Limitations of the Empirical Fisher Approximation

Frederik Kunstner[1,2,3] first.last@gmail.com
Lukas Balles[2,3] lballes@tue.mpg.de
Philipp Hennig[2,3] ph@tue.mpg.de

École Polytechnique Fédérale de Lausanne (EPFL), Switzerland[1]
University of Tübingen, Germany[2]
Max Planck Institute for Intelligent Systems, Tübingen, Germany[3]

Abstract

Natural gradient descent, which preconditions a gradient descent update with the Fisher information matrix of the underlying statistical model, is a way to capture partial second-order information. Several highly visible works have advocated an approximation known as the empirical Fisher, drawing connections between approximate second-order methods and heuristics like Adam. We dispute this argument by showing that the empirical Fisher—unlike the Fisher—does not generally capture second-order information. We further argue that the conditions under which the empirical Fisher approaches the Fisher (and the Hessian) are unlikely to be met in practice, and that, even on simple optimization problems, the pathologies of the empirical Fisher can have undesirable effects.

1 Introduction

Consider a supervised machine learning problem of predicting outputs $y \in \mathbb{Y}$ from inputs $x \in \mathbb{X}$. We assume a probabilistic model for the conditional distribution of the form $p_\theta(y|x) = p(y|f(x,\theta))$, where $p(y|\cdot)$ is an exponential family with natural parameters in $\mathbb{F}$ and $f\colon \mathbb{X}\times\mathbb{R}^D \to \mathbb{F}$ is a prediction function parameterized by $\theta \in \mathbb{R}^D$. Given N iid training samples $(x_n, y_n)_{n=1}^N$, we want to minimize

$$\mathcal{L}(\theta) := -\sum_n \log p_\theta(y_n|x_n) = -\sum_n \log p(y_n|f(x_n,\theta)). \tag{1}$$

This framework covers common scenarios such as least-squares regression ($\mathbb{Y} = \mathbb{F} = \mathbb{R}$ and $p(y|f) = \mathcal{N}(y; f, \sigma^2)$ with fixed σ^2) or C-class classification with cross-entropy loss ($\mathbb{Y} = \{1,\dots,C\}$, $\mathbb{F} = \mathbb{R}^C$ and $p(y=c|f) = \exp(f_c)/\sum_i \exp(f_i)$) with an arbitrary prediction function f.

Eq. (1) can be minimized by gradient descent, which updates $\theta_{t+1} = \theta_t - \gamma_t \nabla\mathcal{L}(\theta_t)$ with step size $\gamma_t \in \mathbb{R}$. This update can be preconditioned, $\theta_{t+1} = \theta_t - \gamma_t B_t^{-1}\nabla\mathcal{L}(\theta_t)$, with a matrix B_t that incorporates additional information such as local curvature. Choosing B_t to be the Hessian yields Newton's method, but its computation is often computationally burdensome and might not even be desirable for non-convex problems. A prominent variant in machine learning is *natural gradient descent* [NGD; Amari, 1998], which preconditions with the Fisher information matrix (or simply "Fisher") of the probabilistic model,

$$\mathrm{F}(\theta) := \sum_n \mathbb{E}_{p_\theta(y|x_n)}\left[\nabla_\theta \log p_\theta(y|x_n)\, \nabla_\theta \log p_\theta(y|x_n)^\top\right]. \tag{2}$$

This adapts to the *information geometry* of the model. While this motivation is conceptually distinct from approximating the Hessian, the Fisher coincides with a generalized Gauss-Newton [GGN; Schraudolph, 2002] approximation of the Hessian for the problems presented here. We discuss this in detail in Section 2. This gives NGD theoretical grounding as an approximate second-order method.

A number of recent works in machine learning have relied on a certain approximation of the Fisher, which is often called the *empirical Fisher (EF)* and is defined as

$$\widetilde{\mathrm{F}}(\theta) := \sum_n \nabla_\theta \log p_\theta(y_n|x_n)\, \nabla_\theta \log p_\theta(y_n|x_n)^\top. \tag{3}$$

Preprint. Under review. Code available at github.com/fkunstner/limitations-empirical-fisher.

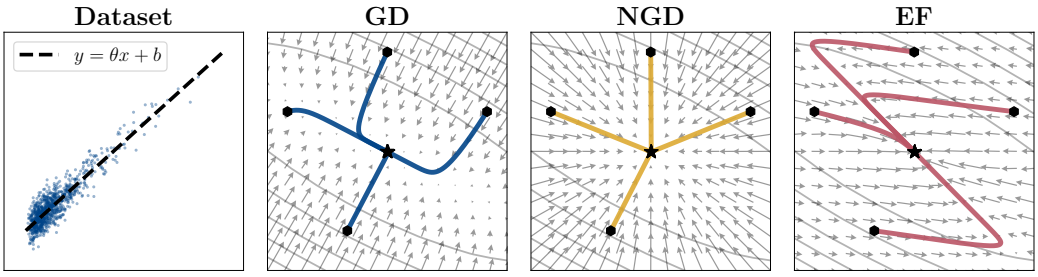

Figure 1: Fisher vs. empirical Fisher as preconditioners for linear least-squares regression on the data shown in the left-most panel. The second plot shows the gradient vector field of the (quadratic) loss function and sample trajectories for gradient descent. The remaining plots depict the vector fields of the natural gradient and the "EF-preconditioned" gradient, respectively. NGD successfully adapts to the curvature whereas preconditioning with the empirical Fisher results in a distorted gradient field.

At first glance, this approximation is merely replacing the expectation over y in Eq. (2) with a sample y_n. However, y_n is a training label and *not* a sample from the model's predictive distribution $p_\theta(y|x_n)$. Therefore, and contrary to what its name suggests, the empirical Fisher is *not* an empirical (i.e. Monte Carlo) estimate of the Fisher. Due to the unclear relationship between the model distribution and the data distribution, the theoretical grounding of the EF approximation is dubious. Despite that, the empirical Fisher approximation has seen widespread adoption, possibly because it is convenient to compute as a simple sum of outer products of individual gradients. We will provide a survey of papers using the EF in Section 1.3.

1.1 Motivation

The main purpose of this work is to provide **a detailed critical discussion of the empirical Fisher approximation.** While the discrepancy between the EF and the Fisher has been mentioned in the literature before [Pascanu and Bengio, 2014, Martens, 2014], we see the need for a detailed elaboration of the subtleties of this important issue. The intricacies of the relationship between the empirical Fisher and the Fisher remain opaque from the current literature. Not all authors using the EF seem to be fully aware of the heuristic nature of this approximation and overlook its shortcomings, which can be seen clearly even on simple linear regression problems, see Fig. 1.

The ubiquity of the EF approximation reaches the extent that the EF is sometimes just called the Fisher [e.g., Chaudhari et al., 2017, Wen et al., 2019]. Possibly as a result of this, there are examples of algorithms involving the Fisher, such as Elastic Weight Consolidation [EWC; Kirkpatrick et al., 2017] and KFAC [Martens and Grosse, 2015], which have been re-implemented by third parties using the empirical Fisher. Interestingly, there is also at least one example of an algorithm that was originally developed using the empirical Fisher and later found to work better with the Fisher [Wierstra et al., 2008, Sun et al., 2009]. As the empirical Fisher is now used beyond preconditioning, for example as an approximation of the Hessian in empirical works studying properties of neural network training objectives [Chaudhari et al., 2017, Jastrzebski et al., 2018], the pathologies of the EF approximation may lead the community to subtly erroneous conclusions; an arguably more worrysome outcome than a suboptimal preconditioner.

The poor theoretical grounding stands in stark contrast to the practical success that EF-based methods have seen. This paper is in no way meant to negate these practical advances but rather points out that the existing justifications for the EF approximation are insufficient and do not stand the test of simple examples. This indicates that there are effects at play that currently elude our understanding, which is not only unsatisfying, but might also prevent advancement of these methods. We hope that this paper helps spark interest in understanding these effects; our final chapter explores a possible direction.

1.2 Overview and contributions

We discuss related work in Section 1.3 and provide a short but complete overview of generalized Gauss-Newton and natural gradient in Section 2. We then proceed to the main contribution in Section 3, a critical discussion of the *specific arguments* used to advocate the empirical Fisher

approximation. A principal conclusion is that, while the EF follows the formal definition of a generalized Gauss-Newton matrix, it is not guaranteed to capture any useful second-order information. We propose a clarifying amendment to the definition of a GGN. Furthermore, while there are conditions under which the EF approaches the true Fisher, we argue that these are unlikely to be met in practice. We illustrate that using the EF in such cases can lead to highly undesirable effects; Fig. 1 shows a first example.

This raises the question: Why are EF-based methods empirically successful? In Section 4, we point to an alternative explanation of EF-preconditioning as adapting to gradient noise in *stochastic* optimization, instead of adapting to curvature.

1.3 Related work

The generalized Gauss-Newton [Schraudolph, 2002] and natural gradient descent [Amari, 1998] methods have inspired a line of work on approximate second-order optimization [Martens, 2010, Botev et al., 2017, Park et al., 2000, Pascanu and Bengio, 2014, Ollivier, 2015]. A successful example in modern deep learning is the KFAC algorithm [Martens and Grosse, 2015], which uses a computationally efficient structural approximation to the Fisher.

Numerous papers have relied on the empirical Fisher approximation for preconditioning and other purposes. Our critical discussion is in no way intended as an invalidation of these works. All of them provide important insights and the use of the empirical Fisher is usually not essential to the main contribution. However, there is a certain degree of vagueness regarding the relationship between the Fisher, the EF, Gauss-Newton matrices and the Hessian. Oftentimes, only limited attention is devoted to possible implications of the empirical Fisher approximation.

The most prominent example of preconditioning with the EF is Adam, which uses a moving average of squared gradients as "an approximation to the diagonal of the Fisher information matrix" [Kingma and Ba, 2015]. The EF has been used in the context of variational inference by various authors [Graves, 2011, Zhang et al., 2018, Salas et al., 2018, Khan et al., 2018, Mishkin et al., 2018], some of which have drawn further connections between NGD and Adam. There are also several works building upon KFAC which substitute the EF for the Fisher [George et al., 2018, Osawa et al., 2018].

The empirical Fisher has also been used as an approximation of the Hessian for other purposes than preconditioning. Chaudhari et al. [2017] use it to investigate curvature properties of deep learning training objectives. It has also been employed to explain certain characteristics of SGD [Zhu et al., 2018, Jastrzebski et al., 2018] or as a diagnostic tool during training [Liao et al., 2018].

Le Roux et al. [2007] and Le Roux and Fitzgibbon [2010] have considered the empirical Fisher in its interpretation as the (non-central) covariance matrix of stochastic gradients. While they refer to their method as "Online Natural Gradient", their goal is explicitly to adapt the update to the *stochasticity* of the gradient estimate, *not to curvature*. We will come back to this perspective in Section 4.

Before moving on, we want to re-emphasize that other authors have previously raised concerns about the empirical Fisher approximation [e.g., Pascanu and Bengio, 2014, Martens, 2014]. This paper is meant as a detailed elaboration of this known but subtle issue, with novel results and insights.

2 Generalized Gauss-Newton and natural gradient descent

This section briefly introduces the generalized Gauss-Newton method and natural gradient descent, summarizes the known relationship between the GGN and the Fisher, and explains in which sense they approximate the Hessian. Details can be found in the appendix.

2.1 Generalized Gauss-Newton

The original Gauss-Newton algorithm is an approximation to Newton's method for nonlinear least squares problems, $\mathcal{L}(\theta) = \frac{1}{2}\sum_n (f(x_n, \theta) - y_n)^2$. By the chain rule, the Hessian can be written as

$$\nabla^2 \mathcal{L}(\theta) = \underbrace{\sum_n \nabla_\theta f(x_n, \theta) \nabla_\theta f(x_n, \theta)^\top}_{:= G(\theta)} + \underbrace{\sum_n r_n \nabla_\theta^2 f(x_n, \theta)}_{:= R(\theta)} \tag{4}$$

where $r_n = f(x_n, \theta) - y_n$ are the residuals. The first part, $G(\theta)$, is the Gauss-Newton matrix. For small residuals, $R(\theta)$ will be small and $G(\theta)$ will approximate the Hessian. In particular, when the model perfectly fits the data, the Gauss-Newton is equal to the Hessian.

Schraudolph [2002] generalized this idea to objectives of the form $\mathcal{L}(\theta) = \sum_n a_n(b_n(\theta))$, with $b_n \colon \mathbb{R}^D \to \mathbb{R}^M$ and $a_n \colon \mathbb{R}^M \to \mathbb{R}$, for which the Hessian can be written as[1]

$$\nabla^2 \mathcal{L}(\theta) = \sum_n (\mathrm{J}_\theta b_n(\theta))^\top \ \nabla_b^2 a_n(b_n(\theta)) \ (\mathrm{J}_\theta b_n(\theta)) + \sum_{n,m} [\nabla_b a_n(b_n(\theta))]_m^2 \nabla_\theta^2 b_n^{(m)}(\theta). \quad (5)$$

The generalized Gauss-Newton matrix (GGN) is defined as the part of the Hessian that ignores the second-order information of b_n,

$$G(\theta) := \sum_n [\mathrm{J}_\theta b_n(\theta)]^\top \ \nabla_b^2 a_n(b_n(\theta)) \ [\mathrm{J}_\theta b_n(\theta)]. \quad (6)$$

If a_n is convex, as is customary, the GGN is positive (semi-)definite even if the Hessian itself is not, making it a popular curvature matrix in non-convex problems such as neural network training. The GGN is ambiguous as it crucially depends on the "split" given by a_n and b_n. As an example, consider the two following possible splits for the least-squares problem from above:

$$a_n(b) = \tfrac{1}{2}(b - y_n)^2, \ b_n(\theta) = f(x_n, \theta), \quad \text{or} \quad a_n(b) = \tfrac{1}{2}(f(x_n, b) - y_n)^2, \ b_n(\theta) = \theta. \quad (7)$$

The first recovers the classical Gauss-Newton, while in the second case, the GGN equals the Hessian. While this is an extreme example, the split will be important for our discussion.

2.2 Natural gradient descent

Gradient descent follows the direction of "steepest descent", the negative gradient. But the definition of *steepest* depends on a notion of distance and the gradient is defined with respect to the Euclidean distance. The natural gradient is a concept from information geometry [Amari, 1998] and applies when the gradient is taken w.r.t. the parameters θ of a probability distribution p_θ. Instead of measuring the distance between parameters θ and θ' with the Euclidean distance, we use the Kullback–Leibler (KL) divergence between the distributions p_θ and $p_{\theta'}$. The resulting steepest descent direction is the negative gradient preconditioned with the Hessian of the KL divergence, which is exactly the *Fisher information matrix* of p_θ,

$$\mathrm{F}(\theta) := \mathbb{E}_{p_\theta(z)} \left[\nabla_\theta \log p_\theta(z) \nabla_\theta \log p_\theta(z)^T \right] = \mathbb{E}_{p_\theta(z)} \left[-\nabla_\theta^2 \log_\theta p(z) \right]. \quad (8)$$

The second equality may seem counterintuitive; it crucially depends on the expectation being taken over the model distribution at θ, see Appendix A. This equivalence highlights the relationship of the Fisher to the Hessian. In our setting, where we only model the conditional distribution $p_\theta(y|x)$, the Fisher is given by Eq. (2).

2.3 Connections between the Fisher, the GGN and the Hessian

While NGD is not explicitly motivated as an approximate second-order method, the following result, noted by several authors,[2] shows that the Fisher captures partial curvature information about the problem defined in Eq. (1).

Proposition 1 (Martens [2014], §9.2)**.** *If $p(y|f)$ is an exponential family distribution with natural parameters f, then the Fisher information matrix coincides with the GGN of Eq.* (1) *using the split*

$$a_n(b) = -\log p(y_n|b), \qquad b_n(\theta) = f(x_n, \theta), \quad (9)$$

and reads $\mathrm{F}(\theta) = G(\theta) = -\sum_n [\mathrm{J}_\theta f(x_n, \theta)]^\top \nabla_f^2 \log p(y_n|f(x_n, \theta)) \, [\mathrm{J}_\theta f(x_n, \theta)]$.

For completeness, a proof can be found in Appendix A. The key insight is that $\nabla_f^2 \log p(y|f)$ does not depend on y for exponential families. One can see Eq. (9) as the "canonical" split, since it matches the classical Gauss-Newton for the probabilistic interpretation of least-squares. From now on, when referencing "the GGN" without further specification, we mean this particular split.

[1] $J_\theta b_n(\theta) \in \mathbb{R}^{M \times D}$ is the Jacobian of b_n; we use the shortened notation $\nabla_b^2 a_n(b_n(\theta)) := \nabla_b^2 a_n(b)|_{b=b_n(\theta)}$; $[\cdot]_m$ selects the m-th component of a vector; and $b_n^{(m)}$ denotes the m-th component function of b_n.

[2] Heskes [2000] showed this for regression with squared loss, Pascanu and Bengio [2014] for classification with cross-entropy loss, and Martens [2014] for general exponential families. However, this has been known earlier in the statistics literature in the context of "Fisher Scoring" (see Wang [2010] for a review).

The GGN, and under the assumptions of Proposition 1 also the Fisher, are well-justified approximations of the Hessian and we can bound their approximation error in terms of the (generalized) residuals, mirroring the motivation behind the classical Gauss-Newton.

Proposition 2. *Let $\mathcal{L}(\theta)$ be defined as in Eq.* (1) *with $\mathbb{F} = \mathbb{R}^M$. Denote by $f_n^{(m)}$ the m-th component function of $f(x_n, \cdot)\colon \mathbb{R}^D \to \mathbb{R}^M$ and assume each $f_n^{(m)}$ is β-smooth. Let $G(\theta)$ be the GGN (Eq. 6). Then,*

$$\left\|\nabla^2 \mathcal{L}(\theta) - G(\theta)\right\|_2^2 \leq r(\theta)\beta, \tag{10}$$

where $r(\theta) = \sum_{n=1}^N \|\nabla_f \log p(y_n|f(x_n,\theta))\|_1$ and $\|\cdot\|_2$ denotes the spectral norm.

The approximation improves as the residuals in $r(\theta)$ diminish, and is exact if the data is perfectly fit.

3 Critical discussion of the empirical Fisher

Two arguments have been put forward to advocate the empirical Fisher approximation. Firstly, it has been argued that the EF follows the definition of a generalized Gauss-Newton matrix, making it an approximate curvature matrix in its own right. We examine this relation in §3.1 and show that, while technically correct, it does not entail the approximation guarantee usually associated with the GGN.

Secondly, a popular argument is that the EF approaches the Fisher at a minimum if the model "is a good fit for the data". We discuss this argument in §3.2 and point out that it requires strong additional assumptions, which are unlikely to be met in practical scenarios. In addition, this argument only applies close to a minimum, which calls into question the usefulness of the empirical Fisher as a preconditioner. We discuss this in §3.3, showing that preconditioning with the EF leads to adverse effects on the scaling and the direction of the updates far from an optimum.

We use simple examples to illustrate our arguments. We want to emphasize that, as these are *counter-examples* to arguments found in the existing literature, they are designed to be as simple as possible, and deliberately do not involve intricate state-of-the art models that would complicate analysis. On a related note, while contemporary machine learning often relies on *stochastic* optimization, we restrict our considerations to the deterministic (full-batch) setting to focus on the adaptation to curvature.

3.1 The empirical Fisher as a generalized Gauss-Newton matrix

Bottou et al. [2018] point out that the EF matches the construction of a GGN (Eq. 6) using the split

$$a_n(b) = -\log b, \qquad\qquad b_n(\theta) = p(y_n|f(x_n,\theta)). \tag{11}$$

Although technically correct[3], we argue that this split does not provide a reasonable approximation.

For example, consider a least-squares problem which corresponds to the log-likelihood $\log p(y|f) = \log \exp[-\frac{1}{2}(y-f)^2]$. In this case, Eq. (11) splits the identity function, $\log\exp(\cdot)$, and takes into account the curvature from the $\log$ while ignoring that of $\exp$. This questionable split runs counter to the basic motivation behind the classical Gauss-Newton matrix, that small residuals lead to a good approximation to the Hessian: The empirical Fisher

$$\widetilde{\mathrm{F}}(\theta) = \textstyle\sum_n \nabla_\theta \log p_\theta(y_n|x_n)\, \nabla_\theta \log p_\theta(y_n|x_n)^\top = \sum_n r_n^2\, \nabla_\theta f(x_n,\theta)\, \nabla_\theta f(x_n,\theta)^\top, \tag{12}$$

approaches zero as the residuals $r_n = y_n - f(x_n,\theta)$ become small. In that same limit, the Fisher $F(\theta) = \sum_n \nabla f(x_n,\theta) \nabla f(x_n,\theta)^\top$ does approach the Hessian, which we recall from Eq. (4) to be given by $\nabla^2 \mathcal{L}(\theta) = F(\theta) + \sum_n r_n \nabla_\theta^2 f(x_n,\theta)$. This argument generally applies for problems where we can fit all training samples such that $\nabla_\theta \log p_\theta(y_n|x_n) = 0$ for all n. In such cases, the EF goes to zero while the Fisher (and the corresponding GGN) approaches the Hessian (Prop. 2).

For the generalized Gauss-Newton, the role of the "residual" is played by the gradient $\nabla_b a_n(b)$; compare Equations (4) and (5). To retain the motivation behind the classical Gauss-Newton, the split should be chosen such that this gradient can in principle attain zero, in which case the residual curvature not captured by the GGN in (5) vanishes. The EF split (Eq. 11) does not satisfy this property, as $\nabla_b \log b$ can never go to zero for a probability $b \in [0,1]$. It might be desirable to amend the definition of a generalized Gauss-Newton to enforce this property (addition in **bold**):

[3] The equality can easily be verified by plugging the split (11) into the definition of the GGN (Eq. 6) and observing that $\nabla_b^2 a_n(b) = \nabla_b a_n(b)\, \nabla_b a_n(b)^\top$ as a special property of the choice $a_n(b) = -\log(b)$.

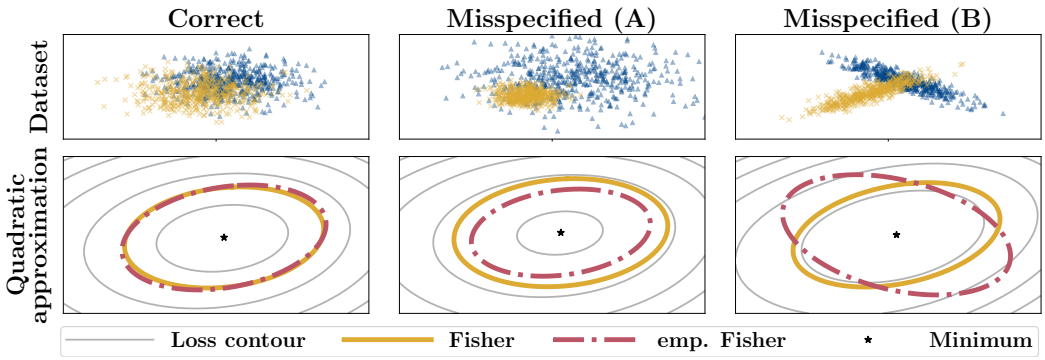

Figure 2: Quadratic approximations of the loss function using the Fisher and the empirical Fisher on a logistic regression problem. Logistic regression implicitly assumes identical class-conditional covariances [Hastie et al., 2009, §4.4.5]. The EF is a good approximation of the Fisher at the minimum if this assumption is fulfilled (left panel), but can be arbitrarily wrong if the assumption is violated, even at the minimum and with large N. Note: we achieve classification accuracies of $\geq 85\%$ in the misspecified cases compared to 73% in the well-specified case, which shows that a well-performing model is not necessarily a well-specified one.

Definition 1 (Generalized Gauss-Newton)**.** *A split $\mathcal{L}(\theta) = \sum_n a_n(b_n(\theta))$ with convex a_n, leads to a generalized Gauss-Newton matrix of $\mathcal{L}$, defined as*

$$G(\theta) = \sum_n G_n(\theta), \qquad G_n(\theta) := [\mathrm{J}_\theta b_n(\theta)]^\top \ \nabla_b^2 a_n(b_n(\theta)) \ [\mathrm{J}_\theta b_n(\theta)], \tag{13}$$

***if the split** a_n, b_n **is such that there is** $b_n^* \in \mathrm{Im}(b_n)$ **such that** $\nabla_b a_n(b)|_{b=b_n^*} = 0$.*

Under suitable smoothness conditions, a split satisfying this condition will have a meaningful error bound akin to Proposition 2. (To avoid confusion, we want to note that this condition does not assume the existence of θ^* such that $b_n(\theta^*) = b_n^*$ for all n; only that the residual gradient for each data point can, in principle, go to zero.)

3.2 The empirical Fisher near a minimum

An oft-repeated argument is that the empirical Fisher converges to the true Fisher when the model is a good fit for the data [e.g., Jastrzebski et al., 2018, Zhu et al., 2018]. Unfortunately, this is often misunderstood to simply mean "near the minimum". The above statement has to be carefully formalized and requires additional assumptions, which we detail in the following.

Assume that the training data consists of iid samples from some data generating distribution $p_{\text{true}}(x, y) = p_{\text{true}}(y|x)p_{\text{true}}(x)$. If the model is realizable, i.e., there exists a parameter setting θ_{T} such that $p_{\theta_{\text{T}}}(y|x) = p_{\text{true}}(y|x)$, then clearly by an MC sampling argument, as the number of data points N goes to infinity, $\widetilde{\mathrm{F}}(\theta_T)/N \to \mathrm{F}(\theta_T)/N$. If, additionally, the maximum likelihood estimate for N samples, $\theta_N^\star$, is consistent in the sense that $p_{\theta_N^\star}(y|x)$ converges to $p_{\text{true}}(y|x)$ as $N \to \infty$, then

$$\frac{1}{N}\widetilde{\mathrm{F}}(\theta_N^\star) \overset{N\to\infty}{\longrightarrow} \frac{1}{N}\mathrm{F}(\theta_N^\star). \tag{14}$$

That is, the empirical Fisher converges to the Fisher *at* the minimum as the number of data points grows. (Both approach the Hessian, as can be seen from the second equality in Eq. 8 and detailed in Appendix C.3.) For the EF to be a useful approximation, we thus need (i) a "correctly-specified" model in the sense of the realizability condition, and (ii) enough data to recover the true parameters.

Even under the assumption that N is sufficiently large, the model needs to be able to realize the true data distribution. This requires that the likelihood $p(y|f)$ is well-specified and that the prediction function $f(x, \theta)$ captures all relevant information. This is possible in classical statistical modeling of, say, scientific phenomena where the effect of x on y is modeled based on domain knowledge. But it is unlikely to hold when the model is only approximate, as is most often the case in machine learning. Figure 2 shows examples of model misspecification and the effect on the empirical and true Fisher.

It is possible to satisfy the realizability condition by using a very flexible prediction function $f(x, \theta)$, such as a deep network. However, "enough" data has to be seen relative to the model capacity. The

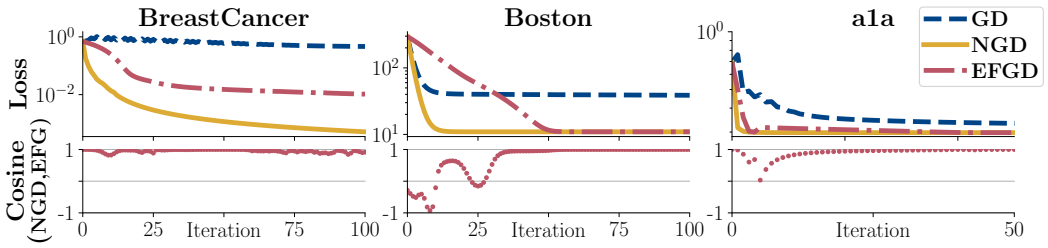

Figure 3: Fisher (NGD) vs. empirical Fisher (EFGD) as preconditioners (with damping) on linear classification (BreastCancer, a1a) and regression (Boston). While the EF *can* be a good approximation for preconditioning on some problems (e.g., a1a), it is not guaranteed to be. The second row shows the cosine similarity between the EF direction and the natural gradient, over the path taken by EFGD, showing that the EF can lead to update directions that are opposite to the natural gradient (see Boston). Even when the direction is correct, the magnitude of the steps can lead to poor performance (see BreastCancer). See Appendix E for details and additional experiments.

massively overparameterized models typically used in deep learning are able to fit the training data almost perfectly, even when regularized [Zhang et al., 2017]. In such settings, the individual gradients, and thus the EF, will be close to zero at a minimum, whereas the Hessian will generally be nonzero.

3.3 Preconditioning with the empirical Fisher far from an optimum

The relationship discussed in §3.2 only holds close to the minimum. Any similarity between $p_\theta(y|x)$ and $p_{\text{true}}(y|x)$ is *very* unlikely when θ has not been adapted to the data, for example, at the beginning of an optimization procedure. This makes the empirical Fisher a questionable preconditioner.

In fact, the empirical Fisher can cause severe, adverse distortions of the gradient field far from the optimum, as evident even on an elementary linear least-squares problem in Fig. 1. As a consequence, EF-preconditioned gradient descent compares unfavorably to NGD even on simple linear regression and classification tasks, as shown in Fig. 3. The cosine similarity plotted in Fig. 3 shows that the empirical Fisher can be arbitrarily far from the Fisher in that the two preconditioned updates point in almost opposite directions.

One particular issue is the scaling of EF-preconditioned updates. As the empirical Fisher is the sum of "squared" gradients (Eq. 3), multiplying the gradient by the inverse of the EF leads to updates of magnitude almost inversely proportional to that of the gradient, at least far from the optimum. This effect has to be counteracted by adapting the step size, which requires manual tuning and makes the selected step size dependent on the starting point; we explore this aspect further in Appendix D.

4 Variance adaptation

The previous sections have shown that, interpreted as a *curvature* matrix, the empirical Fisher is a questionable choice at best. Another perspective on the empirical Fisher is that (in contrast to the Fisher) it contains useful information to adapt to the gradient noise in *stochastic* optimization.

In stochastic gradient descent [SGD; Robbins and Monro, 1951], we sample $n \in [N]$ uniformly at random and use a stochastic gradient $g(\theta) = -N \nabla_\theta \log p_\theta(y_n|x_n)$ as an inexpensive but noisy estimate of $\nabla \mathcal{L}(\theta)$. The empirical Fisher, as a sum of outer products of individual gradients, coincides with the non-central second moment of this estimate and can be written as

$$N\widetilde{\mathrm{F}}(\theta) = \Sigma(\theta) + \nabla \mathcal{L}(\theta) \nabla \mathcal{L}(\theta)^\top, \qquad \Sigma(\theta) := \mathbf{cov}[g(\theta)]. \tag{15}$$

Gradient noise is a major hindrance to SGD and the covariance information encoded in the EF may be used to attenuate its harmful effects, e.g., by scaling back the update in high-noise directions.

A small number of works have explored this idea before. Le Roux et al. [2007] showed that the update direction $\Sigma(\theta)^{-1} g(\theta)$ maximizes the probability of decreasing in function value, while Schaul et al. [2013] proposed a diagonal rescaling based on the signal-to-noise ratio of each coordinate,

$D_{ii} := [\nabla \mathcal{L}(\theta)]_i^2 \,/\, ([\nabla \mathcal{L}(\theta)]_i^2 + \Sigma(\theta)_{ii})$. Balles and Hennig [2018] identified these factors as *optimal* in that they minimize the expected error $\mathbb{E}\left[\|Dg(\theta) - \nabla \mathcal{L}(\theta)\|_2^2\right]$ for a diagonal matrix D.

A straightforward extension of this argument to full matrices yields the variance adaptation matrix

$$M = \left(\Sigma(\theta) + \nabla \mathcal{L}(\theta)\, \nabla \mathcal{L}(\theta)^\top\right)^{-1} \nabla \mathcal{L}(\theta)\, \nabla \mathcal{L}(\theta)^\top = (N\widetilde{\mathrm{F}}(\theta))^{-1} \nabla \mathcal{L}(\theta)\, \nabla \mathcal{L}(\theta)^\top. \tag{16}$$

In that sense, preconditioning with the empirical Fisher can be understood as an adaptation to gradient noise instead of an adaptation to curvature. (Note that Eq. (16) involves a multiplication with $\nabla \mathcal{L}(\theta) \nabla \mathcal{L}(\theta)^\top$ which will counteract the poor scaling discussed in §3.3.)

This perspective on the empirical Fisher is currently not well studied. Of course, there are obvious difficulties ahead: Computing the matrix in Eq. (16) requires the evaluation of all gradients, which defeats its purpose. It is not obvious how to obtain meaningful estimates of this matrix from, say, a mini-batch of gradients that would provably attenuate the effects of gradient noise. Nevertheless, we believe that variance adaptation is a possible explanation for the practical success of existing methods using the EF, and an interesting avenue for future research. To put it bluntly: It may just be that the name "empirical Fisher" is a fateful historical misnomer, and the quantity should instead just be described as the gradient's non-central second moment.

As a final comment, it is worth pointing out that some works actually precondition with the *square-root* of the EF, the prime example being Adam. While this avoids the "inverse gradient" scaling discussed in §3.3, it further widens the conceptual gap between those methods and natural gradient. In fact, such a preconditioning effectively cancels out the gradient magnitude, which has recently been examined more closely as "sign gradient descent" [Balles and Hennig, 2018, Bernstein et al., 2018].

5 Conclusions

We offered a critical discussion of the empirical Fisher approximation, summarized as follows:

- While the EF follows the formal definition of a generalized Gauss-Newton matrix, the underlying split does not retain useful second-order information. We proposed a clarifying amendment to the definition of the GGN.
- A clear relationship between the empirical Fisher and the Fisher only exists at a minimum under strong additional assumptions: (i) a correct model and (ii) enough data relative to model capacity. These conditions are unlikely to be met in practice, especially when using overparametrized general function approximators and settling for approximate minima.
- Far from an optimum, EF preconditioning leads to update magnitudes which are inversely proportional to that of the gradient, complicating step size tuning and often leading to poor performance even for linear models.
- As a possible alternative explanation of the practical success of EF preconditioning, and an interesting avenue for future research, we have pointed to the concept of variance adaptation.

Hence, the existing arguments do not justify the empirical Fisher as a reasonable approximation to the Fisher or the Hessian. Of course, this does not rule out the existence of certain model classes for which the EF might give reasonable approximations. However, as long as we have not clearly identified and understood these cases, the true Fisher is the "safer" choice as a curvature matrix and should be preferred in virtually all cases.

Contrary to conventional wisdom, the Fisher is not inherently harder to compute than the EF. As shown by Martens and Grosse [2015], an unbiased estimate of the true Fisher can be obtained at the same computational cost as the empirical Fisher by replacing the expectation in Eq. (2) with a single sample $\tilde{y}_n$ from the model's predictive distribution $p_\theta(y|x_n)$. Even exact computation of the Fisher is feasible in many cases. We discuss computational aspects further in Appendix B. The apparent reluctance to compute the Fisher might have more to do with the current lack of convenient implementations in deep learning libraries. We believe that it is misguided—and potentially dangerous—to accept the poor theoretical grounding of the EF approximation purely for implementational convenience.

Acknowledgements

We would like to thank Matthias Bauer, Felix Dangel, Filip de Roos, Diego Fioravanti, Si Kai Lee, and Frank Schneider for their helpful comments on the manuscript. We also would like to acknowledge the constructive feedback of the anonymous reviewers on an earlier version of this paper. Frederik Kunstner would like to thank Emtiyaz Khan, Aaron Mishkin, and Didrik Nielsen for many insightful conversations. Lukas Balles kindly acknowledges the support of the International Max Planck Research School for Intelligent Systems (IMPRS-IS).

References

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Lukas Balles and Philipp Hennig. Dissecting Adam: The sign, magnitude and variance of stochastic gradients. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 413–422. PMLR, 2018.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Anima Anandkumar. signSGD: compressed optimisation for non-convex problems. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 559–568. PMLR, 2018.

Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical Gauss-Newton optimisation for deep learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 557–565. PMLR, 2017.

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

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. 2017.

Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a Kronecker-factored eigenbasis. pages 9573–9583, 2018.

Alex Graves. Practical variational inference for neural networks. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 2348–2356, 2011.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Verlag, 2009.

Tom Heskes. On "natural" learning and pruning in multilayered perceptrons. *Neural Computation*, 12(4):881–901, 2000.

Stanislaw Jastrzebski, Zac Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Amos Storkey, and Yoshua Bengio. Three factors influencing minima in SGD. In *International Conference on Artificial Neural Networks*, 2018.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001. URL http://www.scipy.org/.

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. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2616–2625. PMLR, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114 (13):3521–3526, 2017.

Nicolas Le Roux and Andrew W Fitzgibbon. A fast natural Newton method. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 623–630. Omnipress, 2010.

Nicolas Le Roux, Pierre-Antoine Manzagol, and Yoshua Bengio. Topmoumoute online natural gradient algorithm. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 849–856. Curran Associates, Inc., 2007.

Zhibin Liao, Tom Drummond, Ian Reid, and Gustavo Carneiro. Approximate Fisher information matrix to characterise the training of deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

James Martens. Deep learning via Hessian-free optimization. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 735–742. Omnipress, 2010.

James Martens. New insights and perspectives on the natural gradient method. *CoRR*, abs/1412.1193, 2014.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2408–2417. JMLR.org, 2015.

Aaron Mishkin, Frederik Kunstner, Didrik Nielsen, Mark Schmidt, and Mohammad Emtiyaz Khan. SLANG: Fast structured covariance approximations for Bayesian deep learning with natural gradient. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 6248–6258. 2018.

Yann Ollivier. Riemannian metrics for neural networks I: feedforward networks. *Information and Inference: A Journal of the IMA*, 4(2):108–153, 2015.

Kazuki Osawa, Yohei Tsuji, Yuichiro Ueno, Akira Naruse, Rio Yokota, and Satoshi Matsuoka. Second-order optimization method for large mini-batch: Training ResNet-50 on ImageNet in 35 epochs. *arXiv preprint arXiv:1811.12019, To appear at CVPR19*, 2018.

Hyeyoung Park, S-I Amari, and Kenji Fukumizu. Adaptive natural gradient learning algorithms for various stochastic models. *Neural Networks*, 13(7):755–764, 2000.

Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

Arnold Salas, Stefan Zohren, and Stephen Roberts. Practical Bayesian learning of neural networks via adaptive subgradient methods. *CoRR*, abs/1811.03679, 2018.

Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 343–351. JMLR.org, 2013.

Nicol N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural Computation*, 14(7):1723–1738, 2002.

Yi Sun, Daan Wierstra, Tom Schaul, and Juergen Schmidhuber. Efficient natural evolution strategies. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 539–546, 2009.

Yong Wang. Fisher scoring: An interpolation family and its Monte Carlo implementations. *Computational Statistics & Data Analysis*, 54(7):1744–1755, 2010.

Yeming Wen, Kevin Luk, Maxime Gazeau, Guodong Zhang, Harris Chan, and Jimmy Ba. Interplay between optimization and generalization of stochastic gradient descent with covariance noise. *arXiv preprint arXiv:1902.08234*, 2019.

Daan Wierstra, Tom Schaul, Jan Peters, and Juergen Schmidhuber. Natural evolution strategies. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 3381–3387, 2008.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Guodong Zhang, Shengyang Sun, David K. Duvenaud, and Roger B. Grosse. Noisy natural gradient as variational inference. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5847–5856. PMLR, 2018.

Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from minima and regularization effects. *arXiv preprint arXiv:1803.00195*, 2018.

Limitations of the Empirical Fisher Approximation Supplementary Material

Table of contents

§A: Details on natural gradient descent provides additional exposition on the natural gradient and the generalized Gauss-Newton; its relation to distances in probability distribution space (§A.1), a note regarding the treatment of the distribution over inputs (§A.2), the expression of the Fisher for common loss functions (§A.3) and the view of the generalized Gauss-Newton as a linearization of the model (§A.4).

§B: Computational aspects discusses the challenges of computing the empirical Fisher and the Fisher, and possible options.

§C: Additional proofs provides proof of the propositions and statements skipped in the main paper; the relation between the expected Hessian and expected outer product of gradients (Eq. 8), the equivalence between the generalized Gauss-Newton (Prop. 1), and the bound on the difference between the generalized Gauss-Newton and the Hessian (Prop. 2).

§D: Additional plots shows the experiments on different datasets.

§E: Experimental details gives the necessary details to reproduce our experiments.

A Details on natural gradient descent

We give an expanded version of the introduction to natural gradient descent provided in Section 2.2

A.1 Detailed introduction

Gradient descent minimizes some objective function by greedily updating in the "direction of steepest descent". But what, precisely, is meant by the direction of steepest descent? Consider the following definition,

$$\lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \left(\arg\min_{\delta} f(\theta + \delta) \right) \text{ s.t. } d(\theta, \theta + \delta) \leq \varepsilon, \tag{17}$$

where $d(\cdot, \cdot)$ is some distance function. This definition says that we are looking for the update step δ which minimizes f within an ε-ball around the current θ, and subsequently let the radius ε go to zero (to make δ finite, we have to divide by ε). This definition makes clear that the direction of steepest descent is intrinsically tied to the geometry which we impose on the parameter space by the definition of the distance function. If we choose $d(\theta, \theta') = \left\| \theta - \theta' \right\|_2$, the Euclidean distance, Eq. (17) reduces to the (normalized) negative gradient.

Now, assume that θ parameterizes a statistical model $p_\theta(z)$. The parameter vector θ is not the main quantity of interest; the distance between θ and θ' would be better measured in terms of distance between the distributions p_θ and $p_{\theta'}$. A canonical distance function for probability distributions is the Kullback–Leibler (KL) divergence. If we choose $d(\theta, \theta') = D_{KL}\left(p_{\theta'} \parallel p_\theta\right)$, the steepest descent direction becomes the natural gradient, $\mathrm{F}(\theta)^{-1} \nabla \mathcal{L}(\theta)$, where

$$\mathrm{F}(\theta) := \mathbb{E}_{p_\theta(z)} \left[\nabla \log p_\theta(z) \nabla \log p_\theta(z)^T \right] = \mathbb{E}_{p_\theta(z)} \left[-\nabla^2 \log p_\theta(z) \right] \tag{18}$$

is the *Fisher information matrix* of the statistical model, which arises in this context as the Hessian of the KL divergence

$$\mathrm{F}(\theta) = \left. \nabla^2_{\theta'} D_{KL}\left(p_{\theta'} \parallel p_\theta\right) \right|_{\theta' = \theta}. \tag{19}$$

A.2 What about the distribution over the inputs?

When going from Eq. (18) (Eq. 8 in the main text) to the expression for the Fisher of a conditional probability distribution $p(y|f(x,\theta))$ with samples $(x_n, y_n)_{n=1,\dots,N}$ (Eq. 2 in the main text),

$$\mathrm{F}(\theta) = \sum_n \mathbb{E}_{p_\theta(y|x_n)} \left[\nabla_\theta \log p_\theta(y|x_n) \, \nabla_\theta \log p_\theta(y|x_n)^\top \right], \tag{20}$$

we considered our statistical model to be only on the conditional distribution $y|x$ and restricted x to the *empirical distribution*, i.e., a mixture of Dirac distributions centered at the training points x_n, $n \in [N]$. An alternative is to consider a statistical model for the joint distribution (x, y) given by

$$p_\theta(x, y) = p(y|f(x,\theta)) \, p_{\text{true}}(x), \tag{21}$$

where $p_{\text{true}}(x)$ is the true distribution over x. In this case the Fisher would be

$$\mathrm{F}(\theta) = \mathbb{E}_{p_{\text{true}}(x)} \left[\mathbb{E}_{p(y|f(x,\theta))} \left[\nabla_\theta \log p_\theta(y|x) \, \nabla_\theta \log p_\theta(y|x)^\top \right] \right]. \tag{22}$$

However, since p_{true} is generally unknown (and the expectation over it intractable), this is impractical usually approximated using the empirical distribution over the inputs. In the statistic literature, this quantity is occasionally referred to as the empirical Fisher, due to the approximation of $p(x)$ by the empirical distribution but this is *not* what is referred to as the *empirical Fisher* in the main text and in the literature we cite. Going from Eq. (22) to Eq. (20) replaces the expectation over $p_{\text{true}}(x)$ with N samples from *that* distribution. In contrast to that, going from Eq. (2) to Eq. (3) replaces the expectation over $p_\theta(y|x_n)$ with a single sample y_n from a *different* distribution.

A.3 The Fisher for common loss functions

For a probabilistic conditional model of the form $p(y|f(x,\theta))$ where p is an exponential family distribution, the equivalence between the Fisher and the generalized Gauss-Newton leads to a straightforward way to compute the Fisher without expectations, as

$$\mathrm{F}(\theta) = \sum_n (\mathrm{J}_\theta f(x_n,\theta))^\top (\nabla^2 \log p(y_n|f(x_n,\theta))) (\mathrm{J}_\theta f(x_n,\theta)) = \sum_n J_n^\top H_n J_n, \tag{23}$$

where $H_n = \nabla^2 \log p(y_n|f(x_n,\theta))$ and $J_n = \mathrm{J}_\theta f(x_n,\theta)$, and H_n often has an exploitable structure.

The squared-loss used in regression, $\frac{1}{2}\sum_n \left\| y_n - f(x_n,\theta) \right\|^2$, can be cast in a probabilistic setting with a Gaussian distribution with unit variance, $p(y_n|f(x_n,\theta)) = \mathcal{N}\left(y_n; f(x_n,\theta), 1\right)$,

$$p(y_n|f(x_n,\theta)) = \exp\left(-\tfrac{1}{2} \left\| y_n - f(x_n,\theta) \right\|^2 \right).$$

The Hessian of the negative log-likelihood w.r.t. f is then simply given by

$$\nabla_f^2 - \log p(y_n|f) = \nabla_f^2 \left[-\log \exp\left(-\tfrac{1}{2} \|y_n - f\|^2 \right) \right] = \nabla_f^2 \left[\tfrac{1}{2} \|y_n - f\|^2 \right] = 1. \tag{24}$$

The cross-entropy loss used in C-class classification can be cast as an exponential family distribution by using the softmax function on the mapping $f(x_n,\theta)$,

$$p(y_n = c|f(x_n,\theta)) = [\text{softmax}(f)]_c = \tfrac{e^{f_c}}{\sum_i e^{f_i}} = \pi_c,$$

Taking the Hessian of the negative log-likelihood w.r.t. f, we can simply write the Hessian as

$$\nabla_f^2 (-\log p(y = c|f)) = \nabla_f^2 \left[-f_c + \log\left(\textstyle\sum_i e^{f_i} \right) \right] = \nabla_f^2 \left[\log\left(\textstyle\sum_i e^{f_i} \right) \right].$$

Checking the partial derivatives individually, we get that

$$\frac{\partial^2}{\partial f_i^2} \log\left(\sum_c e^{f_c} \right) = \frac{e^{f_i}}{(\sum_c e^{f_c})} - \frac{{e^{f_i}}^2}{(\sum_c e^{f_c})^2}, \quad \text{and} \quad \frac{\partial^2}{\partial f_i \partial f_j} \log\left(\sum_c e^{f_c} \right) = -\frac{e^{f_i} e^{f_j}}{(\sum_c e^{f_c})^2}.$$

Or, using π as the vector of predicted probabilities

$$\nabla_f^2 (-\log p(y|f)) = \mathrm{diag}(\pi) - \pi\pi^\top. \tag{25}$$

A.4 The generalized Gauss-Newton as a linear approximation of the model

In §2.1, we mentioned that the generalized Gauss-Newton with a split $\mathcal{L}(\theta) = \sum_n a_n(b_n(\theta))$ can be interpreted as an approximation of $\mathcal{L}$ where the second-order information of a_n is kept but the second-order information of b_n is ignored as b_n is approximated by a linear function. To see this connection, note that if b_n is a linear function, then the Hessian and the GGN are equal as the Hessian of b_n w.r.t. to θ is zero,

$$\nabla^2 \mathcal{L}(\theta) = \sum_n (\mathrm{J}_\theta b_n(\theta))^\top \ \nabla_b^2 a_n(b_n(\theta)) \ (\mathrm{J}_\theta b_n(\theta)) + \sum_{n,m} [\nabla_b a_n(b_n(\theta))]_m^2 \underbrace{\nabla_\theta^2 b_n^{(m)}(\theta)}_{=0}. \quad (26)$$

Let us write $\bar{b}_n^{(\theta)}(\theta') := b_n(\theta) + \mathrm{J}_\theta b_n(\theta)(\theta' - \theta)$ for the first-order Taylor approximation of b_n *around* θ, which is now a function of θ'. We now approximate $\mathcal{L}(\theta')$, in the vicinity of θ, by replacing b_n by its linear approximation $\bar{b}_n^{(\theta)}(\theta')$. The generalized Gauss-Newton is the Hessian of this approximation, evaluated at $\theta' = \theta$,

$$G(\theta) = \nabla_{\theta'}^2 \sum_n a_n(\bar{b}_n^{(\theta)}(\theta'))\Big|_{\theta'=\theta} = \sum_n (\mathrm{J}_\theta b_n(\theta))^\top \ \nabla_b^2 a_n(b_n(\theta)) \ (\mathrm{J}_\theta b_n(\theta)) \quad (27)$$

B Computational aspects

The empirical Fisher approximation is often motivated as an easier-to-compute alternative to the Fisher. While there is *some* merit to this argument, we argued in the main text that the empirical Fisher computes a wrong quantity. However, it is possible to compute an actual approximation to the Fisher at the same computational complexity and using a very similar implementation: just sample one output $\tilde{y}_n$ from the model distribution $p(y|f(x_n, \theta))$ for each input x_n and compute the outer product of the gradients

$$\sum_n \nabla \log p(\tilde{y}_n|f(x_n, \theta)) \nabla \log p(\tilde{y}_n|f(x_n, \theta))^\top. \quad (28)$$

While noisy, this one-sample Monte Carlo estimate is unbiased and will not suffer from the issues mentioned in the main text. This is the approach used by Martens and Grosse [2015] as well as Zhang et al. [2018].

As a side note, some implementations use a biased estimate in which, instead of sampling $\tilde{y}_n$ from $p(y|f(x_n, \theta))$, they compute the most likely output $\hat{y}_n = \arg\max_y p(y|f(x_n, \theta))$. This scheme could be beneficial in some circumstances, as it reduces variance, but it can also backfire by increasing the bias of the estimation. For the least-squares loss, as $p(y|f(x_n, \theta))$ is a Gaussian distribution centered as $f(x, \theta)$, the most likely output is $f(x_n, \theta)$ and the gradient $\log p(y|f(x_n, \theta))|_{y=f(x_n,\theta)}$ is zero, which defeats the purpose of sampling.

For high quality estimates, however, sampling additional outputs and averaging the results is inefficient. If M MC samples $\tilde{y}_1, \ldots, \tilde{y}_M$ per input x_n are used to compute the gradients $g_m = \nabla - \log p(\tilde{y}_m|f(x_n, \theta))$, most of the computation is repeated. The gradient g_m is

$$g_m = -\nabla \log p(\tilde{y}_m|f(x_n, \theta)) = -(\mathrm{J}_\theta f(x_n, \theta))^\top \nabla_f \log p(\tilde{y}_m|f), \quad (29)$$

where the Jacobian of the model output, $\mathrm{J}_\theta f$, does not depend on $\tilde{y}_m$. As the Jacobian of the model is much more complex to compute than the gradient of the log-likelihood w.r.t. the model output, this approach repeats the most difficult computation, especially when the model is a neural network. The expectation can instead be computed in closed form using the generalized Gauss-Newton equation (Eq. 23, or Eq. 6 in the main text), which requires the computation of the Jacobian only once per sample x_n.

The main issue with this approach is that computing Jacobians is currently not well supported by deep learning auto-differentiation libraries, such as TensorFlow or Pytorch. However, the current the implementations relying on the empirical Fisher also suffer from this lack of support, as they need access to the individual gradients to compute their outer-product. Access to the individual gradients is equivalent to computing the Jacobian of the vector $[-\log p(y_1|f(x_1, \theta)), ..., -\log p(y_N|f(x_N, \theta)]^\top$. The ability to efficiently compute Jacobians and/or individual gradients in parallel would drastically improve the practical performance of methods based on the Fisher and empirical Fisher, as most of the computation of the backward pass can be shared between samples.

C Additional proofs

C.1 The Fisher as the expected Hessian

In §2.2 Eq. (8), we mentioned that the two following representations of the Fisher are equivalent:

$$\mathbb{E}_{p_\theta(z)}\left[\nabla_\theta \log p_\theta(z)\, \nabla_\theta \log p_\theta(z)^\top\right] = \mathbb{E}_{p_\theta(z)}\left[-\nabla^2_\theta \log p_\theta(z)\right]. \tag{30}$$

To see why, apply the chain rule on the $\log$ to split the equation in terms of the Hessian and the outer product of the gradients of p_θ,

$$\mathbb{E}_{p_\theta(z)}\left[-\nabla^2_\theta \log p_\theta(z)\right] = \mathbb{E}_{p_\theta(z)}\left[-\tfrac{1}{p_\theta(z)}\nabla^2_\theta p_\theta(z)\right] + \mathbb{E}_{p_\theta(z)}\left[\tfrac{1}{p_\theta(z)^2}\nabla_\theta p_\theta(z) \nabla_\theta p_\theta(z)^\top\right]. \tag{31}$$

The first term on the right-hand side is zero, since

$$\begin{aligned}\mathbb{E}_{p_\theta(z)}\left[-\tfrac{1}{p_\theta(z)}\nabla^2_\theta p_\theta(z)\right] &:= -\int_z \frac{1}{p_\theta(z)}\nabla^2_\theta p_\theta(z) p_\theta(z)\,\mathrm{d}z = \int_z \nabla^2_\theta p_\theta(z)\,\mathrm{d}z, \\ &= \nabla^2_\theta \int p_\theta(z)\,\mathrm{d}z = \nabla^2_\theta[1] = 0.\end{aligned} \tag{32}$$

The second term of Eq. (31) is the expected outer-product of the gradients, as $\partial_\theta \log f(\theta) = \frac{1}{x}\partial_\theta f(\theta)$,

$$\begin{aligned}\tfrac{1}{p_\theta(z)^2}\nabla_\theta p_\theta(z) \nabla_\theta p_\theta(z)^\top &= \left(\frac{1}{p_\theta(z)}\nabla_\theta p_\theta(z)\right)\left(\frac{1}{p_\theta(z)}\nabla_\theta p_\theta(z)\right)^\top, \\ &= \nabla_\theta \log p_\theta(z)\, \nabla_\theta \log p_\theta(z)^\top.\end{aligned} \tag{33}$$

The same technique also shows that if the empirical distribution over the data is equal to the model distribution $p_\theta(y|f(x,\theta)$, then the Fisher, empirical Fisher and the Hessian are all equal.

C.2 Proof of Propositon 1

In §2.3, Prop. 1, we mentioned that the Fisher and the generalized Gauss-Newton are equivalent for the problems considered in the introduction;

> **Proposition 1** (Martens [2014], §9.2). *If $p(y|f)$ is an exponential family distribution with natural parameters f, then the Fisher information matrix coincides with the GGN of Eq.* (1) *using the split*
>
> $$a_n(b) = -\log p(y_n|b), \qquad b_n(\theta) = f(x_n,\theta),$$
>
> *and reads* $\mathrm{F}(\theta) = G(\theta) = \sum_n [\mathrm{J}_\theta f(x_n,\theta)]^\top\, \nabla^2_f \log p(y_n|f(x_n,\theta))\, [\mathrm{J}_\theta f(x_n,\theta)]$.

Plugging the split into the definition of the GGN (Eq. 6) yields $G(\theta)$, so we only need to show that the Fisher coincides with this GGN. By the chain rule, we have

$$\nabla_\theta \log p(y|f(x_n,\theta)) = \mathrm{J}_\theta f(x_n,\theta)^\top\, \nabla_f \log p(y|f(x_n,\theta)), \tag{34}$$

and we can then apply the following steps.

$$\mathrm{F}(\theta) = \sum_n \mathop{\mathbb{E}}_{y\sim p_\theta(y|x_n)}\left[\mathrm{J}_\theta f(x_n,\theta)^\top\, \nabla_f \log p(y|f_n)\, \nabla_f \log p(y|f_n)^\top \mathrm{J}_\theta f(x_n,\theta)\right], \tag{35}$$

$$= \sum_n \mathrm{J}_\theta f(x_n,\theta)^\top \mathop{\mathbb{E}}_{y\sim p_\theta(y|x_n)}\left[\nabla_f \log p(y|f_n)\, \nabla_f \log p(y|f_n)^\top\right] \mathrm{J}_\theta f(x_n,\theta), \tag{36}$$

$$= \sum_n \mathrm{J}_\theta f(x_n,\theta)^\top \mathop{\mathbb{E}}_{y\sim p_\theta(y|x_n)}\left[-\nabla^2_f \log p(y|f_n)\right] \mathrm{J}_\theta f(x_n,\theta), \tag{37}$$

Eq. (35) rewrites the Fisher using the chain rule, Eq. (36) take the Jacobians out of the expectation as they do not depend on y and Eq. (37) is due to the equivalence between the expected outer product of gradients and expected Hessian shown in the last section.

If p is an exponential family distribution with natural parameters (a linear combination of) f, its log density has the form $\log p(y|f) = f^T T(y) - A(f) + \log h(y)$ (where T are the sufficient statistics, A is the cumulant function and h is the base measure). The Hessian w.r.t. f is simply $\nabla^2_f A(f)$ and is independent of y, and hence,

$$\mathrm{F}(\theta) = \textstyle\sum_n \mathrm{J}_\theta f(x_n,\theta)^\top \nabla^2_f(-\log p(y_n|f_n))\mathrm{J}_\theta f(x_n,\theta), \tag{38}$$

C.3 Proof of Proposition 2

In §2.3, Prop. 2, we show that the difference between the Fisher (or the GNN) and the Hessian can be bounded by the residuals and the smoothness constant of the model f;

> **Proposition 2.** *Let $\mathcal{L}(\theta)$ be defined as in Eq. (1) with $\mathbb{F} = \mathbb{R}^M$. Denote by $f_n^{(m)}$ the m-th component function of $f(x_n, \cdot)\colon \mathbb{R}^D \to \mathbb{R}^M$ and assume each $f_n^{(m)}$ is β-smooth. Let $G(\theta)$ be the GGN (Eq. 6). Then,*
>
> $$\left\|\nabla^2 \mathcal{L}(\theta) - G(\theta)\right\|_2^2 \leq r(\theta)\beta, \tag{39}$$
>
> *where $r(\theta) = \sum_{n=1}^N \|\nabla_f \log p(y_n|f(x_n,\theta))\|_1$ and $\|\cdot\|_2$ denotes the spectral norm.*

Dropping θ from the notation for brevity, the Hessian can be expressed as

$$\nabla^2 \mathcal{L} = G + \sum_{n=1}^N \sum_{m=1}^M r_n^{(m)} \nabla_\theta^2 f_n^{(m)}, \qquad \text{where} \qquad r_n^{(m)} = \left.\frac{\partial \log p(y_n|f)}{\partial f^{(m)}}\right|_{f=f_n(\theta)} \tag{40}$$

is the derivative of $-\log p(y|f)$ w.r.t. the m-th component of f, evaluated at $f = f_n(\theta)$.

If all $f_n^{(m)}$ are β-smooth, we have $-\beta I \preceq \nabla_\theta^2 f_n^{(m)} \preceq \beta I$ and, consequently,

$$-\left|\sum_{n,m} r_n^{(m)}\right| \beta\, \mathbf{I} \preceq \nabla^2 \mathcal{L} - G \preceq \left|\sum_{n,m} r_n^{(m)}\right| \beta\, \mathbf{I}\,. \tag{41}$$

Pulling the absolute value inside the double sum gives the upper bound

$$\left|\sum_{n,m} r_n^{(m)}\right| \leq \sum_n \sum_m \left| \left.\frac{\partial \log p(y_n|f)}{\partial f^{(m)}}\right|_{f=f_n(\theta)} \right| = \sum_n \|\nabla_f \log p(y_n|f_n(\theta))\|_1, \tag{42}$$

and the statement about the spectral norm (the largest singular value of the matrix) follows.

D Additional plots

Fig. 4 repeats the experiment described in Fig. 2 (§3.2), on the effect of model misspecification on the Fisher and empirical Fisher at the minimum, on linear regression problems instead of a classification problem. Similar issues in scaling and directions can be observed.

Fig. 5 repeats the experiment described in Fig. 3 (§3.3) on additional linear regression problems. Those additional examples show that the poor performance of empirical Fisher-preconditioned updates compared to NGD is not isolated to the examples shown in the main text.

Fig. 6 show the linear regression problem on the Boston dataset, originally shown in Fig. 3, where each line is a different starting point, using the same hyperparameters as in Fig. 3. The starting points are selected from $[-\theta^\star, \theta^\star]$, where $\theta^\star$ is the optimum. When the optimization starts close to the minimum (low loss), the empirical Fisher is a good approximation to the Fisher and there are very few differences with NGD. However, when the optimization starts far from the minimum (high loss), the individual gradients, and thus the sum of outer product gradients, are large, which leads to very small steps, regardless of curvature, and slow convergence. While this could be counteracted with a larger step size in the beginning, this large step size would not work close to the minimum and would lead to oscillations. The selection of the step size therefore depends on the starting point, and would ideally be on a decreasing schedule.

E Experimental details

In contrast to the main text of the paper, which uses the sum formulation of the loss function,

$$\mathcal{L}(\theta) = \sum_n \log p(y_n|f(x_n,\theta)),$$

the implementation—and thus the reported step sizes and damping parameters—apply to the average,

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_n \log p(y_n|f(x_n,\theta)).$$

The Fisher and empirical Fisher are accordingly rescaled by a $1/N$ factor.

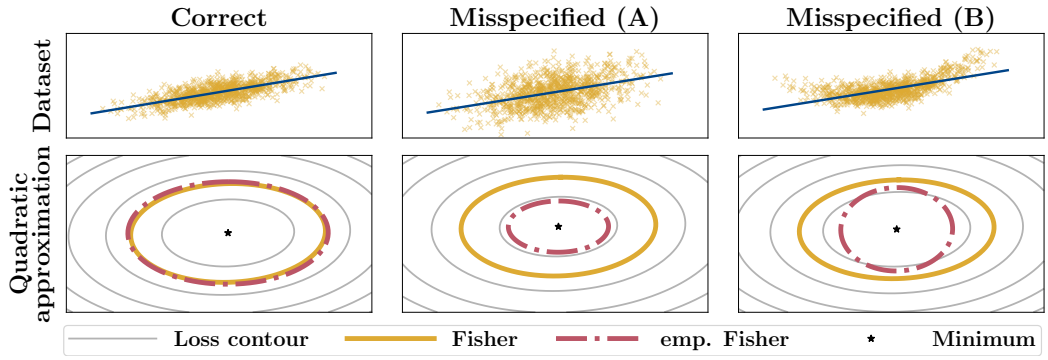

Figure 4: Quadratic approximations of the loss function using the Fisher and the empirical Fisher on a linear regression problem. The EF is a good approximation of the Fisher at the minimum if the data is indeed generated by $y \sim \mathcal{N}(x\theta^* + b^*, 1)$ as the model assumes (left panel), but can be arbitrarily wrong if the assumption is violated, even at the minimum and with large N. In the middle (A), the model is misspecified as it under-estimates the observation noise (data is generated by $y \sim \mathcal{N}(x\theta^* + b^*, 2)$). In the right plot (B), the model is misspecified as it fails to capture the quadratic relationship between x and y.

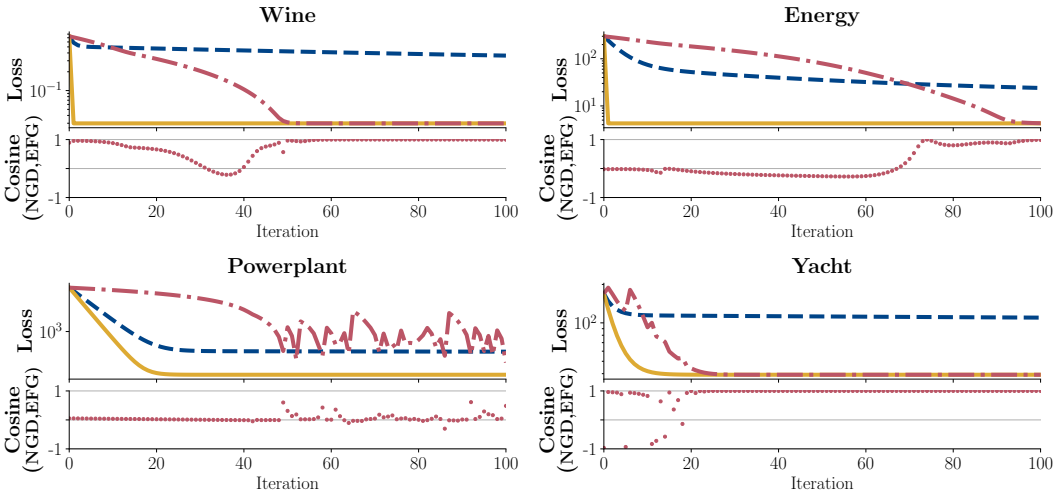

Figure 5: Comparison of the Fisher (NGD) and the empirical Fisher (EFGD) as preconditioners on additional linear regression problems. The second row shows the cosine similarity between the EF-preconditioned gradient and the natural gradient at each step on the path taken by EFGD.

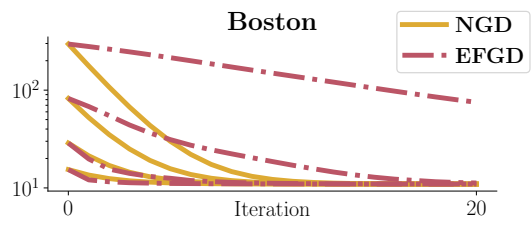

Figure 6: Linear regression on the Boston dataset with different starting points (each line is a different initialization). When the optimization starts close to the minimum (low initial loss), the empirical Fisher is a good approximation to the Fisher and there are very few differences with NGD, but the performance degrades as the optimization procedure starts farther away (large initial loss).

E.1 Vector field of the empirical Fisher preconditioning

The problem used for Fig. 1 is a linear regression on $N = 1000$ samples from

$$x_i \sim \text{Lognormal}\left(0, 3/4\right), \qquad \epsilon_i \sim \mathcal{N}\left(0, 1\right), \qquad y_i = 2 + 2x_i + \epsilon_i. \tag{43}$$

To be visible and of a similar scale, the gradient, natural gradient and empirical Fisher-preconditioned gradient were relatively rescaled by $1/3$, 1 and 3, respectively. The trajectories of each method is computed by running each update,

$$\text{GD:} \qquad \theta_{t+1} = \theta_t - \gamma \nabla \mathcal{L}(\theta_t). \tag{44}$$

$$\text{NGD:} \qquad \theta_{t+1} = \theta_t - \gamma (\mathrm{F}(\theta_t) + \lambda \mathbf{I})^{-1} \nabla \mathcal{L}(\theta_t), \tag{45}$$

$$\text{EFGD:} \qquad \theta_{t+1} = \theta_t - \gamma (\widetilde{\mathrm{F}}(\theta_t) + \lambda \mathbf{I})^{-1} \nabla \mathcal{L}(\theta_t), \tag{46}$$

using a step size of $\gamma = 10^{-4}$ and a small damping parameter of $\lambda = 10^{-8}$ to ensure stability for $50'000$ iterations. The vector field was computed using the same damping parameter. The starting points for the problem are

$$\begin{bmatrix} 2 \\ 4.5 \end{bmatrix}, \qquad \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} 4.5 \\ 3 \end{bmatrix}, \qquad \begin{bmatrix} -0.5 \\ 3 \end{bmatrix}.$$

E.2 EF as a quadratic approximation at the minimum for misspecified models

The problems for using the Scipy [Jones et al., 2001] implementation of BFGS[4] to find the minimum of the problem, and then plots the quadratic approximation of the loss function using the matrix M (the Fisher or empirical Fisher), $\mathcal{L}(\theta) \approx \frac{1}{2}(\theta - \theta^\star)M(\theta - \theta^\star)$, for $\|\theta - \theta^\star\|^2 = 1$. The datasets used for the logistic regression problem of Fig. 2 are $N = 1'000$ samples from the data generating processes described in Table 1. Fig. 4 shows additional examples of model misspecification on a linear regression problem using the data generating processes described in Table 2.

Table 1: Datasets used for Fig. 2. For all datasets, $p(y = 0) = p(y = 1) = 1/2$.

Model	$p(x\|y = 0)$	$p(x\|y = 1)$
Correct model:	$\mathcal{N}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\right)$	$\mathcal{N}\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\right)$
Misspecified (A):	$\mathcal{N}\left(\begin{bmatrix} 1.5 \\ 1.5 \end{bmatrix}, \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}\right)$	$\mathcal{N}\left(\begin{bmatrix} -1.5 \\ -1.5 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$
Misspecified (B):	$\mathcal{N}\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1.5 & -0.9 \\ -0.9 & 1.5 \end{bmatrix}\right)$	$\mathcal{N}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1.5 & 0.9 \\ 0.9 & 1.5 \end{bmatrix}\right)$

Table 2: Datasets used for Fig. 4. For all datasets, $x \sim \mathcal{N}\left(0, 1\right)$.

Model	y	ϵ
Correct model:	$y = x + \epsilon$	$\epsilon \sim \mathcal{N}\left(0, 1\right)$
Misspecified (A):	$y = x + \epsilon$	$\epsilon \sim \mathcal{N}\left(0, 2\right)$
Misspecified (B):	$y = x + \frac{1}{2}x^2 + \epsilon$	$\epsilon \sim \mathcal{N}\left(0, 1\right)$

E.3 Optimization with the empirical Fisher as preconditioner

The optimization experiment uses the update rules described in §E.1 by Eq. (44, 45, 46). The cosine similarity is computed between the gradient preconditioned with the empirical Fisher and the Fisher, without damping, at each step along the path taken by the empirical Fisher preconditioned updates.

[4] https://docs.scipy.org/doc/scipy/reference/optimize.minimize-bfgs.html

The problems are initialized at $\theta_0 = 0$ and run for 100 iterations. Note that this initialization is favorable to the empirical Fisher for the logistic regression problems. Not only is it guaranteed to not be arbitrarily wrong, but the empirical Fisher and the Fisher actually coincide when the predicted probabilities are uniform. For the sigmoid activation of the output of the linear mapping, $\sigma(f)$, the gradient and Hessian of the log-probability w.r.t. f, are given by

$$\nabla_f(-\log p(y|f)) = \sigma(f), \qquad \nabla_f^2(-\log p(y|f)) = \sigma(f)(1-\sigma(f)). \tag{47}$$

The gradient squared and the Hessian thus coincide (only) when $\sigma(f) = \frac{1}{2}$, which happens when the model is initialized at 0.

The step size and damping hyperparameters are selected by a gridsearch, selecting for each optimizer the run with the minimal loss after 100 iterations. The grid used is described in Table 4 as a log-space[5]. Table 3 describes the datasets used and Table 5 the hyperparameters selected by the gridsearch.

[5] https://docs.scipy.org/doc/numpy/reference/generated/numpy.logspace.html

Table 3: Datasets

Dataset	# Features	# Samples	Type	Figure
a1a[6]	1′605	123	Classification	Fig. 3
BreastCancer[7]	683	10	Classification	Fig. 3
Boston Housing[8]	506	13	Regression	Fig. 3
Yacht Hydrodynamics[9]	308	7	Regression	Fig. 5
Powerplant[10]	9′568	4	Regression	Fig. 5
Wine[11]	178	13	Regression	Fig. 5
Energy[12]	768	8	Regression	Fig. 5

Table 4: Grid used for the hyperparameter search for the optimization experiments, in $\log_{10}$. The number of samples to generate was selected as to generate a smooth grid in base 10, e.g., $10^0, 10^{.25}, 10^{.5}, 10^{.75}, 10^1, 10^{1.25}, \ldots$

Parameter		Grid
Step size	γ	`logspace(start=-20, stop=10, num=241)`
Damping	λ	`logspace(start=-10, stop=10, num=41)`

Table 5: Selected hyperparameters, given in $\log_{10}$.

Dataset	Algorithm	γ	λ
Boston	GD	-5.250	
	NGD	0.125	-10.0
	EFGD	-1.250	-8.0
BreastCancer	GD	-5.125	
	NGD	0.125	-10.0
	EFGD	-1.250	-10.0
a1a	GD	0.250	
	NGD	0.250	-10.0
	EFGD	-0.375	-8.0
Wine	GD	-5.625	
	NGD	0.000	-8.5
	EFGD	-1.375	-6.0
Energy	GD	-5.500	
	NGD	0.000	-7.5
	EFGD	0.875	-3.0
Powerplant	GD	-5.750	
	NGD	-0.625	-8.0
	EFGD	3.375	-1.0
Yacht	GD	-1.500	
	NGD	-0.750	-7.5
	EFGD	1.625	-6.5

[6] www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html#a1a
[7] www.csie.ntu.edu.tw/čjlin/libsvmtools/datasets/binary.html#breast-cancer
[8] scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html
[9] archive.ics.uci.edu/ml/datasets/Yacht+Hydrodynamics
[10] archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant
[11] archive.ics.uci.edu/ml/datasets/Wine
[12] archive.ics.uci.edu/ml/datasets/Energy+efficiency