Visualizing high-dimensional loss landscapes with Hessian directions

Lucas Böttcher\textsuperscript{1,2,*} and Gregory Wheeler\textsuperscript{1}

\textsuperscript{1} Department of Computational Science and Philosophy, Frankfurt School of Finance and Management, 60322 Frankfurt am Main, Germany
\textsuperscript{2} Department of Medicine, University of Florida, Gainesville, FL 32610, United States of America
E-mail: lbottcher@ufl.edu

Received 23 August 2023
Accepted for publication 21 November 2023
Published 6 February 2024

Abstract. Analyzing the geometric properties of high-dimensional loss functions, such as local curvature and the existence of other optima around a certain point in loss space, can help provide a better understanding of the interplay between neural-network structure, implementation attributes, and learning performance. In this paper, we combine concepts from high-dimensional probability and differential geometry to study how curvature properties in lower-dimensional loss representations depend on those in the original loss space. We show that saddle points in the original space are rarely correctly identified as such in the expected lower-dimensional representations if random projections are used. The principal curvature in the expected lower-dimensional representation is proportional to the mean curvature in the original loss space. Hence, the mean curvature in the original loss space determines if saddle points appear, on average, as either minima, maxima, or almost flat regions. We use the connection between expected curvature in random projections and mean curvature in the original space (i.e. the normalized Hessian trace) to compute Hutchinson-type trace estimates without calculating Hessian-vector products as in the original Hutchinson method. Because random projections are not suitable for correctly identifying...
saddle information, we propose to study projections along the dominant Hessian directions that are associated with the largest and smallest principal curvatures. We connect our findings to the ongoing debate on loss landscape flatness and generalizability. Finally, for different common image classifiers and a function approximator, we show and compare random and Hessian projections of loss landscapes with up to approximately $7 \times 10^6$ parameters.

**Keywords:** machine learning, loss landscapes, random projections, Hessian directions, principal curvature

---

**Contents**

1. **Introduction** ................................................................. 2
2. **Principal curvature in random projections** .......................... 4
   2.1. Differential and information geometry concepts .................. 4
   2.2. Random projections .................................................... 7
   2.3. Principal curvature .................................................... 8
   2.4. Hessian trace estimates .............................................. 10
3. **Illustrative examples** .................................................... 11
   3.1. Extracting curvature information ................................... 11
   3.2. Hessian trace .......................................................... 15
   3.3. Hessian directions .................................................... 16
4. **Applications to neural networks** ...................................... 17
5. **Discussion and conclusion** .............................................. 23
   **Acknowledgments** .......................................................... 25
   **Appendix A. Concentration inequality** .............................. 25
   **Appendix B. Annihilation method** .................................... 26
   **Appendix C. Image classification** .................................... 27
   **Appendix D. Function approximation** ............................... 30
   **References** ............................................................... 33

---

1. **Introduction**

Every deep neural-network loss function depends on the parameters, $\theta \in \mathbb{R}^N$, which induce a loss landscape that is typically in high dimensions [1]. The loss landscape associated with a neural network and the performance of optimization procedures that operate on it are influenced by several factors, including the structural properties of...
the neural network [2–4] and a range of implementation attributes one may choose [5–9]. The exact nature of these factors and how their combinations influence learning performance, however, remains largely unknown.

One way to better understand the interplay between neural-network structure, implementation attributes, and learning performance is through examining the geometric properties of loss landscapes. For example, Keskar et al [10] analyzed the local curvature around candidate minimizers via the spectrum of the underlying Hessian to characterize the flatness and sharpness of loss minima, and Dinh et al [11] demonstrated that reparameterizations can render flat minima sharp without affecting generalization properties. Another approach that holds out the promise of visualizing the curvature around a point and the existence, if any, of nearby optima aims to visualize high-dimensional loss functions by a lower-dimensional (and often random) projection of two or three dimensions [12–16]. Building on this approach, Horoi et al [16] pursued improvements in learning by dynamically sampling points in projected low-loss regions surrounding local minima during training.

Given the advantages afforded by the promise of visualizing loss landscape optima, one may ask how the curvature properties in low-dimensional visualizations of high-dimensional functions depend on the curvature properties of the original high-dimensional loss space. Specifically, do random two- and three-dimensional loss projections meaningfully represent convexity and concavity properties of high-dimensional loss functions? In short, they do not. Saddle points are generally misrepresented in the expected lower-dimensional representation of an original high-dimensional loss space. Yet, due to the exponential proliferation of saddle points in high dimensions [17, 18], a critical point in high-dimensional loss space is almost certain to be a saddle rather than a minimum. Nevertheless, random projections can be useful for obtaining curvature estimates, as we shall demonstrate by establishing a connection between the expected curvature in random loss projections and Hessian trace estimates based on Hutchinson’s method [19].

Our contributions in this paper are threefold. First, we explain why random projections from a high-dimensional space do not preserve curvature information. This argument is developed in section 2. One consequence of this result is that the saddle points of an original high-dimensional loss landscape are not identified correctly in the associated expected low-dimensional random projection, a point we illustrate with simulation examples in section 3.1. Second, while principal curvatures associated with a random projection are not weighted ensemble means of the principal curvatures in the original high-dimensional space, the principal curvatures in a low-dimensional projection are given by functions of the weighted ensemble means of the Hessian elements in the original high-dimensional space. This result is developed in section 2 as well, and it is also illustrated by a simulation example in section 3.1. A complementary simulation example in section 3.2 shows how one can estimate mean curvature from the slope of random loss projections. Section 2 is intended to be a self-contained description of the relationship between the principal curvatures of a loss function and the principal curvatures associated with a lower-dimensional, random projection.
Our third contribution builds on the results reported in sections 2 and 3.1 to propose an efficient method for estimating and visualizing Hessian information in high-dimensional loss landscapes. Instead of using random projections to visualize loss functions, we propose in sections 3.3 and 4 to analyze projections along dominant Hessian directions that are associated with the largest-magnitude positive and negative principal curvatures, i.e. those directions along which the magnitude of positive and negative changes of the loss function are the largest. For different common image classifiers and a function approximator as studied in [20], we show and compare random and Hessian projections of loss landscapes with up to approximately $7 \times 10^6$ parameters. In accordance with related works unlocking Hessian information in nonlinear and high-dimensional settings [21, 22], our proposal uses Hessian-vector products (HVPs), thereby bypassing the computational expense of computing a Hessian matrix. In short, the role of random projections in our approach is to estimate mean curvature and weighted ensemble means of the elements of the original high-dimensional Hessian.

Finally, we conclude our paper in section 5 with a discussion of our results. Our source code is publicly available in [23].

2. Principal curvature in random projections

To describe the connection between the principal curvature of a loss function $L(\theta)$ and that associated with a lower-dimensional, random projection, we provide in section 2.1 an overview of concepts from differential geometry [24–26] that are useful to mathematically describe curvature in high-dimensional spaces. In sections 2.2 and 2.3, we show the relationship between principal curvature and random projections. Finally, in section 2.4, we highlight the relationship between the measures of curvature presented in section 2.3 and Hutchinson-type Hessian trace estimates.

2.1. Differential and information geometry concepts

In differential geometry, the principal curvatures are the eigenvalues of the shape operator (or Weingarten map$^3$), which is a linear endomorphism defined on the tangent space $T_p$ of $L$ at point $p$. For the original high-dimensional space, we have $(\theta, L(\theta)) \subseteq \mathbb{R}^{N+1}$, and there are $N$ principal curvatures $\kappa_1^\theta \geq \kappa_2^\theta \geq \cdots \geq \kappa_N^\theta$. At a nondegenerate critical point $\theta^*$ where the gradient $\nabla_{\theta} L$ vanishes, the matrix of the shape parameter is given by the Hessian $H_\theta$ with elements $(H_\theta)_{ij} = \frac{\partial^2 L}{\partial \theta_i \partial \theta_j}$ ($i, j \in \{1, \ldots, N\}$).$^4$ Some studies refer to the Hessian as the ‘curvature matrix’ [28] or use it to characterize the curvature properties of $L(\theta)$ [13]. In the vicinity of a nondegenerate critical point $\theta^*$, the eigenvalues of the Hessian $H_\theta$ are the principal curvatures and describe the loss function in the eigenbasis of $H_\theta$ according to

---

$^3$ Some authors distinguish between the shape operator and the Weingarten map depending on whether the change in the underlying tangent vector is described in the original manifold or in Euclidean space (see, e.g. chapter 3 in [27]).

$^4$ A critical point is degenerate if the Hessian $H_\theta$ at this point is singular (i.e. $\det(H_\theta) = 0$). At degenerate critical points, one cannot use the eigenvalues of $H_\theta$ to determine whether the critical point is a minimum (positive definite $H_\theta$) or a maximum (negative definite $H_\theta$). Geometrically, at a degenerate critical point, a quadratic approximation fails to capture the local behavior of the function that one wishes to study.

https://doi.org/10.1088/1742-5468/ad13fc
Visualizing high-dimensional loss landscapes with Hessian directions

\[ L(\theta^* + \Delta \theta) = L(\theta^*) + \frac{1}{2} \sum_{i=1}^{N} \kappa_i^2 \Delta \theta_i^2. \]  

(1)

The Morse lemma states that if a critical point \( \theta^* \) of \( L(\theta) \) is nondegenerate, then there exists a chart \( (\tilde{\theta}_1, \ldots, \tilde{\theta}_N) \) in a neighborhood of \( \theta^* \) such that

\[ L(\tilde{\theta}) = -\tilde{\theta}_1^2 - \cdots - \tilde{\theta}_i^2 + \tilde{\theta}_{i+1}^2 + \cdots + \tilde{\theta}_N^2 + L(\theta^*), \]  

(2)

where \( \tilde{\theta}_i(\theta^*) = 0 \) for \( i \in \{1, \ldots, N\} \). The loss function \( L(\tilde{\theta}) \) in equation (2) is decreasing along \( i \) directions and increasing along the remaining \( i+1 \) to \( N \) directions. Further, the index \( i \) of a critical point \( \theta^* \) is the number of negative eigenvalues of the Hessian \( H_\theta \) at that point.

In the standard basis, the Hessian is

\[ H_\theta := \nabla_\theta \nabla_\theta L(\theta) = \begin{pmatrix} \frac{\partial^2 L}{\partial \theta_1^2} & \cdots & \frac{\partial^2 L}{\partial \theta_i \partial \theta_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 L}{\partial \theta_N \partial \theta_1} & \cdots & \frac{\partial^2 L}{\partial \theta_N^2} \end{pmatrix}. \]  

(3)

Closely related to the Hessian is the Fisher information matrix (FIM) \[ 29, 30 \], whose elements \( F_{ij} \) are given by

\[ F_{ij} = \mathbb{E} \left[ \frac{\partial}{\partial \theta_i} \log p(X | \theta) \frac{\partial}{\partial \theta_j} \log p(X | \theta) \right], \]  

(4)

where the expectation is taken over the random variable \( X \), and \( p(x | \theta) \) denotes the probability density function (PDF) associated with \( X \) conditioned on parameter \( \theta \). The probability that \( X \) falls within the infinitesimal interval \([x, x + dx]\), given a known value of \( \theta \), is \( p(x | \theta) dx \). Provided that certain regularity conditions such as

\[ \frac{d}{d\theta_i} \mathbb{E} \left[ \frac{\partial}{\partial \theta_j} \log p(X | \theta) \right] = \int \frac{\partial}{\partial \theta_i} \left[ \left( \frac{\partial}{\partial \theta_j} \log p(x | \theta) \right) p(x | \theta) \right] dx \]  

are satisfied \[ 29, 31 \], we may rewrite equation (4) as

\[ F_{ij} = -\mathbb{E} \left[ \frac{\partial^2}{\partial \theta_i \partial \theta_j} \log p(X | \theta) \right]. \]  

(5)

Notice the structural similarity between \( F_{ij} \) and the elements of \( H_\theta \) (see equation (3)). In natural gradient descent \[ 32 \], the metric associated with a stochastic feedforward neural network is the FIM.

Considering a neural network with input–output pairs \((x, y)\) and parameters \( \theta \), a possible choice of a model distribution, which takes on the role of the probability density in equation (4), is \( p(x, y | \theta) = p(y | x, \theta)q(x) \) with \( p(y | x, \theta) = \exp(-\|y - f_\theta(x)\|_2^2)/\sqrt{2\pi} \). Here, \( q(x) \) denotes the distribution of inputs \( x \), \( f_\theta(x) \) is the output of a neural network given input \( x \), and \( \| \cdot \|_2 \) is the Euclidean norm \[ 33 \]. The expected values associated with
FIM elements $F_{ij}$ for $p(x, y|\theta)$ are calculated with respect to (w.r.t.) input–output pairs $(x, y)$ of the joint distribution $p(x, y|\theta)$.

For a set of input samples $x(t)$ ($t \in \{1, \ldots, T\}$) and corresponding outputs $f_\theta(t) \equiv f_\theta(x(t))$, the empirical FIM is

$$F_\theta = \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} \nabla_{\theta} f_{\theta,k}(t) \nabla_{\theta} f_{\theta,k}(t)^T,$$

(7)

where $f_{\theta,k}$ is the $k$th entry of the neural-network output, and $k \in \{1, \ldots, C\}$ [33, 34]. The empirical FIM converges toward the FIM, equation (4), as $T \to \infty$.

Given the mean squared loss function

$$L(\theta) = \frac{1}{2T} \sum_{t=1}^{T} \|y(t) - f_\theta(t)\|^2_2,$$

(8)

the Hessian $H_\theta$ of $L(\theta)$ and the empirical FIM $F_\theta$ are connected via

$$H_\theta = F_\theta - \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} (y_k(t) - f_{\theta,k}(t)) \nabla_{\theta} f_{\theta,k}(t) .$$

(9)

Here, $y(t)$ denotes the desired output associated with sample $x(t)$. At the global optimum, which is usually attained for overparameterized neural networks [1], the difference $y_k(t) - f_{\theta,k}(t)$ vanishes for all $k, t$, and the empirical FIM is equal to the Hessian $H_\theta$ [34].

We can also establish a connection between $H_\theta$ and $F_\theta$ for the cross-entropy loss $L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} y_k(t) \log(f_{\theta,k}(t))$, where softmax outputs $f_{\theta,k}(t)$ are used to approximate binary class labels $y_k(t)$. Similar to equation (9), we obtain

$$H_\theta = \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} y_k(t) \left( \frac{\nabla_{\theta} f_{\theta,k}(t) \nabla_{\theta} f_{\theta,k}(t)^T}{f_{\theta,k}(t)^2} - \frac{f_{\theta,k}(t) \nabla_{\theta} f_{\theta,k}(t)}{f_{\theta,k}(t)^2} \right) .$$

(10)

Given the scenario in which $f_{\theta,k}(t)$ approaches $y_k(t)$ for all $k, t$ and using $\sum_{k=1}^{C} f_{\theta,k}(t) = 1$ (i.e. $\sum_{k=1}^{C} \nabla_{\theta} f_{\theta,k}(t) = 0$), we again have $H_\theta = F_\theta$ where

$$F_\theta = \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} \frac{1}{f_{\theta,k}(t)} \nabla_{\theta} f_{\theta,k}(t) \nabla_{\theta} f_{\theta,k}(t)^T$$

(11)

is the empirical FIM associated with the cross-entropy loss [35].

The neural-network examples that we consider in section 4 are trained using cross-entropy loss. In appendix D, we also consider a function approximation problem, in which we train neural networks using a mean squared loss function.

---

5 In this case, the underlying conditional probability distribution is $p(y|x, \theta) = \prod_{k=1}^{C} (f_{\theta,k}(t))^n$ [35].
### 2.2. Random projections

To graphically explore an $N$-dimensional loss function $L$ around a critical point $\theta^*$, one may wish to work in a lower-dimensional representation. For example, a two-dimensional projection of $L$ around $\theta^*$ is provided by

$$ L\left(\theta^* + \alpha \eta + \beta \delta\right), $$

where the parameters $\alpha, \beta \in \mathbb{R}$ scale the directions $\eta, \delta \in \mathbb{R}^N$. The corresponding graph representation is $(\alpha, \beta, L(\alpha, \beta)) \subseteq \mathbb{R}^3$.

In high-dimensional spaces, there are many more almost-orthogonal directions than orthogonal directions. In fact, if the dimension of our space is large enough, with high probability, random vectors will be sufficiently close to orthogonal [36]. Following this result, many related studies [13, 15, 16] have used random Gaussian directions with independent and identically distributed vector elements $\eta_i, \delta_i \sim \mathcal{N}(0,1)$ ($i \in \{1, \ldots, N\}$).

We now turn to showing that $\eta, \delta$ are almost orthogonal using a concentration inequality for chi-squared distributed random variables. To do so, we first note that the scalar product of random Gaussian vectors $\eta, \delta$ is a sum of the difference between two chi-squared distributed random variables because

$$ \sum_{i=1}^{N} \eta_i \delta_i = \sum_{i=1}^{N} \frac{1}{4} (\eta_i + \delta_i)^2 - \frac{1}{4} (\eta_i - \delta_i)^2 = \sum_{i=1}^{N} \frac{1}{2} X_i^2 - \frac{1}{2} Y_i^2, $$

where $X_i, Y_i \sim \mathcal{N}(0,1)$. In the last step of equation (13), we rescaled $X_i$ and $Y_i$ by a factor $\sqrt{2}$ to obtain normally distributed random variables with variance 2. Since $E[Z] = N$ for $Z = \sum_{i=1}^{N} X_i^2$, the right-hand side of equation (13) vanishes in expectation. That is,

$$ \lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i = 0. $$

Hence, the vectors $\eta, \delta$ are orthogonal in the limit $N \to \infty$. For finite $N$, we can bound equation (13) using the concentration inequalities

$$ \Pr\left( \frac{1}{N} \sum_{i=1}^{N} X_i^2 - 1 \geq \epsilon + \frac{1}{2} \epsilon^2 \right) \leq e^{-\frac{\epsilon^2}{4}} $$

$$ \Pr\left( 1 - \frac{1}{N} \sum_{i=1}^{N} X_i^2 \geq \epsilon \right) \leq e^{-\frac{\epsilon^2}{4}} $$

for any $\epsilon > 0$ [37]. In the limit $\epsilon \to 0$, the above inequalities can be cast in the following form:

$$ \Pr\left( \left| \frac{1}{N} \sum_{i=1}^{N} X_i^2 - 1 \right| \geq \epsilon \right) \leq 2e^{-\frac{\epsilon^2}{4}}. $$

https://doi.org/10.1088/1742-5468/ad13fc
Using \(|N^{-1}\sum_i \eta_i \delta_i| = |(2N)^{-1}\sum_i X_i^2 - 1 + 1 - Y_i^2|\), we obtain
\[
\Pr \left( \left| \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i \right| \geq \epsilon \right) \leq \sqrt{2e^{-\frac{\epsilon^2}{2}}}
\] (18)
in the limit of large \(N\). For further details on the derivation of equation (18), see appendix A.

2.3. Principal curvature

With the form of random Gaussian projections in hand, we now analyze the principal curvatures in both the original and lower-dimensional spaces. The Hessian associated with the two-dimensional loss projection (12) is
\[
H_{\alpha,\beta} = \left( \begin{array}{cc}
\frac{\partial^2 L}{\partial \theta_i \partial \theta_j} & \frac{\partial^2 L}{\partial \theta_i \partial \eta_j} \\
\frac{\partial^2 L}{\partial \theta_i \partial \eta_j} & \frac{\partial^2 L}{\partial \eta_i \partial \eta_j}
\end{array} \right) = \left( \begin{array}{cc}
\delta_{ij} \delta_{ij} & \delta_{ij} \delta_{ij} \\
\delta_{ij} \delta_{ij} & \delta_{ij} \delta_{ij}
\end{array} \right) = \left( \begin{array}{cc}
(H_{\theta})_{ij} \eta^i \eta^j & (H_{\theta})_{ij} \eta^i \delta^j \\
(H_{\theta})_{ij} \delta^i \eta^j & (H_{\theta})_{ij} \delta^i \delta^j
\end{array} \right),
\] (19)
where we use Einstein notation in the last equality.

In analogy with equation (9), the dimension-reduced Hessian \(H_{\alpha,\beta}\) is connected to the corresponding FIM \(F_{\alpha,\beta}\) according to
\[
H_{\alpha,\beta} = F_{\alpha,\beta} - \frac{1}{T} \sum_{t=1}^{T} \sum_{k=1}^{C} \left( y_k(t) - f_{(\alpha,\beta),k}(t) \right) \nabla_{(\alpha,\beta)} f_{(\alpha,\beta),k}(t),
\] (20)
where \(f_{(\alpha,\beta),k}(t)\) is the \(k\)th entry of the output of a neural network with parameters \(\theta^* + \alpha \eta + \beta \delta\). The operator \(\nabla_{(\alpha,\beta)}\) denotes the gradient w.r.t. parameters \(\alpha, \beta\). Recall that equation (9) (and hence equation (20)) is based on a mean squared loss function. One may also use equation (10) to obtain a connection between the dimension-reduced Hessian and the corresponding cross-entropy FIM.

Lower-dimensional approximations \(H_{\alpha,\beta}\) of the Hessian \(H_{\theta}\) in the original high-dimensional space might be useful in generalized Gauss–Newton algorithms that are based on positive semidefinite and computationally affordable Hessian approximations [38, 39].

Because the elements of \(\delta, \eta\) arising in \(H_{\alpha,\beta}\) are distributed according to a standard normal distribution, the second derivatives of the loss function \(L\) in equation (19) have prefactors that are products of standard normal variables and hence can be expressed as sums of chi-squared distributed random variables, as in equation (13). To summarize, the elements of \(H_{\alpha,\beta}\) are the sums of the second derivatives of \(L\) in the original space weighted with chi-squared distributed prefactors.

The principal curvatures \(\kappa_{\pm}^{\alpha,\beta}\) (i.e. the eigenvalues of \(H_{\alpha,\beta}\)) are
\[
\kappa_{\pm}^{\alpha,\beta} = \frac{1}{2} \left( A + C \pm \sqrt{4B^2 + (A - C)^2} \right),
\] (21)
https://doi.org/10.1088/1742-5468/ad13fc
where $A = (H_\theta)_{ij} \eta^i \eta^j$, $B = (H_\theta)_{ij} \eta^i \delta^j$, and $C = (H_\theta)_{ij} \delta^i \delta^j$. To the best of our knowledge, a closed analytical expression for the distribution of the quantities $A, B, C$ is not yet known [37, 40–42].

We now turn to the relationship between random projections and principal curvature. The appeal of random projections is that pairwise distances between points in a high-dimensional space can be nearly preserved by a lower-dimensional linear embedding, affording a low-dimensional representation of mean and variance information with minimal distortion [43]. The question is whether random projections also preserve curvature information, and if so, what the nature of the relationship is between random Gaussian directions and principal curvature. For instance, Li et al [13] asserted that ‘the principal curvatures of a dimensionality-reduced plot (with random Gaussian directions) are weighted averages of the principal curvatures of the full-dimensional surface’. However, our results show that the principal curvatures $\kappa^i_{\pm}$ in a two-dimensional loss projection are weighted averages of the Hessian elements $(H_\theta)_{ij}$ in the original space, and not weighted averages of the principal curvatures $\kappa^i_{\pm}$, which instead are solutions of an $N$th degree polynomial. Similar arguments apply to projections with dimensions larger than 2.

In section 3, we discuss examples that show that lower-dimensional projections of $L(\theta)$ can be misleading because high-dimensional saddle points may appear to be minima, maxima, or almost flat regions, depending on the index of the underlying Hessian $H_\theta$ in the original space.

Returning to principal curvature, since $\sum_{i,j} a_{ij} \eta^i \eta^j = \sum_i a_{ii} \eta^i \eta^i + \sum_{i \neq j} a_{ij} \eta^i \eta^j$ ($a_{ij} \in \mathbb{R}$), we obtain $E[A] = E[C] = (H_\theta)^i_i$ and $E[B] = 0$, where $(H_\theta)^i_i \equiv \text{tr}(H_\theta) = \sum_{i=1}^N \kappa^i_\theta$. To show that the expected values of $a_{ij} \eta^i \eta^j$ ($i \neq j$) or $a_{ij} \eta^i \delta^j$ vanish, one can either invoke independence of $\eta^i, \eta^j$ ($i \neq j$) and $\eta^j, \delta^j$ or transform both products into corresponding differences of two chi-squared random variables with the same mean (see equation (13)).

Hence, the expected dimension-reduced Hessian (19) is

$$E[H_{\alpha,\beta}] = \begin{pmatrix} (H_\theta)^i_i & 0 \\ 0 & (H_\theta)^j_j \end{pmatrix}.$$  \hspace{1cm} (22)

Therefore, the corresponding eigenvalue (or principal curvature) $\bar{\kappa}^\alpha_{\alpha,\beta}$ is given by the sum of all principal curvatures in the original space (i.e. $\bar{\kappa}^\alpha_{\alpha,\beta} = \sum_{i=1}^N \kappa^i_\theta$). Hence, the value of the principal curvature $\bar{\kappa}^\alpha_{\alpha,\beta}$ in the expected dimension-reduced space will be either positive (if the positive principal curvatures in the original space dominate), negative (if the negative principal curvatures in the original space dominate), or close to zero (if the positive and negative principal curvatures in the original space cancel each other out).

As a result, saddle points will not appear as such in the expected random projection.

In addition to the connection between $\bar{\kappa}^\alpha_{\alpha,\beta}$ and the principal curvatures $\kappa^i_\theta$, we now provide an overview of additional mathematical relations between different curvature

---

6 The expected values of the quantities $A, B, C$ correspond to ensemble means (29) over $S$ realizations of random projections in the limit $S \to \infty$. 

https://doi.org/10.1088/1742-5468/ad13fc
Table 1. Overview of several quantities that are useful to characterize curvature of high-dimensional loss functions. We summarized the employed definitions of Hessians and several derived curvature measures that are useful to quantify the curvature of \( N \)-dimensional loss functions \( L(\theta^*) : \mathbb{R}^N \to \mathbb{R} \) and their two-dimensional random projections \( L(\theta^* + \alpha \eta + \beta \delta) \) at a nondegenerate critical point \( \theta^* \). The vectors \( \eta, \delta \in \mathbb{R}^N \) are the directions along which we project the original loss function, and \( \alpha, \beta \in \mathbb{R} \) are the corresponding scale factors. Random loss projections are usually based on Gaussian vectors \( \eta, \delta \), which are almost orthogonal in high-dimensional spaces (i.e. for large \( N \)).

| Symbol | Definition |
|--------|------------|
| \( H_\theta \in \mathbb{R}^{N \times N} \) | Hessian in the original loss space |
| \( \kappa_i^\theta \in \mathbb{R} \) | Principal curvatures in the original loss space, where \( i \in \{1, \ldots, N\} \) (i.e. the eigenvalues of \( H_\theta \)) |
| \( H_{\alpha, \beta} \in \mathbb{R}^{2 \times 2} \) | Hessian in a two-dimensional projection of an \( N \)-dimensional loss function |
| \( \kappa_{\alpha, \beta}^\pm \in \mathbb{R} \) | Principal curvatures in a two-dimensional loss projection (i.e. the eigenvalues of \( H_{\alpha, \beta} \)) |
| \( \bar{\kappa}_{\alpha, \beta} \in \mathbb{R} \) | Principal curvature in the expected two-dimensional loss projection (i.e. the eigenvalue of \( \mathbb{E}[H_{\alpha, \beta}] \)) |
| \( H \in \mathbb{R} \) | Mean curvature (i.e. \( \sum_{i=1}^{N} \kappa_i^\theta / N = \bar{\kappa}_{\alpha, \beta} / N \)) |

measures that are useful for quantifying the curvature properties of high-dimensional loss functions and their two-dimensional random projections.

Invoking equation (21), we can relate \( \bar{\kappa}_{\alpha, \beta} \) to \( \text{tr}(H_{\theta}) \) and \( \kappa_{\alpha, \beta}^\pm \). Because \( \kappa_+ \alpha, \beta + \kappa_- \alpha, \beta = A + C \), we have

\[
\text{tr}(H_{\theta}) = \bar{\kappa}_{\alpha, \beta} = \sum_{i=1}^{N} \kappa_i^\theta = \frac{1}{2} \left( \mathbb{E} \left[ \kappa_+ \alpha, \beta \right] + \mathbb{E} \left[ \kappa_- \alpha, \beta \right] \right). \tag{23}
\]

The mean curvature \( H \) in the original space is related to \( \bar{\kappa}_{\alpha, \beta} \) via

\[
H = \frac{1}{N} \bar{\kappa}_{\alpha, \beta} = \frac{1}{N} \sum_{i=1}^{N} \kappa_i^\theta. \tag{24}
\]

We summarize the definitions of the employed Hessians and curvature measures in table 1.

2.4. Hessian trace estimates

Finally, we point to a connection between the above curvature measures and existing Hessian trace estimates. The trace (or unnormalized mean curvature) of the Hessian \( H_{\theta} \) has already found applications in characterizing loss landscapes of neural networks [21, 44]. A common way of estimating \( \text{tr}(H_{\theta}) \) without explicitly computing all eigenvalues of \( H_{\theta} \) is based on Hutchinson’s method [19] and random numerical linear algebra.
The basic idea behind this approach is to (a) use a random vector $z \in \mathbb{R}^N$ with elements $z_i$ that are distributed according to a distribution function with zero mean and unit variance (e.g. a Rademacher distribution with $\Pr(z_i = \pm 1) = 1/2$), and (b) compute $z^\top H_\theta z$, an unbiased estimator of $\text{tr}(H_\theta)$, using HVPs. That is,

$$\text{tr}(H_\theta) = \mathbb{E}[z^\top H_\theta z].$$

Recall that equation (23) shows that the principal curvature of the expected random loss projection, $\kappa^{\alpha,\beta}$, is equal to $\text{tr}(H_\theta)$. Instead of estimating $\text{tr}(H_\theta)$ using the original Hutchinson’s method (25), an alternative Hutchinson-type estimate is provided by the mean of the expected values of $\kappa^{\alpha,\beta}_-$ and $\kappa^{\alpha,\beta}_+$ (see equation (23)).

3. Illustrative examples

In this section, we summarize the main concepts derived in section 2 in terms of different examples. These examples are intended to aid intuition on how (a) principal and mean curvatures of the original loss function relate to those of random projections, (b) curvature-based Hessian trace estimates relate to those obtained with the original Hutchinson’s method, and (c) Hessian directions (i.e. the eigenbasis of the Hessian $H_\theta$ of $L(\theta)$) can be used for a guided visual analysis of saddle information in high-dimensional spaces.

3.1. Extracting curvature information

We now study two examples that will help build intuition for the concepts discussed in the previous sections. In the first example, we study a critical point $\theta^*$ of an $N$-dimensional loss function $L(\theta)$ for which (a) all principal curvatures have the same magnitude and (b) the number of positive curvature directions is equal to the number of negative curvature directions. The mean curvature of this saddle point is zero. In accordance with equation (24), we will show that the mean curvature can be approximated by a function of ensemble means of elements of the dimension-reduced Hessian $H_{\alpha,\beta}$, and that the principal curvatures $\kappa^{\alpha,\beta}_\pm$ are able to correctly identify the saddle point in the majority of simulated projections. In the second example, we use a loss function associated with an unequal number of negative and positive curvature directions. For the different curvature measures derived in section 2.3, we find that random projections cannot identify the underlying saddle point.

The loss function of our first example is

$$L(\theta) = \frac{1}{2}\theta_{2n+1} \left( \sum_{i=1}^{n} \theta_i^2 - \theta_{i+n}^2 \right), \quad n \in \mathbb{Z}_+,$$

(26)
where we set $N = 2n + 1$. A critical point $\theta^*$ of the loss function (26) satisfies

$$(\nabla_\theta L)(\theta^*) = \begin{pmatrix} \theta_1^* \theta_{2n+1}^* \\ \vdots \\ \theta_n^* \theta_{2n+1}^* \\ -\theta_n^* \theta_{2n+1}^* \\ \vdots \\ \frac{1}{2} \left( \sum_{i=1}^{n} \theta_i^{2} - \theta_{i+n}^{2} \right) \end{pmatrix} = 0.$$ (27)

The Hessian at the critical point $\theta^* = (\theta_1^*, \ldots, \theta_{2n}^*, \theta_{2n+1}^*) = (0, \ldots, 0, 1)$ is

$$H_\theta = \text{diag}(1, \ldots, 1, -1, \ldots, -1, 0).$$ (28)

Because $H_\theta$ has positive and negative eigenvalues, the critical point is a saddle. The corresponding principal curvatures are $\kappa_1^\theta \in \{-1, 1\}$ ($i \in \{1, \ldots, N - 1\}$) and $\kappa_N^\theta = 0$. In this example, the mean curvature $H$, as defined in equation (24), is equal to 0. According to equation (22), the principal curvature, $\bar{\kappa}_{\alpha, \beta}$, associated with the expected, dimension-reduced Hessian $H_{\alpha, \beta}$ is also equal to 0, erroneously indicating an apparently flat loss landscape if one would use $\bar{\kappa}_{\alpha, \beta}$ as the main measure of curvature. To compare the convergence of different curvature measures as a function of the number of loss projections $S$, we will now study the ensemble mean

$$\langle X \rangle = \frac{1}{S} \sum_{k=1}^{S} X^{(k)}.$$ (29)

of different quantities of interest $X$, such as Hessian elements and principal curvature measures in the dimension-reduced space. Here, $X^{(k)}$ is the $k$th realization (or sample) of $X$.

We first study the dependence of the ensemble means $\langle (H_{\alpha, \beta})_{ij} \rangle$ ($i, j \in \{1, 2\}$) of the elements of the dimension-reduced Hessian $H_{\alpha, \beta}$ on the number of samples $S$. According to equation (22), the diagonal elements of $\mathbb{E}[H_{\alpha, \beta}]$ are proportional to the mean curvature of the original high-dimensional space and are thus useful for examining the curvature properties of the high-dimensional loss functions. Figure 1(a) shows the convergence of the ensemble means $\langle (H_{\alpha, \beta})_{ij} \rangle$ toward the expected values $\mathbb{E}[(H_{\alpha, \beta})_{ij}]$ as a function of $S$. Note that $\mathbb{E}[(H_{\alpha, \beta})_{ij}] = 0$ for all $i, j$. For a few dozen loss projections, the deviations of some of the ensemble means from the corresponding expected values reach values larger than 20. A relatively large number of loss projections $S$ between $10^3$ and $10^4$ is required to keep these deviations at values smaller than approximately 2–4. The solid black and red lines in figure 1(b), respectively, show the ensemble means $\langle \kappa_{\pm, \alpha, \beta} \rangle$ and the eigenvalues

$$\langle \kappa_{\pm, \alpha, \beta} \rangle = \frac{1}{2} \left( \langle A \rangle + \langle C \rangle \pm \sqrt{4 \langle B \rangle^2 + (\langle A \rangle - \langle C \rangle)^2} \right).$$ (30)
Figure 1. Convergence of the ensemble mean (29) of Hessian elements and curvature measures as a function of the number of random projections $S$. (a), (c) The deviation of the ensemble means $\langle (H_{\alpha,\beta})_{ij} \rangle$ $\langle i,j \rangle \in \{1,2\}$ of Hessian elements from the corresponding expected values as a function of $S$. Notice that the expected value of the diagonal elements $(H_{\alpha,\beta})_{11}$ and $(H_{\alpha,\beta})_{22}$ is equal to $\bar{\kappa}_{\alpha,\beta}$ (i.e. to the sum of principal curvatures in the original space) (see equations (22) and (23)). A relatively large number of random projections between $10^3$ and $10^4$ is required to keep the deviations at values smaller than about 2–4. (b), (d) The ensemble means $\langle \bar{\kappa}_{\alpha,\beta} \rangle$ (see equation (21)) and $\langle \bar{\tilde{\kappa}}_{\alpha,\beta} \rangle$ (see equation (30)) as a function of $S$. Dashed gray lines represent $\bar{\kappa}_{\alpha,\beta} = \text{tr}(H_{\theta})$. In panels (a), (b) and (c), (d), the $N$-dimensional loss functions are given by equations (26) and (31), respectively. We evaluate the corresponding Hessians (28) and (32) at the saddle point $\theta^* = (\theta_{1}^*, ..., \theta_{2n}^*, \theta_{2n+1}^*) = (0, ..., 0, 1)$. In both loss functions, we set $n = 500$, and in loss function (31) we set $\tilde{n} = 800$. 

of the ensemble-averaged dimension-reduced Hessian as a function of $S$. Since $\text{E}[A] = \text{E}[C] = (H_{\theta})^i_i$ and $\text{E}[B] = 0$ (see equation (22)), we have that $\langle \bar{\tilde{\kappa}}_{\alpha,\beta} \rangle = \bar{\kappa}_{\alpha,\beta}$ in the limit $S \to \infty$. In the current example, the ensemble means $\langle \bar{\tilde{\kappa}}_{\alpha,\beta} \rangle$ thus approach $\bar{\kappa}_{\alpha,\beta} = 0$ for large numbers of samples $S$, represented by the dashed red lines in figure 1(b). The ensemble means $\langle \bar{\kappa}_{\alpha,\beta} \rangle$ converge toward values of opposite signs, indicating a saddle point.

For a sample size of $S = 10^4$, we show the distribution of the principal curvatures $\kappa_{\alpha,\beta}^\pm$ in figure 2(a). We observe that the distributions are plausibly Gaussian. We also calculate the probability $\text{Pr}(\kappa_{\alpha,\beta}^+ \kappa_{\alpha,\beta}^- > 0)$ that the critical point in the lower-dimensional random projection does not appear as a saddle (i.e. $\kappa_{\alpha,\beta}^+ \kappa_{\alpha,\beta}^- > 0$). For the example shown in figure 2(a), we found that $\text{Pr}(\kappa_{\alpha,\beta}^+ \kappa_{\alpha,\beta}^- > 0) \approx 0.3$. That is, in approximately 30% of
Figure 2. Distribution of principal curvatures $\kappa_\pm^{\alpha,\beta}$ (red bars) and $\kappa_+^{\alpha,\beta}$ (black bars). In panels (a) and (b), the loss functions are given by equations (26) and (31), respectively. We evaluate the corresponding Hessians (28) and (32) at the saddle point $\theta^* = (\theta^*_1, \ldots, \theta^*_n, \theta^*_{2n+1}) = (0, \ldots, 0, 1)$. In both loss functions, we set $n = 500$, and in loss function (31) we set $\tilde{n} = 800$. In panel (a) the probability $\Pr(\kappa_+^{\alpha,\beta} - \kappa_-^{\alpha,\beta} > 0)$ that the critical point in the lower-dimensional random projection does not appear as a saddle is about 0.3, and it is 1 in panel (b). Histograms are based on 10,000 random projections that are used to compute $\kappa_{\pm}^{\alpha,\beta}$. Solid gray lines indicate Gaussian approximations of the empirical distributions.

The simulated projections, the lower-dimensional loss landscape wrongly indicates that it does not correspond to a saddle.

Our first example, which is based on the loss function (26), shows that the principal curvatures in the lower-dimensional representation of $L(\theta)$ may capture the saddle behavior in the original space if one computes the ensemble means $\langle \kappa_{\pm}^{\alpha,\beta} \rangle$ in the lower-dimensional space (figure 1(b)). However, if one first calculates the ensemble means of the elements of the dimension-reduced Hessian $H_{\alpha,\beta}$ to infer $\langle \tilde{\kappa}_{\pm}^{\alpha,\beta} \rangle$, the loss landscape appears to be flat in this example. We thus conclude that different ways of computing ensemble means (either before or after calculating the principal curvatures) may lead to different results with respect to the ‘flatness’ of a dimension-reduced loss landscape.

In the next example, we show that random projections cannot identify certain saddle points regardless of the underlying averaging process. We consider the loss function

$$L(\theta) = \frac{1}{2} \theta_{2n+1} \left( \sum_{i=1}^{\tilde{n}} \theta_i^2 - \sum_{i=\tilde{n}+1}^{2n} \theta_i^2 \right) , \quad n \in \mathbb{Z}^+, n < \tilde{n} \leq 2n ,$$

where we use the convention $\sum_{i=a}^{b} (\cdot) = 0$ if $a > b$. The Hessian at the critical point $(\theta^*_1, \ldots, \theta^*_n, \theta^*_{2n+1}) = (0, \ldots, 0, 1)$ is

$$H_\theta = \text{diag}(1, \ldots, 1, -1, \ldots, -1, 0) .$$

As in the previous example, the critical point is again a saddle, but the mean curvature is $H = 2(\tilde{n} - n)/N > 0$. In the following numerical experiments, we set $n = 500$ and $\tilde{n} = 800$. Figure 1(c) shows that the ensemble means $\langle (H_{\alpha,\beta})_{ij} \rangle$ converge toward the expected value $\mathbb{E}[(H_{\alpha,\beta})_{ij}]$ as the number of samples increases. We again observe that a
relatively large number of random loss projections $S$ between $10^3$ and $10^4$ is required to keep the deviations of ensemble means from their corresponding expected values small. Because of the dominance of positive principal curvatures $\kappa^\theta_i$ in the original space, the corresponding ensemble means of principal curvatures (i.e. $\langle \kappa^\alpha_+ \rangle$, $\langle \kappa^\alpha_- \rangle$) in the lower-dimensional representation approach positive values (figure 1(d)). The distribution of $\kappa^\alpha_\pm$ indicates that the probability of observing a saddle in the lower-dimensional loss landscape is vanishingly small (figure 2(b)). In this second example, both ways of computing ensemble means, before and after calculating the lower-dimensional principal curvatures, mistakenly suggest that the saddle in the original space is a minimum in the dimension-reduced space.

To summarize, for both loss functions (26) and (31), the saddle point $\theta^*_i = (0,\ldots,0,1)$ in the original loss function $L(\theta)$ is often misrepresented in lower-dimensional representations $L(\theta + \alpha \eta + \beta \delta)$ if random directions are used. Depending on (a) the employed curvature measure and (b) the index of the underlying Hessian $H_\theta$ in the original space, the saddle $\theta^*_i = (0,\ldots,0,1)$ often appears erroneously as either a minimum, maximum, or an almost flat region.

If the critical point were a minimum or maximum (i.e. a critical point associated with a positive definite or negative definite Hessian $H_\theta$), it would be correctly represented in the corresponding expected random projection because the sign of its principal curvature $\bar{\kappa}_i^\alpha$, which is proportional to the sum of all eigenvalues $\kappa^\theta_i$ of the Hessian $H_\theta$ in the original loss space, would be equal to the sign of the principal curvatures $\kappa^\theta_i$. However, such points are scarce in high-dimensional loss spaces [17, 18].

Finally, because of the confounding factors associated with the inability of random projections to correctly identify saddle information, it does not appear advisable to use ‘flatness’ around a critical point in a lower-dimensional random loss projection as a measure of generalization error [13].

### 3.2. Hessian trace

We now use the loss functions (26) and (31) to compare the convergence behavior between the original Hutchinson method (25) and the curvature-based trace estimation (23). Instead of two random directions, we use one random Gaussian direction $\eta$ and perform least-squares fits for 50 equidistant values of $\alpha$ in the interval $[-0.05,0.05]$ to extract estimates of $\text{tr}(H_\theta)$ from $L(\theta^* + \alpha \eta)$. We use the same random Gaussian directions in Hutchinson’s method.

Figure 3 shows how the Hutchinson and curvature-based trace estimates converge toward the true trace values: 0 for loss function (26) and 600 for loss function (31) with $n = 500$ and $\tilde{n} = 800$. Given that we use the same random vectors in both methods, their convergence behavior toward the true trace value is similar. Using the curvature-based method, one can produce Hutchinson-type trace estimates without computing HVPs. However, it requires the user to specify an appropriate interval for $\alpha$ so that the mean curvature can be properly estimated in a quadratic-approximation regime. This also requires a sufficiently large number of points in this interval. Therefore, this method may be less accurate than the original Hutchinson method.
Figure 3. Estimating the trace of the Hessian $H_\theta$. In panels (a) and (b), the loss functions are given by equations (26) and (31), respectively. We evaluate the corresponding Hessians (28) and (32) at the saddle point $\theta^s = (\theta_1^*, ..., \theta_{2n}^*, \theta_{2n+1}^*) = (0, ..., 0, 1)$. In both loss functions, we set $n = 500$, and in loss function (31) we set $\tilde{n} = 800$. The solid black and dashed–dotted red lines represent Hutchinson ($\langle z^\top H_\theta z \rangle$; see equation (25)) and the curvature-based ($\langle \kappa^\alpha \rangle$) estimates of $\text{tr}(H_\theta)$, respectively. We compute the ensemble means $\langle \cdot \rangle$ as defined in equation (29) for different numbers of random projections $S$. The trace estimates in panels (a) and (b), respectively, converge toward the true trace values $\text{tr}(H_\theta) = 0$ and $\text{tr}(H_\theta) = 600$, which are indicated by dashed gray lines. In both methods, the same random vectors with elements that are distributed according to a standard normal distribution are used. For the curvature-based estimation of $\text{tr}(H_\theta)$, we perform least-squares fits of $L(\theta^s + \alpha \eta)$ over an interval $\alpha \in [-0.05, 0.05]$.

3.3. Hessian directions

Given the described shortcomings of random projections (e.g. in correctly identifying saddles of high-dimensional loss functions), we suggest using Hessian directions (i.e. the eigenbasis of $H_\theta$) as directions $\eta, \delta$ in $L(\theta^s + \alpha \eta + \beta \delta)$. For the loss function (31) with $n = 900, \tilde{n} = 1000$, we show projections along different Hessian directions in figures 4(a)–(c). We observe that different Hessian directions indicate different types of critical points in a dimension-reduced space. If the eigenvalues associated with the Hessian directions $\eta, \delta$ have different signs, the corresponding lower-dimensional loss landscape is a saddle (figure 4(a)). If both eigenvalues have the same sign, the loss landscape is either a minimum (figure 4(b); both signs are positive) or a maximum (figure 4(c); both signs are negative). If one uses a random projection instead, the resulting lower-dimensional loss landscape often appears to be a minimum in this example (figure 4(d)). To quantify the proportion of random projections that correctly identify the saddle with $n = 900, \tilde{n} = 1000$, we generated 10 000 realizations of $\kappa^\alpha, \beta$ (see equation (21)). We found that the signs of $\kappa^\alpha, \beta$ are different in only approximately 0.5% of all simulated realizations. In this example, the principal curvatures $\kappa^\alpha, \beta$ indicate a saddle in only approximately 0.5% of the studied projections.

In the next section, we show that Hessian directions that are associated with the largest-magnitude positive and negative eigenvalues of $H_\theta$ are useful for appropriately identifying saddle information and visually studying the geometric properties of saddle points in high-dimensional loss spaces of image classifiers.
Figure 4. Dimensionality-reduced loss $L(\theta^* + \alpha\eta + \beta\delta)$ of equation (31) with $n = 900, \tilde{n} = 1000$ for different directions $\eta, \delta$. (a)–(c) The directions $\eta, \delta$ correspond to eigenvectors of the Hessian $H_\theta$ of equation (31). If the eigenvalues associated with $\eta, \delta$ have different signs, the corresponding loss landscape is a saddle, as depicted in panel (a). If the eigenvalues associated with $\eta, \delta$ have the same sign, the corresponding loss landscape is either a minimum (both signs are positive), as shown in panel (b), or a maximum (both signs are negative), as shown in panel (c). Because there is an excess of $\tilde{n} - n = 100$ positive eigenvalues in $H_\theta$, a projection onto a dimension-reduced space that is spanned by the random directions $\eta, \delta$ is often associated with an apparently convex loss landscape. An example of such an apparent minimum is shown in panel (d). We selected a single pair of random directions $\eta, \delta$ (i.e. no averaging over random directions has been performed).

4. Applications to neural networks

We now compare projections $L(\theta^* + \alpha\eta + \beta\delta)$ using (a) dominant Hessian directions (i.e. eigenvectors that are associated with the largest-magnitude positive and negative eigenvalues of $H_\theta$) and (b) random directions. The Hessian directions are computed using HVPs without an explicit representation of $H_\theta$ (see algorithm 1). We first computed the largest-magnitude eigenvalue and then used an annihilation method [47] to compute the second largest-magnitude eigenvalue of the opposite sign. More details on the annihilation algorithm are provided in appendix B. Other deflation techniques can be used to find additional Hessian directions.

Instead of explicitly representing the Hessian $H_\theta$, algorithm 1 evaluates products between $H_\theta$ and a vector $v \in \mathbb{R}^N$ using the identity

$$\nabla_\theta \left( (\nabla_\theta L)^\top v \right) = (\nabla_\theta \nabla_\theta L) v + (\nabla_\theta L)^\top \nabla_\theta v = H_\theta v. \tag{33}$$

In the first step, the gradient $(\nabla_\theta L)^\top$ is computed using reverse-mode autodifferentiation (AD) to then compute the scalar product $((\nabla_\theta L)^\top v)$. In the second step, we again apply reverse-mode AD to the computational graph associated with the scalar product $((\nabla_\theta L)^\top v)$. Because the vector $v$ does not depend on $\theta$ (i.e. $\nabla_\theta v = 0$), the result is $H_\theta v$. One may also use forward-mode AD in the second step to provide a more memory-efficient implementation$^7$.

We used the LinearOperator module available in the Python package scipy to represent HVPs. As an alternative to implementing HVPs manually, one may use the hvp

$^7$ Different forward-mode AD functions have been made available in PyTorch since version 1.11, starting from March 2022.
function provided in the `torch.autograd.functional` module. Moreover, the Python package `PyHessian` [21] can be used to evaluate Hessians of high-dimensional functions in a distributed manner. For the computation of dominant Hessian directions, we used the `scipy` function `eigsh`, which is based on the implicitly restarted Lanczos method [48].

The computational cost of algorithm 1 can be summarized as follows. For the HVPs, the computational cost is $\mathcal{O}(N)$ [49], where $N$ is the number of neural-network parameters. To identify extremal eigenvalues along with their corresponding eigenvectors using a Lanczos-type method, one has to perform a certain number of matrix–vector multiplications. The required number of iterations depends on several factors, including the desired accuracy, characteristics of the underlying Hessian matrix, and choice of the initial vector.

We first focus on a ResNet-56 architecture that has been trained in [13] on CIFAR-10 using stochastic gradient descent (SGD) with Nesterov momentum. The number of parameters is 855,770. The training and test losses at the local optimum found by SGD are $9.20 \times 10^{-4}$ and 0.29, respectively; the corresponding accuracies are 100.00 and 93.66, respectively.

**Algorithm 1.** Compute dominant Hessian directions.

```
1: L_1 = LinearOperator((N,N), matvec = hvp)  ▷ Initialize linear operator for HVP calculation
2: eigval1, eigvec1 = solve_eigsh(L_1)  ▷ Compute the largest-magnitude eigenvalue and corresponding eigenvector associated with operator $L_1$.
3: shifted_hvp(vec) = hvp(vec) - eigval1*vec  ▷ Define shifted HVP
4: L_2 = LinearOperator((N,N), matvec = shifted_hvp)  ▷ Initialize linear operator for shifted HVP calculation
5: eigval2, eigvec2 = solve_eigsh(L_2)  ▷ Compute the largest-magnitude eigenvalue and corresponding eigenvector associated with operator $L_2$.
6: eigval2 += eigval1
7: if eigval1 >= 0 then
8:   maxeigval, maxeigvec = eigval1, eigvec1
9: else
10:   maxeigval, maxeigvec = eigval2, eigvec2
11: if eigval1 >= 0 then
12:   mineigval, mineigvec = eigval1, eigvec1
13: else
14:   mineigval, mineigvec = eigval2, eigvec2
15: return maxeigval, maxeigvec, mineigval, mineigvec
```

In accordance with [13], we apply filter normalization to random directions. This and related normalization methods are often employed when generating random projections. One reason is that simply adding random vectors to the parameters of a neural-network loss function (or parameters of other functions) does not consider the range of parameters associated with different elements of that function. As a result, random perturbations may be too small or large to properly resolve the influence of certain parameters on a given function.

When calculating Hessian directions, we are directly taking into account the parameterization of the underlying functions that we want to visualize. Therefore, there is no need for additional rescaling of different parts of the perturbation vector. Still, reparameterizations of a neural network can result in changes in curvature properties (see, e.g. theorem 4 in [11]).
Visualizing high-dimensional loss landscapes with Hessian directions

Figure 5. Loss landscape projections for ResNet-56. (a), (c) The projection directions $\eta, \delta$ are given by the eigenvectors associated with the largest and smallest eigenvalues of the Hessian $H_{\theta}$, respectively. The zoomed inset in panel (c) shows the loss landscape for $(\alpha, \beta) \in [-0.01,0.005] \times [-0.05,0.05]$. We observe a decreasing loss along the negative $\beta$-axis. (b), (d) The projection directions $\eta, \delta$ are given by random vectors. We selected a single pair of random directions $\eta, \delta$ (i.e. no averaging over random directions has been performed). The domains of $(\alpha, \beta)$ in panels (a), (b) and (c), (d) are $[-1,1] \times [-1,1]$ and $[-0.05,0.05] \times [-0.05,0.05]$, respectively. All shown cross-entropy loss landscapes are based on evaluating the CIFAR-10 training dataset that consists of 50 000 images.

Figure 5 shows the two-dimensional projections of the loss function (cross-entropy loss) around a local critical point. The smallest and largest eigenvalues are $-16.4$ and $5007.9$, respectively. This means that the found critical point is a saddle with a maximum negative curvature that is more than two orders of magnitude smaller than the maximum positive curvature at that point. The saddle point is clearly visible in figures 5(a) and (c). We observe in the zoomed inset in figure 5(c) that the loss decreases along the negative $\beta$-axis.

If random directions are used, the corresponding projections indicate that the optimizer converged to a local minimum and not to a saddle point (figures 5(b) and (d)). Overall, the ResNet-56 visualizations that we show in figure 5 exhibit structural similarities to those that we generated using the simple loss model (31) (figure 4(d)).

https://doi.org/10.1088/1742-5468/ad13fc
As a second neural-network structure, we consider DenseNet-121 that has also been trained in [13] on CIFAR-10 using SGD with Nesterov momentum. In this neural network, the number of parameters is 6956298. Figure 6 shows the projections \( L(\theta^* + \alpha \eta + \beta \delta) \) of the loss function (cross-entropy loss) around a local optimum. The training and test losses of the local optimum found by SGD are \( 8.07 \times 10^{-4} \) and \( 1.69 \times 10^{-1} \), respectively; the corresponding accuracies are 100.00 and 95.63, respectively. We again observe the typical saddle shape of \( L \) if dominant Hessian directions are used and an apparent local minimum for projections along random directions. One should keep in mind that the principal curvatures in the random direction plot are given by \( \kappa_{\pm \beta} \) (see equation (21)). At a critical point, the principal curvatures in dimension-reduced random-projection space are, on average, equal to the sum of the principal curvatures in the original space. If only very few small-magnitude negative principal curvatures are present in the original space, the expected random projections are associated with an apparent local minimum in dimension-reduced random-projection space.

Analyzing the Hessian \( H_\theta \) of DenseNet-121 around its local optimum, we found that the minimum and maximum principal curvatures are \(-109.1\) and \(1937.5\), respectively. In comparison to ResNet-56, the relative difference between these two curvatures is substantially smaller. In particular, the largest-magnitude negative principal curvature in DenseNet-121 is by a factor of about 7 larger than that in ResNet-56, indicating that more substantial loss improvements may be possible in the former by changing the neural-network parameters \( \theta \) along the negative curvature direction.

In figure 7, we compare changes in the projected loss along the random and dominant Hessian directions for DenseNet-121 and ResNet-56. In addition to evaluating the loss on training data (50,000 images) as in figures 5 and 6, we also provide a comparison to loss changes in the test dataset (10,000 images). The black crosses in figure 7 indicate loss minima. For both neural networks, we observe that random projections are associated with loss values that increase along both directions \( \delta, \eta \) and for both training and test data (figures 7(a), (b), (e), and (f)). For these projections, the loss minima are very close to the origin of the loss space. Different observations can be made if Hessian directions are employed in the loss projections. Figure 7(c) shows that the training loss minimum of DenseNet-121 in the Hessian projection is not located at the origin but at \((\alpha, \beta) \approx (0, 0.2)\). A value of \( \beta \approx 0.2 \) means that one has to move along the negative curvature direction to find a loss function value that is smaller than the one at the origin. We found that the value of the loss at that point is more than one order of magnitude smaller than the original loss at the origin or the smallest loss found in a random-projection plot. Similar observations can be made in the test dataset. However, the smallest loss value found in the Hessian direction plot is associated with a relatively large loss in the test dataset (figure 7(d)). Using Hessian directions to improve training and test performance requires one to balance potential performance improvements in the test and training datasets. Because of the relatively small negative principal curvature of ResNet-56, one can only achieve small loss improvements along the corresponding Hessian direction, and differences in minimum losses between random and Hessian projections are less pronounced (figures 7(e)–(h)).
Figure 6. Loss landscape projections for DenseNet-121. (a), (c) The projection directions $\eta, \delta$ are given by the eigenvectors associated with the largest and smallest eigenvalues of the Hessian $H_\theta$, respectively. (b), (d) The projection directions $\eta, \delta$ are given by random vectors. We selected a single pair of random directions $\eta, \delta$ (i.e. no averaging over random directions has been performed). The domains of $(\alpha, \beta)$ in panels (a), (b) and (c), (d) are $[-1, 1] \times [-1, 1]$ and $[-0.4, 0.4] \times [-0.4, 0.4]$, respectively. All shown cross-entropy loss landscapes are based on evaluating the CIFAR-10 training dataset, which consists of 50 000 images.

In appendix C, we study the random and Hessian loss projections of three additional architectures: the ResNet-56 without skip connections [13], MobileNetV2 [50], and GoogLeNet [50]. For ResNet-56 without skip connections, we found that the training loss minimum in the Hessian projection is associated with slightly larger training and test accuracies than the training loss minimum in the random projection. For MobileNetV2 and GoogLeNet, we found that the minimum training loss in the Hessian projection is associated with an either slightly larger training or test accuracy compared to the training loss minimum in the random projection. In the examples we study in appendix C, the training and test loss minima in the Hessian projections are better aligned than those in the corresponding random projections. Table 2 in appendix C summarizes the results for all five architectures.

Image classification architectures and their training are already highly optimized to achieve very good performance in common image classification tasks. Our results
Figure 7. Heatmaps of cross-entropy loss functions along random and dominant Hessian directions. (a)–(d) Loss heatmaps for DenseNet-121 ((a), (c) training data; (b), (d) test data). (e)–(h) Loss heatmaps for ResNet-56 ((e), (g) training data; (f), (h) test data). Random directions are used in panels (a), (b), (e), and (f), whereas Hessian directions are used in panels (c), (d), (g), and (h). The green and red regions indicate small and large loss values, respectively. Black crosses indicate the positions of loss minima in the shown domains.

show that it is possible to improve the training loss values along the dominant negative curvature direction. Smaller loss values may also be achievable with further gradient-based training iterations, so one has to consider tradeoffs between gradient...
and curvature-based optimization methods. In addition, assessing the overall performance of the dominant Hessian directions on this task is not straightforward. For instance, loss and accuracy measurements are not proportional to one another for this image classification task (see table 2 in appendix C). A further complication is that, because these architectures are highly optimized and the underlying generative processes are unknown, it is difficult to interpret variations in performance that may appear between training loss, test loss, and accuracy. Therefore, to better address the question of whether better performance in training by dominant Hessian directions can translate to better test performance, we need to control for some of the suspected confounds that appear in the image classification task. So, we also study possible loss improvements along the dominant negative curvature direction in a function approximation task [20] that we describe in appendix D.

Unlike image classification, function approximation involves sampling from a known generative distribution. Our results, summarized in appendix D, show that both training loss and test loss are smaller in our function approximation task for Hessian projections than for random projections. Specifically, for two-layer and ten-layer fully connected neural networks (FCNNs) with 20 501 and 101 301 parameters, respectively, training loss is more than 9% smaller for the two-layer network and 26% smaller for the ten-layer network for Hessian projections than for random projections, and test loss is 1% smaller and 6% smaller, respectively. Using this function approximation example, we have also included in the supplemental information [51] an animation that shows the evolution of random and Hessian loss projections during training.

5. Discussion and conclusion

Given that a global understanding of high-dimensional loss landscapes still remains very limited [1], one- and two-dimensional projections of such functions are a common tool to study the geometric properties of loss regions nearby critical points and along optimizer trajectories [12–16]. For an informed way of projecting high-dimensional functions to lower-dimensional representations, it is important to mathematically interpret the geometric information contained in such visualizations. To put it in the words of Li [52], ‘There are too many directions to project a high-dimensional data set and unguided plotting can be time-consuming and fruitless’. Therefore, it is important to understand how projection directions \( \eta, \delta \) affect the resulting loss visualizations. A standard choice in the literature is to use random Gaussian directions to obtain lower-dimensional loss representations \( L(\theta^* + \alpha \eta + \beta \delta) \). Combining concepts from high-dimensional probability and differential geometry, we show that saddle points in the original space may not be correctly identified as such in lower-dimensional representations if such random projections are used.

As formalized in equation (24), the expected principal curvature \( \bar{\kappa}^{\alpha, \beta} = \text{tr}(H_\theta) \), obtained by averaging over Hessian elements \( (H_\theta)_{ij} \) in a dimension-reduced Hessian (see equation (22)), is proportional to the mean curvature \( H = \text{tr}(H_\theta)/N \) in the original loss space. The connection between \( \bar{\kappa}^{\alpha, \beta} \) and \( \text{tr}(H_\theta) \) shows that Hutchinson-type trace estimates can be computed directly from curvature information in lower-dimensional random projections [19].

https://doi.org/10.1088/1742-5468/ad13fc
Depending on the value of the mean curvature in the original loss space, saddle points appear, on average, as either minima \((H > 0)\), maxima \((H < 0)\), or almost flat regions \((H \approx 0)\). Instead of computing ensemble means of elements of dimension-reduced Hessians, one may also compute ensemble means of principal curvatures. As illustrated in different numerical experiments, both ways of averaging are associated with different geometric interpretations, and random projections are generally not able to correctly identify saddle points. We therefore propose to study projections along dominant Hessian directions that are associated with the largest-magnitude positive and negative principal curvatures in the original space. Similar methods have been developed and applied in related work on exploratory data analysis [52]. Thus, our work may be viewed as part of a recent ‘Hessian turn’ in machine learning characterized by computationally tractable methods for unlocking Hessian information in nonlinear and high-dimensional settings [21, 22]. Projecting high-dimensional loss functions along Hessian directions provides a genuine way of illustrating their geometric properties around critical points. Negative curvature directions may be used as a starting point to further improve the performance of a given neural network. Analyzing principal curvatures (or eigenvalues of \(H_\theta\)) at critical points \(\theta^*\) is also useful to identify global minima. It has been shown by Cooper [1] that for any global minimum of \(L(\theta)\) in overparameterized neural networks, the corresponding Hessian \(H_\theta\) has \(rn\) positive eigenvalues, \(N - rn\) eigenvalues equal to 0, and no negative eigenvalues. Here, \(n\) and \(r\) are the number of input samples (i.e. data points) and the output dimension, respectively.

Our numerical experiments on different image classifiers and a function approximation task are in accordance with our reasoning and identify marked differences in the projections that are based on Hessian and random directions. Our results also address the ongoing debate on the connection between the flatness of a certain loss region and generalizability (i.e. the ability to effectively use trained neural networks on unseen data). Dinh et al [11] provided a detailed discussion on different mechanisms (e.g. reparameterization of flat regions) that indicate that sharp minima can achieve good generalization properties. Their work also highlighted the importance of clear definitions of flatness. Our study shows that flatness in randomly projected loss landscapes might be just apparent and a consequence of (large) principal curvatures of opposite sign that cancel out each other.

There are many interesting and worthwhile directions for future research building on the work reported here. To provide more insights into the geometric properties of high-dimensional loss functions of deep neural networks, future work may study other convolutional architectures [53, 54], vision transformers [55], and residual multilayer perceptrons [56]. Our results suggest that negative curvature directions may be useful for finding better local optima, but further research is needed to gain a clearer understanding of the tradeoffs associated with optimization protocols that are based on gradients and curvature information. Further research on learning algorithms that minimize loss functions along negative curvature directions [57] can contribute to the development of more effective optimizers. Such guided optimization algorithms may be of particular use in applications (e.g. neural network-based control [58–60] and function approximation in numerical analysis [20]), where training data can be generated with

https://doi.org/10.1088/1742-5468/ad13fc
Visualizing high-dimensional loss landscapes with Hessian directions

J. Stat. Mech. (2024) 023401

high resolution and used to approximate a function as closely as possible. In applications where one wishes to reduce the generalization error, the proposed mean-curvature estimation method may also be useful for improving existing curvature regularization approaches [61]. Furthermore, to improve our understanding of the properties of loss landscapes, it may be helpful to devote more attention to the study of the connections between random matrix theory and the characterization of high-dimensional functions [62, 63]. For example, for critical points of Gaussian fields in high-dimensional spaces, Bray and Dean [62] showed that there is a monotonic relationship between the Hessian index (i.e. the number of negative eigenvalues) and the loss value; the larger the index is, the larger is the corresponding loss. Bray and Dean [62] also showed that the eigenvalues of the Hessian at a critical point of such fields are distributed according to Wigner’s semicircle law [64] up to an additional shift. However, traditional random-matrix-theory results, such as Wigner’s semicircle law or the Marčenko–Pastur law [65], are based on various simplifying assumptions, including Gaussian weight, error, and data distributions, which do not hold in practical applications of neural networks [22].

Acknowledgments

We thank Mingtao Xia and Thomas Asikis for the helpful discussions and an anonymous reviewer for their valuable feedback and suggestions.

Funding

L B acknowledges funding from the Army Research Office through Grant W911NF-23-1-0129.

Appendix A. Concentration inequality

For \( \eta_i, \delta_i \sim \mathcal{N}(0,1) \) and \( X_i = \eta_i + \delta_i, Y_i = \eta_i - \delta_i \), our goal is to construct a concentration inequality for \( |N^{-1} \sum_i \eta_i \delta_i| = |(2N)^{-1} \sum_i X_i^2 - 1 + 1 - Y_i^2| = |(\tilde{X} - 1 + 1 - \tilde{Y})/2| \), where \( \tilde{X} = N^{-1} \sum_{i=1}^N X_i^2 \) and \( \tilde{Y} = N^{-1} \sum_{i=1}^N Y_i^2 \). Notice that \( \tilde{X} \) and \( \tilde{Y} \) are independent because the covariance associated with the corresponding bivariate normal distribution, which is valid in the limit of large \( N \), vanishes. Using the cumulative distribution function of the sum of independent random variables given by the convolution of the individual distributions, we obtain

https://doi.org/10.1088/1742-5468/ad13fc
Visualizing high-dimensional loss landscapes with Hessian directions

\[ \Pr \left( \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i \geq \epsilon \right) = \Pr \left( \frac{1}{2N} \sum_{i=1}^{N} X_i^2 - 1 + 1 - Y_i^2 \geq \epsilon \right) = \Pr \left( \tilde{X} - 1 + 1 - \tilde{Y} \geq 2\epsilon \right) = \int_{0}^{\infty} \Pr \left( \tilde{X} - 1 \geq 2\epsilon + \tilde{Y} - 1 | \tilde{Y} = y \right) f_{\tilde{Y}}(y) \, dy \leq \int_{0}^{\infty} e^{-N(2\epsilon - 1 + 1 \epsilon^2)} f_{\tilde{Y}}(y) \, dy, \quad (A1) \]

where we used inequality (15) for \( \epsilon \to 0 \) in the last step. In the limit of large \( N \), the distribution function \( f_{\tilde{Y}}(y) \) of the random variable \( \tilde{Y} \) approaches a Gaussian with mean 1 and variance \( 2/N \). With this approximation, we find

\[ \Pr \left( \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i \geq \epsilon \right) \leq e^{-\frac{N^2 \epsilon}{4}}. \quad (A2) \]

The same bound applies to \( \Pr \left( 1 - \tilde{X} + \tilde{Y} - 1 \geq 2\epsilon \right) \), so we have

\[ \Pr \left( \left| \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i \right| \geq \epsilon \right) \leq \sqrt{2e^{-\frac{N^2 \epsilon}{4}}}. \quad (A3) \]

For two independent random variables \( X, Y \), a similar bound can be derived using the inequality

\[ \Pr (X + Y \geq \epsilon) = \int_{\mathbb{R}^2} \mathbb{1}_{x+y \geq \epsilon} f_{X,Y}(x,y) \, dxdy = \int_{\mathbb{R}^2} \mathbb{1}_{x+y \geq \epsilon} \mathbb{1}_{x \geq \epsilon/2} f_{X,Y}(x,y) \, dxdy + \int_{\mathbb{R}^2} \mathbb{1}_{x+y \geq \epsilon} \mathbb{1}_{y < \epsilon/2} f_{X,Y}(x,y) \, dxdy \leq \int_{-\infty}^{\infty} \mathbb{1}_{x \geq \epsilon/2} f_X(x) \, dx + \int_{-\infty}^{\infty} \mathbb{1}_{y \geq \epsilon/2} f_Y(y) \, dy = \Pr (X \geq \epsilon/2) + \Pr (Y \geq \epsilon/2), \quad (A4) \]

where \( f_{X,Y}(x,y) = f_X(x) f_Y(y) \) denotes the joint distribution associated with \( X, Y \). Using inequality (A4), we obtain

\[ \Pr \left( \left| \frac{1}{N} \sum_{i=1}^{N} \eta_i \delta_i \right| \geq \epsilon \right) \leq 4e^{-\frac{N^2 \epsilon}{4}}. \quad (A5) \]
Appendix B. Annihilation method

Let $A \in \mathbb{R}^{N \times N}$ be a symmetric matrix with eigenvalues $\{\lambda_i\}_{i \in \{1, \ldots, N\}}$ satisfying

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_N| \quad \text{and} \quad |\lambda_k - \lambda_1| > |\lambda_j - \lambda_1| \quad \text{for } j \neq k. \quad (B1)$$

Here, $\lambda_k$ is the largest-magnitude eigenvalue that belongs to the set of eigenvalues of the opposite sign of $\lambda_1$. An example of a matrix with eigenvalues that satisfy equation (B1) is a Hessian $H_\theta$ at a saddle point with distinct largest-magnitude eigenvalues (i.e. principal curvatures). Because the matrix $A$ is symmetric, it has an orthonormal eigenbasis, which we denote by $\{v_j\}_{j \in \{1, \ldots, N\}}$. Furthermore, let $B = A - \lambda_1 I_{N \times N}$. For a vector $z_0 \in \mathbb{R}^N$, we calculate

$$z_1 \equiv Bz_0 = Az_0 - \lambda_1 z_0 = \sum_{j=1}^N c_j A v_j - \lambda_1 \sum_{j=1}^N c_j v_j = \sum_{j=2}^N c_j (\lambda_j - \lambda_1) v_j, \quad (B2)$$

where $c_j = \langle v_j, z_0 \rangle$. The corresponding Rayleigh quotient associated with $z_n = Bz_{n-1}$ ($n \in \{1, 2, \ldots\}$) is

$$\lambda^{(n)} \equiv \frac{z_n^\top B z_n}{z_n^\top z_n} = \frac{\sum_{j=2}^N c_j^2 (\lambda_j - \lambda_1)^{2n+1}}{\sum_{j=2}^N c_j^2 (\lambda_j - \lambda_1)^{2n}}, \quad (B3)$$

where we used that $\langle v_i, v_j \rangle = 0$ if $i \neq j$ and 1 otherwise. Notice that $\lambda^{(n)} \rightarrow \lambda_k - \lambda_1$ in the limit $n \rightarrow \infty$. The described method is reminiscent of eigenvalue annihilation, as commonly used in deflation techniques [47]. The main difference in the outlined deflation method w.r.t. standard annihilation approaches is that we are not interested in the second largest-magnitude eigenvalue but in the largest-magnitude eigenvalue that belongs to the set of eigenvalues of the opposite sign of $\lambda_1$. The eigenvalues $\lambda_1, \lambda_k$ can be identified with the largest-magnitude negative and positive principal curvatures, and the corresponding eigenvectors are the dominant Hessian directions.

Appendix C. Image classification

To complement our results on loss landscape visualizations of common image classifiers (see section 4), we considered three additional architectures: the ResNet-56 architecture from the main text but without skip connections [13], MobileNetV2 [50], and GoogLeNet [50]. The numbers of parameters of these three networks are 853,018, 2,236,682, and 5,490,986, respectively. As in the main text, all architectures were trained using SGD with Nesterov momentum. Further implementation details are available in [13, 50].

A complete summary of the results for all the five image classifiers is presented in table 2. For ResNet-56 without skip connections, MobileNetV2, and GoogLeNet, we show heatmaps of the corresponding cross-entropy loss along the random and dominant Hessian directions in figure 8. Overall, table 2 shows that the random projections and Hessian directions are comparable in performance, as measured by the minimum loss and maximum accuracy. However, the heatmaps in figure 8 revealed differences between
Visualizing high-dimensional loss landscapes with Hessian directions

Figure 8. Heatmaps of cross-entropy loss functions along random and dominant Hessian directions. (a)–(d) Loss heatmaps for ResNet-56 without skip connections ((a), (c) training data; (b), (d) test data). (e)–(h) Loss heatmaps for MobileNetV2 ((e), (g) training data; (f), (h) test data). (i)–(l) Loss heatmaps for GoogLeNet ((i), (k) training data; (j), (l) test data). Random directions are used in panels (a), (b), (e), (f), (i), and (j), whereas Hessian directions are used in panels (c), (d), (g), (h), (k), and (l). The green and red regions indicate small and large loss values, respectively. The black crosses indicate the positions of loss minima in the shown domain.

https://doi.org/10.1088/1742-5468/ad13fc
Table 2. Summary of image classification results. We summarize the four performance measures training loss, test loss, training accuracy, and test accuracy of ResNet-56, DenseNet-121, ResNet-56 without skip connections, MobileNetV2, and GoogLeNet. For each neural network, we report all performance measures after initial training in the first row of each cell. We abbreviate ‘accuracy’ as ‘acc’. The second and third rows in each cell list the performance measures that are associated with minimum training loss in random projections (‘rand proj’) and Hessian projections (‘Hess’), respectively. The column ‘dHd’ lists the principal curvatures associated with the dominant Hessian directions at the initial training optimum. Bold numbers indicate the best value of each performance measure associated with a given neural-network architecture.

|                | Training loss | Test loss | Training acc | Test acc | dHd               |
|----------------|---------------|-----------|--------------|----------|-------------------|
| RN-56          | $9.20 \times 10^{-4}$ | $2.88 \times 10^{-1}$ | 99.998       | 93.66    | $\{-16.4,5007.9\}$ |
| rand proj      | $9.10 \times 10^{-4}$ | $2.88 \times 10^{-1}$ | 99.998       | 93.68    | —                 |
| Hess           | $8.75 \times 10^{-4}$ | $2.89 \times 10^{-1}$ | 99.998       | 93.66    | —                 |
| DN-121         | $8.07 \times 10^{-4}$ | $1.69 \times 10^{-1}$ | 100          | 95.63    | $\{-109.1,1937.5\}$ |
| rand proj      | $8.03 \times 10^{-4}$ | $1.69 \times 10^{-1}$ | 100          | 95.64    | —                 |
| Hess           | $6.14 \times 10^{-5}$ | $2.46 \times 10^{-1}$ | 100          | 95.32    | —                 |
| RN-56 (w/o skips) | $1.37 \times 10^{-1}$ | $4.33 \times 10^{-1}$ | 95.546       | 86.69    | $\{-39308.9,50064044.0\}$ |
| rand proj      | $1.36 \times 10^{-1}$ | $4.32 \times 10^{-1}$ | 95.578       | 86.73    | —                 |
| Hess           | $1.35 \times 10^{-1}$ | $4.33 \times 10^{-1}$ | 95.584       | 86.74    | —                 |
| Mobile NetV2   | $4.93 \times 10^{-2}$ | $2.27 \times 10^{-1}$ | 99.518       | 93.91    | $\{-405.0,1061115.0\}$ |
| rand proj      | $4.92 \times 10^{-2}$ | $2.26 \times 10^{-1}$ | 99.524       | 93.90    | —                 |
| Hess           | $4.90 \times 10^{-2}$ | $2.26 \times 10^{-1}$ | 99.516       | 93.93    | —                 |
| GoogLeNet      | $3.95 \times 10^{-2}$ | $2.45 \times 10^{-1}$ | 99.878       | 92.82    | $\{-8208.8,1427483.0\}$ |
| rand proj      | $3.91 \times 10^{-2}$ | $2.45 \times 10^{-1}$ | 99.878       | 92.89    | —                 |
| Hess           | $3.91 \times 10^{-2}$ | $2.44 \times 10^{-1}$ | 99.884       | 92.76    | —                 |

The lack of skip connections in the studied ResNet-56 architecture is associated with larger loss values (training loss: $1.37 \times 10^{-1}$; test loss: $4.33 \times 10^{-1}$) and smaller accuracies (training accuracy: 95.55; test accuracy: 86.69) compared to the ResNet-56 loss values reported in the main text. Because of the large absolute values between $10^4$ and $10^8$ of the principal curvatures associated with the dominant Hessian directions, we study loss projections in a small domain $[-10^{-3},10^{-3}] \times [-10^{-3},10^{-3}]$ around the determined optimum. Figures 8(a) and (b) show that the minimum training and test loss points in the random projection heatmaps are not aligned. The minimum training loss is $1.36 \times 10^{-1}$ with an associated accuracy of 95.58. The corresponding test loss and accuracy are $4.32 \times 10^{-1}$ and 86.73, respectively. In the Hessian direction heatmaps shown in
figures 8(c) and (d), the minimum training loss is $1.35 \times 10^{-1}$, which is slightly smaller than the minimum loss in the random projection plot. The corresponding accuracies are slightly higher than the corresponding random projection values (training accuracy: 95.584 vs. 95.578; test accuracy: 86.74 vs. 86.73).

For MobileNetV2, as shown in figures 8(e) and (f), the minimum training and test loss values in the random projection heatmaps are not aligned. The minimum training loss is $4.92 \times 10^{-2}$, and the corresponding test loss is $2.26 \times 10^{-1}$. The associated training and test accuracies are 99.52 and 93.90, respectively. The minimum training and test loss points in the Hessian projections in figures 8(g) and (h) are more closely aligned than those in the random projections. The minimum training loss in the Hessian projection is $4.90 \times 10^{-2}$, again slightly smaller than that in the random projection. The training accuracy is also slightly smaller (99.516 vs. 99.524), whereas the corresponding test accuracy is slightly larger (93.93 vs. 93.90) than the random projection values.

In contrast to the other random projections in figure 8, the GoogLeNet random projection in figures 8(i) and (j) has better aligned training and test loss minima. The minimum training loss is $3.91 \times 10^{-2}$, and the corresponding test loss is $2.45 \times 10^{-1}$. The associated training and test accuracies are 99.88 and 92.89, respectively. The minimum training loss in the Hessian projection is also $3.91 \times 10^{-2}$, but the training accuracy is slightly larger than that in the corresponding random projection (99.884 vs. 99.878). The corresponding test accuracy is slightly smaller (92.76 vs. 92.89) than that in the random projection.

Appendix D. Function approximation

We compare loss function visualizations that are based on random and Hessian directions in a function approximation task. In accordance with [20], we considered the smooth one-dimensional function

$$f(x) = \log(\sin(10x) + 2) + \sin(x), \quad (D1)$$

where $x \in [-1,1)$. To approximate $f(x)$, we used two FCNNs with two and ten layers, respectively. Each layer has 100 ReLU activations. The numbers of parameters of the two- and ten-layer architectures are 20 501 and 101 301, respectively. The training data are based on 50 points that were sampled uniformly at random from the interval $[-1,1)$. We trained both neural networks using a mean squared error (MSE) loss function and SGD with a learning rate of 0.1. The two- and ten-layer architectures were trained for 100 000 and 50 000 epochs, respectively, to reach loss values of less than $10^{-4}$. The best model was saved and evaluated by calculating the MSE loss for 1000 points that were sampled uniformly at random from the interval $[-1,1)$. Figure 9 shows heatmaps of the training and test loss landscapes of both neural networks along random and dominant Hessian directions. The black crosses in figure 9 indicate loss minima. As in the main text, we observe for both neural networks that random projections are associated with loss values that increase along both directions $\delta, \eta$ and for both the training and test data (figures 9(a), (b), (e), and (f)). For these
projections, we found that the loss minima are very close to or at the origin of the loss space. The situation is different for the loss projections that are based on Hessian directions. Figure 6(c) shows that the loss minimum for the two-layer FCNN is not located at the origin but at $\alpha, \beta \approx (0, 0.04)$. The value of the training loss at that point is more than 9% smaller than the smallest training loss found in a random projection plot. The corresponding test loss is approximately 1% smaller than the test loss associated with the smallest training loss in the random projection plot. For the ten-layer FCNN, the smallest training loss in the Hessian direction projection is more than 26% smaller than the smallest training loss in the randomly projected loss landscape (figures 9(e) and (g)). The corresponding test loss in the Hessian direction plot is approximately 6% smaller than the corresponding test loss minimum in the random direction plot (figures 9(f) and (h)). Notice that both the training and test loss minima in the random direction heatmaps in figures 9(e) and (f) are located at the origin, whereas they are located at $\alpha, \beta \approx (0, -0.05)$ in the Hessian direction heatmaps in figures 9(g) and (h).

In the supplemental information [51], we include an animation that shows the evolution of lower-dimensional loss projections of the ten-layer FCNN.
Figure 9. Heatmaps of the mean squared error (MSE) loss along the random and dominant Hessian directions for a function approximation task. (a)–(d) Loss heatmaps for an FCNN with two layers and 100 ReLU activations per layer ((a), (c) training data; (b), (d) test data). (e)–(h) Loss heatmaps for an FCNN with ten layers and 100 ReLU activations per layer ((e), (g) training data; (f), (h) test data). Random directions are used in panels (a), (b), (e), and (f), whereas Hessian directions are used in panels (c), (d), (g), and (h). The green and red regions indicate small and large MSE loss values, respectively. The black crosses indicate the positions of loss minima in the shown domain. Both neural networks, FCNN 1 and FCNN 2, are trained to approximate the smooth one-dimensional function (D1).
References

[1] Cooper Y 2021 Global minima of overparameterized neural networks SIAM J. Math. Data Sci. 3 676
[2] Hornik K, Stinchcombe M and White H 1989 Multilayer feedforward networks are universal approximators Neural Netw. 2 359
[3] Hornik K 1991 Approximation capabilities of multilayer feedforward networks Neural Netw. 4 251
[4] Park S, Yun C, Lee J and Shin J 2021 Minimum width for universal approximation 9th Int. Conf. on Learning Representations, ICLR 2021 (Virtual, 3–7 May 2021) (OpenReview.net)
[5] Hardt M, Recht B and Singer Y 2016 Train faster, generalize better: stability of stochastic gradient descent Proc. 33rd Int. Conf. on Machine Learning (New York, USA, 20–22 June 2016) ed M F Balcan and K Q Weinberger (PMLR) pp 1225–34 (https://proceedings.mlr.press/v48/hardt16.html)
[6] Wilson A C, Roelofs R, Stern M, Srebro N and Recht B 2017 The marginal value of adaptive gradient methods in machine learning Advances in Neural Information Processing Systems (NIPS 2017) (Long Beach, CA, USA, 4–9 December 2017) ed I Guyon et al vol 30 (Curran Associates, Inc.) (https://proceedings.neurips.cc/paper_files/paper/2017/file/81b3833e2504647f9d794f7d7d719b341-Paper.pdf)
[7] Choi D, Shalhoub C J, Nado Z, Lee J, Maddison C J and Dahl G E 2019 On empirical comparisons of optimizers for deep learning (arXiv:1910.05446)
[8] Baldassi C, Pittorino F and Zecchina R 2020 Shaping the learning landscape in neural networks around wide flat minima Proc. Natl Acad. Sci. 117 161
[9] Pittorino F, Lucibello C, Feinauer C, Perugini G, Baldassi C, Denyneneko E and Zecchina R 2021 Entropic gradient descent algorithms and wide flat minima J. Stat. Mech. 124015
[10] Keskar N S, Mudigere D, Nocedal J, Smelyanskiy M and Tang P T P 2017 On large-batch training for deep learning: generalization gap and sharp minima 5th Int. Conf. on Learning Representations, ICLR 2017, Conf. Track Proc. (Toulon, France, 21–26 April 2017)
[11] Dinh L, Pascanu R, Bengio S and Bengio Y 2017 Sharp minima can generalize for deep nets Proc. 34th Int. Conf. on Machine Learning, ICML 2017 (Sydney, NSW, Australia, 6–11 August 2017) ed D Precup and Y W Teh (PMLR) vol 70 pp 1019–28 (https://proceedings.mlr.press/v70/dinh17b.html)
[12] Goodfellow I J and Vinyals O 2015 Qualitatively characterizing neural network optimization problems 3rd Int. Conf. on Learning Representations, ICLR 2015, Conf. Track Proc. (San Diego, CA, USA, May 2015) ed Y Bengio and Y LeCun
[13] Li H, Xu Z, Taylor G, Studer C and Goldstein T 2018 Visualizing the Loss Landscape of Neural Nets Advances in Neural Information Processing Systems: Annual Conf. on Neural Information Processing Systems 2018, NeurIPS 2018 (Montreal, Canada, 3–8 December 2018) vol 31, ed S Bengio, H M Wallach, H Larochelle, K Grauman, N Cesa-Bianchi and R Garnett pp 6391–401
[14] Hao Y, Dong L, Wei F and Xu K 2019 Visualizing and Understanding the Effectiveness of BERT Proc. 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing, EMNLP-IJCNLP 2019 (Hong Kong, China, 3–7 November 2019) ed K Inui, J Jiang, V Ng and X Wan (Association for Computational Linguistics) pp 4141–50
[15] Wu D, Xia S-T and Wang Y 2020 Adversarial weight perturbation helps robust generalization Advances in Neural Information Processing Systems 33 (NeurIPS 2020) (Virtual, 6–12 December 2020) ed H Larochelle et al (Curran Associates, Inc.) p 2958 (https://proceedings.neurips.cc/paper/2020/hash/1ef91e212c30614bf125e93742624011-Abstract.html)
[16] Horoi S, Huang J, Rieck B, Lajoie G, Wolf G and Krishnaswamy S 2022 Exploring the Geometry and Topology of Neural Network Loss Landscapes Advances in Intelligent Data Analysis XX—20th Int. Symp. on Intelligent Data Analysis, IDA 2022 (Renness, France, 20–22 April 2022) (Lecture Notes in Computer Science vol 13205) ed T Bonardi, E Fromont and E Hüllermeier (Springer) pp 171–84
[17] Baldi P and Hornik K 1989 Neural networks and principal component analysis: learning from examples without local minima Neural Netw. 2 53
[18] Dauphin Y N, Pascanu R, Gulıçehre Ç, Cho K, Ganguli S and Bengio Y 2014 Identifying and attacking the saddle point problem in high-dimensional non-convex optimization Advances in Neural Information Processing Systems: Annual Conf. on Neural Information Processing Systems 2014 (Montreal, Quebec, Canada, 8–13 December 2014) vol 27, ed Z Ghahramani, M Welling, C Cortes, N D Lawrence and K Q Weinberger pp 2933–41
[19] Hutchinson M F 1989 A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines Commun. Stat. Simul. Comput. 18 1059
[20] Adcock B and Dexter N 2021 The gap between theory and practice in function approximation with deep neural networks SIAM J. Math. Data Sci. 3 624

https://doi.org/10.1088/1742-5468/ad13fc
Visualizing high-dimensional loss landscapes with Hessian directions

[21] Yao Z, Gholami A, Keutzer K and Mahoney M W 2020 PyHessian: neural networks through the lens of the Hessian 2020 IEEE Int. Conf. on Big Data (Big Data) (IEEE) pp 581–90
[22] Liao Z and Mahoney M W 2021 Hessian eigenspectra of more realistic nonlinear models Advances in Neural Information Processing Systems 34 (NeurIPS 2021) (Virtual, 6–14 December 2021) ed M Ranzato et al (Curran Associates, Inc) pp 20104–17 (https://proceedings.neurips.cc/paper/2021/hash/a7d8ae4569120b5b5e12c7b6e0648686-Abstract.html)
[23] Böttcher L 2022 GitLab repository (available at: https://gitlab.com/ComputationalScience/loss-visualization)
[24] Lee J M 2006 Riemannian Manifolds: An Introduction to Curvature vol 176 (Springer Science & Business Media)
[25] Berger M and Gostiaux B 2012 Differential Geometry: Manifolds, Curves and Surfaces vol 115 (Springer Science & Business Media)

Kühnel W 2015 Differential Geometry vol 77 (American Mathematical Society)

Crane K 2018 Discrete differential geometry: an applied introduction Not. AMS Commun. 1153

Martens J 2020 New insights and perspectives on the natural gradient method J. Mach. Learn. Res. 21 146

Casella G and Berger R L 2002 Statistical Inference 2nd edn (Duxbury Press)

Lehmann E L and Casella G 2006 Theory of Point Estimation 2nd edn (Springer)

Rao S S 2017 Advanced Statistical Inference (Lecture Notes) (Texas A & M University)

Amari S-I 1998 Natural gradient works efficiently in learning Neural Comput. 10 251

Karakida R, Akaho S and Amari S-I 2020 universal statistics of Fisher information in deep neural networks: mean field approach J. Stat. Mech. 124005

Amari S-I, Park H and Fukumizu K 2000 Adaptive method of realizing natural gradient learning for multilayer perceptrons Neural Comput. 12 1399

Park H, Amari S-I and Fukumizu K 2000 Adaptive natural gradient learning algorithms for various stochastic models Neural Netw. 13 755

Hecht-Nielsen R 1994 Context vectors: general purpose approximate meaning representations self-organized from raw data Computational Intelligence: Imitating Life vol 43 (IEEE Press)

Laurent B and Massart P 2000 Adaptive estimation of a quadratic functional by model selection Ann. Stat. 28 1302

Schraudolph N N 2002 Fast curvature matrix-vector products for second-order gradient descent Neural Comput. 14 1723

Diehl M and Messerer F 2019 Local convergence of generalized Gauss–Newton and sequential convex programming 58th IEEE Conf. on Decision and Control, CDC 2019 (Nice, France, 11–13 December 2019) (IEEE) pp 3942–7

Davies R B 1980 Algorithm AS 155: the distribution of a linear combination of chi-square random variables Appl. Stat. 29 323

Bausch J 2013 On the efficient calculation of a linear combination of chi-square random variables with an application in counting string vacua J. Phys. A: Math. Theor. 46 505202

Chen T and Lumley T 2019 Numerical evaluation of methods approximating the distribution of a large quadratic form in normal variables Stat. Data Anal. 139 75

Matoušek J 2008 On variants of the Johnson–Lindenstrauss lemma Random Struct. Algorithms 33 142

Yao Z, Gholami A, Keutzer K and Mahoney M K 2018 Hessian-based analysis of large batch training and robustness to adversaries Advances in Neural Information Processing Systems (NIPS 2018) vol 31, ed S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi and R Garnett (Curran Associates, Inc.) pp 4954–64

Bai Z, Fahey G and Golub G 1996 Some large-scale matrix computation problems J. Comput. Appl. Math. 74 71

Avron H and Toledo S 2011 Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix J. ACM 58 1

Burdyn R L, Faires J D and Burden A M 2015 Numerical Analysis 10th edn (Cengage Learning)

Lelouch R B, Sorensen D C and Yang C 1998 Arpack Users’ Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods (SIAM)

Pearlmutter B A 1994 Fast exact multiplication by the Hessian Neural Comput. 6 147

Phan H 2022 Pretrained TorchVision models on CIFAR-10 dataset (available at: https://github.com/huyvphan/PyTorch_CIFAR10)

Böttcher L 2023 Video of loss evolution (available at: https://vimeo.com/787174225)

Li K-C 1992 On principal Hessian directions for data visualization and dimension reduction: another application of Stein’s lemma J. Am. Stat. Assoc. 87 1025

Zagoruyko S and Komodakis N 2016 Wide residual networks Proc. British Machine Vision Conf. 2016, BMVC 2016 (York, UK, 19–22 September 2016) ed R C Wilson, E R Hancock and W A P Smith (BMVA Press)
