t \boldsymbol{\nabla}_{\mathbf{w}} \left[L_{\mathcal{B}}(\mathbf{w}_t) + \rho \left(\boldsymbol{\nabla}_{\mathbf{w}} L_{\mathcal{B}}(\mathbf{w}_t)\right)^2 / (\mathbf{f}_t + \delta)\right] \\ \mathbf{f}_{t+1} &= (1-\beta)\mathbf{f}_t + \beta \left(\boldsymbol{\nabla_w} L_{\mathcal{B}}(\mathbf{w}_t)\right)^2 \end{aligned} \tag{61}$$

We show in Appendix A.6 that the algorithm converges to a neighborhood of a local minimum of $L(\boldsymbol{w})$ of size $\mathcal{O}(\rho^2)$. We note in passing that, in this simplest form (after applying the gradient magnitude approximation), the update equations amount to regularizing with a (scale-adjusted) gradient norm. In principle (particularly for the large batch case) we would expect to see significant improvements by moving to per-gradient calculations (which are theoretically no more expensive to compute, but require additional work under most current ML frameworks).

A.2 Multivariate Gaussian Fisher Information Matrix

For a probability distribution with density q with parameters $\boldsymbol{\phi}$, the Fisher Information Matrix (FIM) can be written as the expected negative log-likelihood Hessian:

$$F = \mathbb{E}_q \left[-\boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q\right] \tag{62}$$

In particular, for a multivariate Gaussian with pdf: $q(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$, parameterized by $\boldsymbol{\phi} = \begin{bmatrix}\boldsymbol{\mu} \\ \text{vec}(\boldsymbol{\Lambda})\end{bmatrix}$ the negative log-likelihood is, up to constant terms:

$$-\log q(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Lambda}(\boldsymbol{x}-\boldsymbol{\mu}) + \frac{1}{2}\log|\boldsymbol{\Lambda}^{-1}| \tag{63}$$

Taking gradients w.r.t. $\boldsymbol{\mu}$, we have: $-\boldsymbol{\nabla}_{\boldsymbol{\mu}} \log q(\boldsymbol{x}) = \boldsymbol{\Lambda}(\boldsymbol{x} - \boldsymbol{\mu})$ and therefore $\mathbb{E}_q \left[-\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\mu}} \log q\right] = \boldsymbol{\Lambda}$. Taking gradients w.r.t. the covariance, and since $\boldsymbol{\nabla}_{\boldsymbol{\Lambda}}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Lambda}(\boldsymbol{x} - \boldsymbol{\mu}) = (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T$ and $\boldsymbol{\nabla}_{\boldsymbol{\Lambda}} \log |\boldsymbol{\Lambda}^{-1}| = \boldsymbol{\nabla}_{\boldsymbol{\Lambda}} \log |\boldsymbol{\Lambda}|^{-1} = -\boldsymbol{\nabla}_{\boldsymbol{\Lambda}} \log |\boldsymbol{\Lambda}| = -(\boldsymbol{\Lambda}^T)^{-1} = -(\boldsymbol{\Lambda})^{-1}$ we have:

$$-\boldsymbol{\nabla}_{\boldsymbol{\Lambda}} \log q(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T - \frac{1}{2}\boldsymbol{\Lambda}^{-1} \tag{64}$$

Finally, writing $\boldsymbol{\nabla}_{\boldsymbol{\Lambda}} \boldsymbol{\Lambda}^{-1}$ as $-\boldsymbol{\Lambda} \otimes \boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}^{-1} := \boldsymbol{\Sigma}$ we have:

$$-\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\Lambda}} \log q(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{\Sigma}^{-1} \otimes \boldsymbol{\Sigma}^{-1} \tag{65}$$

so that the FIM is given by:

$$F = \mathbb{E}_q \left[-\boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q\right] = \begin{bmatrix} \boldsymbol{\Sigma}^{-1} & 0 \\ 0 & \frac{1}{2}\boldsymbol{\Sigma}^{-1} \otimes \boldsymbol{\Sigma}^{-1} \end{bmatrix} \tag{66}$$

A.3 Approximate expected Hessian

Lemma 1. *To second order, we can approximate the expected Hessian w.r.t. a multivariate Gaussian with pdf:* $q(x) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$ *by its value at the mean:*

$$\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})] \approx \boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}} \tag{67}$$

Proof. Following Khan & Rue (2021), by Price's theorem, we have:

$$\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})] = 2\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\Lambda}^{-1}} \mathbb{E}_q[L(\boldsymbol{w})] \tag{68}$$

expanding the r.h.s. to second order using a Taylor series, this is equivalent to:

$$2\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\Lambda}^{-1}} \mathbb{E}_q[(\boldsymbol{w} - \boldsymbol{\mu})^T \boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}(\boldsymbol{w} - \boldsymbol{\mu})] \tag{69}$$

Finally, noting that $\mathbb{E}_q[(\boldsymbol{w} - \boldsymbol{\mu})^T \boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}(\boldsymbol{w} - \boldsymbol{\mu})] = \operatorname{Tr}\left[\frac{1}{2}\boldsymbol{\Lambda}^{-1} \boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}\right]$, we have, to second order:

$$\mathbb{E}_q[\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})] \approx 2\boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{\Lambda}^{-1}} \operatorname{Tr}\left[\frac{1}{2}\boldsymbol{\Lambda}^{-1} \boldsymbol{\nabla}^{\mathbf{2}}_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}}\right] = \boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{\mu}} \tag{70}$$

□

A.4 Objective function gradient

Lemma 2. *The gradient of the objective 40 towards* $\boldsymbol{\phi}' = \begin{bmatrix} \boldsymbol{\mu} \\ \operatorname{vec}(\boldsymbol{\Sigma}) \end{bmatrix}$ *is given by:*

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}} \mathcal{L} = \mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho \boldsymbol{\nabla}_{\boldsymbol{w}} \log p(\boldsymbol{w})] \tag{71}$$

$$\boldsymbol{\nabla}_{\boldsymbol{\Sigma}} \mathcal{L} = \frac{1}{2}\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho \boldsymbol{\nabla}^2_{\boldsymbol{w}} \log p(\boldsymbol{w})] - \frac{\rho}{2}\boldsymbol{\Sigma}^{-1} \tag{72}$$

Proof. Taking the negative gradient of the objective w.r.t. to $\boldsymbol{\mu}$, and applying Bonnet's theorem (Khan & Rue, 2021), and the fact that the expectation of the score is 0, we have:

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}} \left(\mathbb{E}_q[L(\boldsymbol{w})] + \rho \mathbb{D}_{KL}[q(\boldsymbol{w}), p(\boldsymbol{w})]\right) = \mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w})] - \rho \mathbb{E}_q \left[\boldsymbol{\nabla}_{\boldsymbol{w}} \log p(\boldsymbol{w})\right] \tag{73}$$

Taking the gradient w.r.t. $\boldsymbol{\Sigma}$, and applying Price's theorem, we have:

$$\boldsymbol{\nabla}_{\boldsymbol{\Sigma}} \left(\mathbb{E}_q[L(\boldsymbol{w})] + \rho \mathbb{D}_{KL}[q(\boldsymbol{w}), p(\boldsymbol{w})]\right) = \frac{1}{2}\mathbb{E}_q \left[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) + \rho \boldsymbol{\nabla}^2_{\boldsymbol{w}} \log q(\boldsymbol{w}) - \rho \boldsymbol{\nabla}^2_{\boldsymbol{w}} \log p(\boldsymbol{w})\right] \tag{74}$$

and since:

$$\mathbb{E}_q\left[\boldsymbol{\nabla}^2_{\boldsymbol{w}} \log q(\boldsymbol{w})\right] = -\frac{1}{2}\mathbb{E}_q\left[\boldsymbol{\nabla}^2_{\boldsymbol{w}}\left(\log|\boldsymbol{\Sigma}| + (\boldsymbol{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{w}-\boldsymbol{\mu})\right)\right] = -\boldsymbol{\Sigma}^{-1} \tag{75}$$

We obtain

$$\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathcal{L} = \mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho\boldsymbol{\nabla}_{\boldsymbol{w}} \log p(\boldsymbol{w})] \tag{76}$$

$$\boldsymbol{\nabla}_{\boldsymbol{\Sigma}}\mathcal{L} = \frac{1}{2}\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho\boldsymbol{\nabla}^2_{\boldsymbol{w}} \log p(\boldsymbol{w})] - \frac{\rho}{2}\boldsymbol{\Sigma}^{-1} \tag{77}$$

□

A.5 Objective function natural gradient

Proposition 2. *The natural gradients of the objective 40 w.r.t. the parameters* $\boldsymbol{\phi} = (\boldsymbol{\mu}, \boldsymbol{\Lambda})$ *of* $q(x) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$ *are given by:*

$$\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\mu}}\mathcal{L} = \boldsymbol{\Sigma}\mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}) + \rho\boldsymbol{\nabla}_{\boldsymbol{w}} \log p(\boldsymbol{w})] \tag{78}$$

$$\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\Lambda}}\mathcal{L} = -\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho\boldsymbol{\nabla}^2_{\boldsymbol{w}} p(\boldsymbol{w})] + \rho\boldsymbol{\Lambda} \tag{79}$$

Proof. By Lemma 2, the gradients $\boldsymbol{\nabla}_{\boldsymbol{\phi}'}$ of the objective $\mathcal{L}(\boldsymbol{\phi})$ w.r.t. $\boldsymbol{\phi}' = \begin{bmatrix}\boldsymbol{\mu} \\ \text{vec}(\boldsymbol{\Sigma})\end{bmatrix}$ are given by:

$$\begin{aligned}\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathcal{L} &= \mathbb{E}_q[\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho\boldsymbol{\nabla}_{\boldsymbol{w}} \log p(\boldsymbol{w})] \\ \boldsymbol{\nabla}_{\boldsymbol{\Sigma}}\mathcal{L} &= \frac{1}{2}\mathbb{E}_q[\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}) - \rho\boldsymbol{\nabla}^2_{w} \log p(\boldsymbol{w})] - \frac{\rho}{2}\boldsymbol{\Sigma}^{-1}\end{aligned} \tag{80}$$

The Fisher Information Matrix is given by equation 66:

$$F = \mathbb{E}_{q_{\boldsymbol{\phi}}}\left[-\boldsymbol{\nabla}^2_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}\right] = \begin{bmatrix}\boldsymbol{\Sigma}^{-1} & 0 \\ 0 & \frac{1}{2}\boldsymbol{\Sigma}^{-1}\otimes\boldsymbol{\Sigma}^{-1}\end{bmatrix} \tag{81}$$

and therefore

$$F^{-1}\boldsymbol{\nabla}_{\boldsymbol{\phi}}\mathcal{L}(\boldsymbol{\phi}) = \begin{bmatrix}\boldsymbol{\Sigma} & 0 \\ 0 & 2\boldsymbol{\Sigma}\otimes\boldsymbol{\Sigma}\end{bmatrix}\begin{bmatrix}\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathcal{L} \\ \text{vec}(\boldsymbol{\nabla}_{\boldsymbol{\Lambda}}\mathcal{L})\end{bmatrix} = \begin{bmatrix}\boldsymbol{\Sigma}\boldsymbol{\nabla}_{\boldsymbol{\mu}}\mathcal{L} \\ \text{vec}(2\boldsymbol{\Sigma}\boldsymbol{\nabla}_{\boldsymbol{\Lambda}}\mathcal{L}\boldsymbol{\Sigma})\end{bmatrix} \tag{82}$$

where we used the identities $(\boldsymbol{B}^{\boldsymbol{T}}\otimes\boldsymbol{A})\text{vec}(\boldsymbol{X}) = \text{vec}(\boldsymbol{AXB})$ and $(\boldsymbol{A}\otimes\boldsymbol{B})^{\boldsymbol{-1}} = \boldsymbol{A}^{\boldsymbol{-1}}\otimes\boldsymbol{B}^{\boldsymbol{-1}}$. The gradient $\tilde{\boldsymbol{\nabla}}_{\mu}\mathcal{L}$ then follows immediately from the definition of the natural gradient operator. Using the chain rule for matrix derivatives we have that:

$$\boldsymbol{\nabla}_{\boldsymbol{\Lambda}}\mathcal{L} = -\boldsymbol{\Lambda}\boldsymbol{\nabla}_{\boldsymbol{\Sigma}}\boldsymbol{\Lambda} \tag{83}$$

Since $\text{vec}(2\boldsymbol{\Sigma}\boldsymbol{\nabla}_{\boldsymbol{\Lambda}}\mathcal{L}\boldsymbol{\Sigma}) = \text{vec}(-2\boldsymbol{\nabla}_{\boldsymbol{\Sigma}}\mathcal{L})$, we have the required updates. □

A.6 Convergence analysis

With $T(\boldsymbol{w}_t) := \left\langle(\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t))^2, (\overline{\boldsymbol{f}}+\delta)_t^{-1}\right\rangle$, as $\rho\to 0$, the iterates $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha_t\boldsymbol{\nabla}_{\boldsymbol{w}}\left[L(\boldsymbol{w}_t) + \rho\boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\right]$ will converge to those of SGD. For $\rho > 0$, the algorithm is biased away from a pure descent direction, and convergence then depends on the magnitude of ρ. The key assumption in the following convergence proof is that $\|\rho\boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|_2^2 \leq \kappa\|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|_2^2 + \zeta$, which controls the bias. This follows from the standard assumption of twice-differentiability of $L(\boldsymbol{w})$ and the Lipschitz continuity of $\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)$, which imply that the Hessian has a bounded spectral norm:

$$\begin{aligned}\|\rho\boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|_2^2 &\leq 4\rho^2\|\boldsymbol{\nabla}^2_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|_2^2\|(\overline{\boldsymbol{f}}+\delta)_t^{-1}\|_2^2 \\ &\leq 4\left(\frac{\rho}{\delta}\right)^2 C^2 p\end{aligned} \tag{84}$$

so that ζ depends on the Lipschitz constant C and the ratio $\frac{\rho}{\delta}$.

Theorem 3. *Let* $T(\boldsymbol{w}_t) := \left\langle \left(\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\right)^2, \overline{\boldsymbol{f}}_t^{-1} \right\rangle$*, and assume the objective (loss)* $L : \mathbb{R}^p \to \mathbb{R}$ *is Lipschitz continuous, twice differentiable, and has Lipschitz-continuous gradient. Let us assume, following Bottou et al. (2016) and Ajalloeian & Stich (2021) that we have a stochastic direction* $g(\boldsymbol{w}_t, \boldsymbol{\xi}_t)$ *which has the following properties,* $\forall t$*:*

$$\mathbb{E}\left[g(\boldsymbol{w}_t, \boldsymbol{\xi}_t)\right] = \boldsymbol{\nabla}_{\boldsymbol{w}} L + \rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t) \tag{85}$$

and further assuming that there exist M*,* M_G *such that,* $\forall t$*,*

$$\mathbb{E}\left[\|g(\boldsymbol{w}_t, \boldsymbol{\xi}_t)\|^2\right] \le M + M_G \|\boldsymbol{\nabla}_{\boldsymbol{w}} L + \rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|^2 \tag{86}$$

and the following bound on the bias:

$$\|\rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|^2 \le \kappa \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|_2^2 + \zeta \tag{87}$$

then the iteration:

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \alpha_t \boldsymbol{\nabla}_{\boldsymbol{w}} \left[L(\boldsymbol{w}_t) + \rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\right] \\ \overline{\boldsymbol{f}}_{t+1} &= (1-\beta)\overline{\boldsymbol{f}}_t + \beta \left(\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\right)^2 \end{aligned} \tag{88}$$

converges to a neighborhood of a stationary point with $\|\boldsymbol{\nabla} L(\boldsymbol{w})\|_2^2 = \mathcal{O}(\zeta)$*.*

Proof. By the Lipschitz continuity of the objective function we have the quadratic bound:

$$L(\boldsymbol{y}) \le L(\boldsymbol{x}) + \langle \boldsymbol{\nabla}_{\boldsymbol{w}} L(x), \boldsymbol{y} - \boldsymbol{x} \rangle + \frac{C}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2 \tag{89}$$

By the quadratic upper bound, the iterates generated by the algorithm satisfy:

$$L(\boldsymbol{w}_{t+1}) - L(\boldsymbol{w}_t) \le -\alpha_t \langle \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t), g(\boldsymbol{w}_k, \boldsymbol{\xi}_k) \rangle + \frac{1}{2}\alpha_t^2 C \|g(\boldsymbol{w}_k, \boldsymbol{\xi}_k)\|_2^2 \tag{90}$$

Taking expectations and applying the variance bound we have:

$$\begin{aligned} \mathbb{E}L(\boldsymbol{w}_{t+1}) - L(\boldsymbol{w}_t) &\le -\alpha_t \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 - \alpha_t \rho \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)^T \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t) + \frac{1}{2}\alpha_t^2 C \mathbb{E}\left[\|g(\boldsymbol{w}_k, \boldsymbol{\xi}_k)\|_2^2\right] \\ &= -\alpha_t \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 - \alpha_t \rho \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)^T \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t) + \frac{1}{2}\alpha_t^2 C \left[M + M_G \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(x) + \rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|_2^2\right] \\ &= -\alpha_t \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 - \alpha_t (1 - \alpha C M_G) \rho \boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)^T \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t) + \frac{1}{2}\alpha_t^2 C M + \frac{1}{2}\alpha_t^2 C M_G \left(\|\boldsymbol{\nabla}_{\boldsymbol{w}} L(x)\|_2^2 + \rho \|\boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|_2^2\right) \end{aligned} \tag{91}$$

So that, choosing $\alpha_t < \frac{1}{C M_G}$ and applying the bound on $\|\boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|$ we have:

$$\begin{aligned} \mathbb{E}L(\boldsymbol{w}_{t+1}) - L(\boldsymbol{w}_t) &\le -\frac{1}{2}\alpha_t \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 + \frac{1}{2}\alpha_t^2 C M + \frac{1}{2}\alpha_t \|\rho \boldsymbol{\nabla}_{\boldsymbol{w}} T(\boldsymbol{w}_t)\|_2^2 \\ &\le -\frac{1}{2}\alpha_t (1 - \kappa) \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 + \frac{1}{2}\alpha_t^2 C M + \frac{\alpha_t}{2}\zeta \end{aligned} \tag{92}$$

Taking the total expectation, for a fixed α, we then have:

$$L_{inf} - L(\boldsymbol{w}_1) \le \mathbb{E}\left[L(\boldsymbol{w}_{K+1})\right] - L(\boldsymbol{w}_1) \le -\frac{1}{2}\alpha(1 - \kappa) \sum_{t=1}^{K} \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 + \frac{1}{2} K \alpha^2 C M + \frac{K\alpha}{2}\zeta \tag{93}$$

Finally, we have that:

$$\frac{1}{K}\sum_{t=1}^{K} \|\boldsymbol{\nabla}_{\boldsymbol{w}} L(\boldsymbol{w}_t)\|^2 = \frac{\alpha C M}{1 - \kappa} + 2\frac{F(\boldsymbol{w}_1) - F_{inf}}{K\alpha(1 - \kappa)} \xrightarrow{K \to \infty} \frac{\alpha C M}{1 - \kappa} + \frac{\zeta}{1 - \kappa} \tag{94}$$

$\square$

References

Ahmad Ajalloeian and Sebastian U. Stich. On the convergence of SGD with biased gradients, 2021.

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

T. W. Anderson. *An introduction to multivariate statistical analysis.* Wiley Series in Probability and Statistics, 2003.

Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 639–668. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/andriushchenko22a.html`.

Christina Baek, J Zico Kolter, and Aditi Raghunathan. Why is SAM robust to label noise? In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=3aZCPl3ZvR`.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32): 15849–15854, July 2019. ISSN 1091-6490. doi: 10.1073/pnas.1903070116. URL `http://dx.doi.org/10.1073/pnas.1903070116`.

P. G. Bissiri, C. C. Holmes, and S. G. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):1103–1130, feb 2016. doi: 10.1111/rssb.12158. URL `https://doi.org/10.1111%2Frssb.12158`.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. 2016. doi: 10.48550/ARXIV.1606.04838. URL `https://arxiv.org/abs/1606.04838`.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer T. Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. *CoRR*, abs/1611.01838, 2016. URL `http://arxiv.org/abs/1611.01838`.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina N. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL 2019*, 2019. URL `https://arxiv.org/abs/1905.10044`.

Y Cooper. The loss landscape of overparameterized neural networks. *arXiv e-prints*, art. arXiv:1804.10200, April 2018. doi: 10.48550/arXiv.1804.10200.

Alicia Curth, Alan Jeffares, and Mihaela van der Schaar. A u-turn on double descent: Rethinking parameter counting in statistical learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=O0Lz8XZT2b`.

F. Dangel, F. Kunstner, and P. Hennig. BackPACK: Packing more into backprop. In *8th International Conference on Learning Representations (ICLR)*, April 2020. URL `https://openreview.net/forum?id=BJlrF24twB`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. *CoRR*, abs/1703.04933, 2017. URL `http://arxiv.org/abs/1703.04933`.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL `http://jmlr.org/papers/v12/duchi11a.html`.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. 2017. doi: 10.48550/ARXIV.1703.11008. URL `https://arxiv.org/abs/1703.11008`.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *CoRR*, abs/2010.01412, 2020. URL `https://arxiv.org/abs/2010.01412`.

A. Gray and T. J. Willmore. Mean-value theorems for riemannian manifolds. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics*, 92(3–4):343–364, 1982. doi: 10.1017/S0308210500032571.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *The Annals of Statistics*, 50(2):949 – 986, 2022. doi: 10.1214/21-AOS2133. URL `https://doi.org/10.1214/21-AOS2133`.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pp. 5–13, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916115. doi: 10.1145/168304.168306. URL `https://doi.org/10.1145/168304.168306`.

Sepp Hochreiter and Jürgen Schmidhuber. Flat Minima. *Neural Computation*, 9(1):1–42, 01 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.1.1. URL `https://doi.org/10.1162/neco.1997.9.1.1`.

Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 448–456. JMLR.org, 2015.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax, 2016. URL `https://arxiv.org/abs/1611.01144`.

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. *CoRR*, abs/1609.04836, 2016. URL `http://arxiv.org/abs/1609.04836`.

Mohammad Emtiyaz Khan and Håvard Rue. The Bayesian learning rule, 2021. URL `https://arxiv.org/abs/2107.04562`.

Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. 2018. doi: 10.48550/ARXIV.1806.04854. URL `https://arxiv.org/abs/1806.04854`.

Minyoung Kim, Da Li, Shell X Hu, and Timothy Hospedales. Fisher SAM: Information geometry and sharpness aware minimisation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11148–11161. PMLR, 17–23 Jul 2022a. URL `https://proceedings.mlr.press/v162/kim22f.html`.

Minyoung Kim, Da Li, Shell Xu Hu, and Timothy M. Hospedales. Fisher SAM: Information geometry and sharpness aware minimisation. 2022b. doi: 10.48550/ARXIV.2206.04920. URL `https://arxiv.org/abs/2206.04920`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL `https://arxiv.org/abs/1412.6980`.

Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. Generalized variational inference: Three arguments for deriving new posteriors, 2019. URL `https://arxiv.org/abs/1904.02063`.

Agustinus Kristiadi, Felix Dangel, and Philipp Hennig. The geometry of neural nets' parameter spaces under reparametrization, 2023. URL `https://arxiv.org/abs/2302.07384`.

Frederik Kunstner, Lukas Balles, and Philipp Hennig. Limitations of the empirical Fisher approximation. *CoRR*, abs/1905.12558, 2019. URL `http://arxiv.org/abs/1905.12558`.

Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks, 2021.

John Langford and Rich Caruana. (Not) bounding the true error. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL `https://proceedings.neurips.cc/paper/2001/file/98c7242894844ecd6ec94af67ac8247d-Paper.pdf`.

J.M. Lee. *Introduction to Riemannian Manifolds*. Graduate Texts in Mathematics. Springer International Publishing, 2019. ISBN 9783319917542. URL `https://books.google.co.uk/books?id=UIPltQEACAAJ`.

Dawei Li, Tian Ding, and Ruoyu Sun. Over-parameterized deep neural networks have no strict local minima for any continuous activations. *ArXiv*, abs/1812.11039, 2018. URL `https://api.semanticscholar.org/CorpusID:57189277`.

Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116, 2022. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2021.12.009. URL `https://www.sciencedirect.com/science/article/pii/S106352032100110X`. Special Issue on Harmonic Analysis and Machine Learning.

Brice Loustau. Mean value properties for harmonic functions on riemannian manifolds, Aug 2015. URL `https://cuhkmath.wordpress.com/2015/08/14/mean-value-theorems-for-harmonic-functions-on-riemannian-manifolds/`.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020. URL `http://jmlr.org/papers/v21/17-678.html`.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 2408–2417. JMLR.org, 2015.

James Martens and Ilya Sutskever. *Training Deep and Recurrent Networks with Hessian-Free Optimization*, pp. 479–535. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi: 10.1007/978-3-642-35289-8_27. URL `https://doi.org/10.1007/978-3-642-35289-8_27`.

David A. McAllester. Some PAC-Bayesian theorems. 1999. URL `https://doi.org/10.1023/A:1007618624809`.

Thomas Möllenhoff and Mohammad Emtiyaz Khan. SAM as an optimal relaxation of bayes. *ArXiv*, abs/2210.01620, 2022. URL `https://api.semanticscholar.org/CorpusID:252693360`.

Mohammad Taher Pilehvar and José Camacho-Collados. Wic: 10, 000 example pairs for evaluating context-sensitive representations. *CoRR*, abs/1808.09121, 2018. URL `http://arxiv.org/abs/1808.09121`.

Levent Sagun, Utku Evci, V. Ugur Güney, Yann N. Dauphin, and Léon Bottou. Empirical analysis of the Hessian of over-parameterized neural networks. *CoRR*, abs/1706.04454, 2017. URL `http://arxiv.org/abs/1706.04454`.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks, 2016.

Sihyeon Seong, Yegang Lee, Youngwook Kee, Dongyoon Han, and Junmo Kim. Towards flatter loss surface via nonmonotonic learning rate scheduling. In *Conference on Uncertainty in Artificial Intelligence*, 2018. URL `https://api.semanticscholar.org/CorpusID:54065338`.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems, 2020.

Ming-Wei Wei and David Schwab. Implicit regularization of SGD via thermophoresis. In *Advances in Neural Information Processing Systems, Machine Learning for Physical Sciences Workshop*, 2020.

Luke Wood, Zhenyu Tan, Ian Stenbit, Jonathan Bischof, Scott Zhu, François Chollet, Divyashree Sreepathihalli, Ramesh Sampath, et al. Kerascv. `https://github.com/keras-team/keras-cv`, 2022.

Zhewei Yao, Amir Gholami, Sheng Shen, Kurt Keutzer, and Michael W. Mahoney. ADAHESSIAN: an adaptive second order optimizer for machine learning. *CoRR*, abs/2006.00719, 2020. URL `https://arxiv.org/abs/2006.00719`.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016. URL `http://arxiv.org/abs/1611.03530`.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger B. Grosse. Noisy natural gradient as variational inference. *CoRR*, abs/1712.02390, 2017. URL `http://arxiv.org/abs/1712.02390`.