Flatness is a False Friend

Anonymous authors
Paper under double-blind review

Abstract

Hessian based measures of flatness, such as the trace, Frobenius and spectral norms, have been argued, used and shown to relate to generalisation. In this paper we demonstrate that, for feed-forward neural networks under the cross-entropy loss, low-loss solutions with large neural network weights have small Hessian based measures of flatness. This implies that solutions obtained without L2 regularisation should be less sharp than those with despite generalising worse. We show this to be true for logistic regression, multi-layer perceptrons, simple convolutional, pre-activated and wide residual networks on the MNIST and CIFAR-100 datasets. Furthermore, we show that adaptive optimisation algorithms using iterate averaging, on the VGG-16 network and CIFAR-100 dataset, achieve superior generalisation to SGD but are 30× sharper. These theoretical and experimental results further advocate the need to use flatness in conjunction with the weights scale to measure generalisation (Neyshabur et al., 2017; Dziugaite and Roy, 2017).

1 Introduction

Deep Neural Networks (DNNs), with more parameters than data-points, trained with many passes of the same data, still manage to perform exceptionally on test data. The reasons for this remain laregly unsolved (Neyshabur et al., 2017). However, DNNs are not completely immune to the classical problem of over-fitting. Zhang et al. (2016) show that DNNs can perfectly fit random labels. Schedules with initially low or sharply decaying learning rates, lead to identical training but much higher testing error (Berrada et al., 2018; Granziol et al., 2020a; Jastrzebski et al., 2020). In Wilson et al. (2017) the authors argue that specific adaptive gradient optimisers lead to solutions which don't generalise. This has lead to a significant development in partially adaptive algorithms (Chen and Gu, 2018; Keskar and Socher, 2017). Given the importance of accurate predictions on unseen data, understanding exactly what helps deep networks generalise has been a fundamental area of research. A key concept which has taken a foothold in the community, allowing for the comparison of different training loss minima using only the training data, is the concept of *flatness*. From both a Bayesian and minimum description length framework, flatter minima should generalize better than sharp minima (Hochreiter and Schmidhuber, 1997).

Sharpness is usually measured by properties of the second derivative of the loss,, the Hessian $\boldsymbol{H} = \nabla^2 L(\boldsymbol{w})$ (Keskar et al., 2016; Jastrzebski et al., 2017b; Chaudhari et al., 2016; Wu et al., 2017; 2018), such as the spectral norm or trace. The assumption is that due to finite numerical precision (Hochreiter and Schmidhuber, 1997) or from a Bayesian perspective (MacKay, 2003), the test surface is *shifted* from the training surface. The difference between train and test loss for a shift $\Delta \boldsymbol{w}$ is given by

$$L(\boldsymbol{w}^* + \Delta \boldsymbol{w}) - L(\boldsymbol{w}^*) \approx \Delta \boldsymbol{w}^T \boldsymbol{H} \Delta \boldsymbol{w} + ... \approx \sum_i^P \lambda_i |\boldsymbol{\phi}_i^T \Delta \boldsymbol{w}|^2 \approx \frac{\mathrm{Tr}(\boldsymbol{H})}{P} ||\Delta \boldsymbol{w}||^2 \leq \lambda_1 ||\Delta \boldsymbol{w}||^2 \tag{1}$$

in which $\boldsymbol{w}^*$ is the final training point and $[\lambda_i, \boldsymbol{\phi}_i]$ are the eigenvalue/eigenvector pairs of $\boldsymbol{H} \in \mathbb{R}^{P \times P}$. We have dropped the terms beyond second-order by assuming that the gradient at training end is small. In general we have no a priori reason to assume that shift should preferentially lie along any of the Hessian eigenvectors, hence by taking a maximum entropy prior (MacKay, 2003; Jaynes, 1982) we expect strong high dimensional concentration results

(Vershynin, 2018) to hold, hence $|\boldsymbol{\phi}_i^T \Delta\hat{\boldsymbol{w}}|^2 \approx 1/P$, where $\hat{\boldsymbol{w}}$ is simply the normalised version of $\boldsymbol{w}$. This justifies the trace as a measure of sharpness. In the worst case scenario the shift is completely aligned with the eigenvector corresponding to the largest eigenvalue λ_1, i.e. $\Delta\boldsymbol{w}^T\boldsymbol{\phi}_1 = 1$. Hence the spectral norm λ_1 of $\boldsymbol{H}$ serves as a local[1] upper bound to the loss change. The idea of a *shift* between the training and testing loss surface is prolific in the literature and regularly related to generalisation (He et al., 2019; Izmailov et al., 2018; Maddox et al., 2019). Alternative, yet closely related, measures of flatness are also used. Keskar et al. (2016) define a sharp minimiser as one "with a significant number of large positive eigenvalues", in fact as can be seen by the Rayleigh-Ritz theorem, the metric which they propose, shown in Equation 2 is proportional to the largest eigenvalue.

$$\phi_{\boldsymbol{w},L}(\epsilon, \boldsymbol{A}) := \frac{(\max_{\boldsymbol{y}\in\mathcal{C}_\epsilon} L(\boldsymbol{w} + \boldsymbol{A}\boldsymbol{y})) - L(\boldsymbol{w})}{1 + L(\boldsymbol{w})} \leq \kappa(\epsilon)\lambda_1 \tag{2}$$

$\mathcal{C}_\epsilon$ is the constraint box as defined in (Keskar et al., 2016), where ϵ controls the box size. As shown by Dinh et al. (2017), this definition of sharpness is approximately given by $\lambda_1\epsilon^2/2(1 + L(\boldsymbol{w}))$, proportional to the largest eigenvalue. This result can be explained intuitively as within a small vicinity of $\boldsymbol{w}$ the largest change in loss is along the leading eigenvector and is proportional to the largest eigenvalue. Wu et al. (2017) consider the logarithm of the product of the top k eigenvalues as a proxy measure the volume of the minimum (a truncated log determinant). In this paper we will exclusively consider the Hessian trace, spectral and Frobenius norm as measures of sharpness.

Motivation: There have been numerous positive empirical results relating sharpness and generalisation. Keskar et al. (2016); Rangamani et al. (2019) consider how large batch vs small batch stochastic gradient descent (SGD) alters the sharpness of solutions, with smaller batches leading to convergence to flatter solutions, leading to better generalisation. Jastrzebski et al. (2017a) look at the importance of the ratio learning rate and batch size in terms of generalisation, finding that large ratios lead to flatter minima (as measured by the spectral norm) and better generalisation. Yao et al. (2018) investigated flat regions of weight space (small spectral norm) showing them to be more robust under adversarial attack. Zhang et al. (2018) show that SGD concentrates in probability on flat minima. Certain algorithmic design choices, such as Entropy-SGD (Chaudhari et al., 2016) and the use of Polyak averaging (Izmailov et al., 2018) have been motivated by considerations of flatness. However Dinh et al. (2017) show that by exploiting ReLUs (Rectified Linear Units) positive homogeneity property $f(\alpha\boldsymbol{x}) = \alpha f(\boldsymbol{x})$, any flat minima can be mapped into a sharp minimum, without altering the loss. As these measures can be arbitrarily distorted, this implies they serve little value as generalisation measures. However such transformations alter other properties, such as the weight norm. In practice the use of $L2$ regularisation, which penalises weight norm, means that optimisers are unlikely to converge to such a solution. It can even be shown that unregularised SGD converges to the minimum norm solution for simple problems (Wilson et al., 2017), further limiting the practical relevance of such reparameterisation arguments. The question which remains and warrants investigation, is *are Hessian based sharpness metrics at the end of training meaningful metrics for generalisation?* We demonstrate both theoretically and experimentally that the answer to this question is an affirmative **no**.

Contributions: To the best of our knowledge, this is the first work which demonstrates theoretically motivated empirical results contrary to purely flatness based generalisation measures. For the fully connected feed-forward network with ReLU activation and cross entropy loss, we demonstrate in the limit of 0 training loss, that the spectral norm and trace of the Hessian also go to 0. The key insight is that in order for the loss to go to 0, the weight vector components $\boldsymbol{w}_c$ must tend to infinity. Conversely, this implies that methods which reduce the weight magnitudes extensively used to aid generalisation (Bishop, 2006; Krogh and Hertz, 1992), makes solutions sharper. We present the counter-intuitive result that adding $L2$ regularisation increases both sharpness and generalisation, for Logistic Regression, MLP, simple CNN, PreResNet-164 and WideResNet-28 $\times$ 10 for the MNIST and CIFAR-100

[1] we use the word local here because the largest eigenvalue/eigenvector pair may change along the path taken

datasets. We also present and discuss various amendments to the Hessian, which are robust against the arguments presented here and those of Dinh et al. (2017).

Related work: Empirically negative results on flatness and its effect on generalisation have been previously observed. Neyshabur et al. (2017) show that it captures generalisation for large but not small networks. Golatkar et al. (2019) show that the maximum of the trace of the Fisher information correlates better with generalisation than its final value. Jastrzebski et al. (2018) show that it is possible to optimise faster and attain better generalisation performance whilst finding a final sharper region. For small networks, those trained on random labels (with no generalisation) are less sharp than those trained on the true labels. However this does not rule out that for the same network trained on true labels, solutions which are flatter generalise better. Instead the main focus has centered around the Hessians lack of reparameterisation invariance (Neyshabur et al., 2017; Tsuzuku et al., 2019; Rangamani et al., 2019). This has been a primary motivator for normalised definitions of flatness Tsuzuku et al. (2019); Rangamani et al. (2019) often in a PAC-Bayesian framework. In Ballard et al. (2017); Mehta et al. (2018), it was shown that adding the L2 regularization on weights including the bias weights removed singular modes of the Hessian matrix for a feed-forward artificial neural network with one hidden layer, with tanh activation function, employed to fit the XOR data. In Mehta et al. (2018), with the help of an algebraic geometry interpretation of the loss landscape of the deep linear networks, it was proven that a generalized L2 regularization guaranteed to remove all singular solutions leaving the Hessian matrix strictly non-singular at every critical point.

2 Gedanken Experiment: why the Hessian won't do

For a simple illustration let us consider the deep linear model, with exponential loss. The deep linear model is often employed as a theoretical tool for its analytical tractability (Kawaguchi, 2016; Lu and Kawaguchi, 2017). In Section 3 we formalise the results to the fully connected feed forward network with cross entropy loss. Intuitively we can think of a feed forward network as a sum of deep linear networks and the cross entropy as an approximation to the exponential loss. For 3 parameters and a single datum X, the loss is given by $L = \exp(w_1 w_2 w_3 X)$. The Hessian, its trace and spectral norm $\boldsymbol{H}, \mathrm{Tr}(\boldsymbol{H}), \lambda_1(\boldsymbol{H})$ are given by

$$\boldsymbol{H} = \begin{bmatrix} w_2^2 w_3^2 & w_1 w_2 w_3^2 & w_2^2 w_1 w_3 \\ w_1 w_2 w_3^2 & w_1^2 w_3^2 & w_1^2 w_2 w_3 \\ w_2^2 w_1 w_3 & w_1^2 w_2 w_3 & w_1^2 w_2^2 \end{bmatrix} X \exp(w_1 w_2 w_3) X \tag{3}$$

$$\mathrm{Tr}(\boldsymbol{H}) = \lambda_1(\boldsymbol{H}) = (w_2^2 w_3^2 + w_2^2 w_1^2 + w_1^2 w_3^2) X \exp(w_1 w_2 w_3 X) \tag{4}$$

Smaller losses imply flatter Hessians: Equation 4 shows that under this model the trace and maximum eigenvalue are products of a polynomial function of the weights and an exponential in the weights. As the optimiser drives the loss $L \to 0$ we expect the exponential to dominate the polynomial[2]. This implies that methods to reduce the weight magnitude, such as $L2$ regularisation, which has been extensively shown to aid generalisation (Krogh and Hertz, 1992; Bishop, 2006) should increase Hessian measures of sharpness. We show that this is the case experimentally in Section 4.

3 Theoretical Framework

In this Section we extend our intuition developed under the deep linear network with exponential loss, to more realistic scenarios. Similar to the prior work of Choromanska et al. (2015); Milne (2019), we consider a neural network with a d_x dimensional input $\boldsymbol{x}$. Our network has $H-1$ hidden layers and we refer to the output as the H'th layer and the input as the 0'th layer. We denote the ReLU activation function as $f(x)$ where $f(x) = \max(0, x)$.

[2]consider the function $x^n \exp(-x^m) \to 0 \ \forall n, m$ as $x \to \infty$

Let $\boldsymbol{W}_i$ be the matrix of weights between the $(i-1)$'th and i'th layer. For a d_y dimensional output our q'th component of the output can be written as

$$\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_q = f(\boldsymbol{W}_H^T f(\boldsymbol{W}_{H-1}^T ....f(\boldsymbol{W}_1\boldsymbol{x}))) = \sum_{i=1}^{d_x}\sum_{j=1}^{\gamma} \boldsymbol{x}_i A_{i,j} \prod_{k=1}^{H} w_{i,j}^{(k)} \tag{5}$$

where the indices i, j denote the sum over network inputs and paths respectively and γ is the number of paths. $A_{i,j} \in [0,1]$ denotes whether the path is active or not and $w_{i,j}^{(q)}$ denotes the the weight of the path segment which connects node i in layer $q-1$ with node j in layer q. layer i has n_i nodes and $\gamma = \prod_q^{H-1} n_q$ We formalise our intuition from Section 2 with the following theorem:

Theorem 1. *For any feed forward neural network with ReLU output activation functions $f(x) = \max(0, x)$, coupled with a softmax output in the final layer and cross entropy loss, in the limit that the training loss $L(\boldsymbol{w}) \to 0$ the spectral norm λ_1 of the empirical Hessian $\boldsymbol{H} = \nabla\nabla L(\boldsymbol{w}) \in \mathbb{R}^{P\times P}$ also tends to 0.*

Proof. The cross-entropy loss $\ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y})$ of a single sample $\boldsymbol{x}_i$ is defined by:

$$\ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i) = -\sum_{c}^{d_{\boldsymbol{y}}} (1 - \mathbb{1}[h(\boldsymbol{x}_i;\boldsymbol{w})_c \neq y_c]) \log h(\boldsymbol{x}_i;\boldsymbol{w})_c \tag{6}$$

Where d_y is the number of classes and $\mathbb{1}$ is the indicator function which takes the value of 1 for the incorrect class and 0 for the correct class, $\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})$ is the softmax input. The softmax output $\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_c$ for class c is given by

$$h(\boldsymbol{x}_i,\boldsymbol{w})_c = \sigma(\boldsymbol{z})_c = \frac{\exp \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_c}{\sum_{k=1}^{d_y} \exp \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_k} = \left(1 + \sum_{k\neq c}^{d_y} \exp\left(\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_k - \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_c\right)\right)^{-1} \tag{7}$$

Hence combining Equations 6 and 7, the loss per sample can be written as

$$\ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i) = \log\left(1 + \sum_{k\neq c(i)}^{d_y} \exp\left(z_{k,c(i)}\right)\right) \tag{8}$$

in which $c(i)$ denotes the correct class for the data point $\boldsymbol{x}_i$ and $z_{k,c(i)} = \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_k - \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_{c(i)}$. Note that, for the per sample loss $\ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i) \to 0$, $\exp(z_{k,c(i)}) \to 0$ $\forall k \neq c(i)$. Using the chain rule

$$\begin{aligned}\frac{\partial^2 \ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i)}{\partial w_l \partial w_m} = &\frac{\sum_{k\neq c(i)} \exp(z_{k,c(i)})[\frac{\partial^2 z_{k,c(i)}}{\partial w_l \partial w_m} + \frac{\partial z_{k,c(i)}}{\partial w_l}\frac{\partial z_{k,c(i)}}{\partial w_m}]}{1 + \sum_{i\neq c} \exp(z_{k,c(i)})} \\ &- \frac{\sum_{k\neq c(i), u\neq c(i)} \exp(z_{k,c}) \frac{\partial z_{k,c(i)}}{\partial w_l} \exp(z_{u,c(i)}) \frac{\partial z_{u,c(i)}}{\partial w_m}}{(1 + \sum_{k\neq c(i)} \exp(z_{k,c(i)}))^2}\end{aligned} \tag{9}$$

As shown in Milne (2019), the network is differentiable at the majority of points in weight space. Specifically the zero set of non-zero real analytic functions (the network is piecewise analytic in the weights) has Lebesgue measure zero. Hence all we need to show is that the output derivatives tend to ∞ more slowly than the loss tends to 0. To do this we consider

$$\frac{\partial^m}{\prod_m \partial w_{\mu_m,\phi_m}^m} \sum_{i=1}^{d_x}\sum_{j=1}^{\gamma} \boldsymbol{x}_i A_{i,j} \prod_{k=1}^{H} w_{i,j}^{(k)} = \sum_{i=1}^{d_x} \boldsymbol{x}_i A_{i,j} \sum_{k\neq m}^{\gamma/\prod_m n_m} w_{i,j}^{(k)} (1 - \mathbb{1}[w_{\mu_m,\phi_m}^m]) \tag{10}$$

We are interested in the limit where the output for the correct class $\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_{c(i)} \to \infty$. Although any individual weight segment $w_{i,j}^k$ may be zero, this simply deactivates that path segment and reduces the contributing sum. Hence we can absorb such zero weights into the $A_{i,j}$ and without loss of generality assume that the weights are bounded to a minimum

absolute value of $|w_{i,j}^{(k)}| \geq \epsilon > 0$. Note from Equation 10 that only paths containing the weight segment corresponding to the differentiated nodes contribute, hence the bracket on the RHS of this equation is also upper bounded by 1. We can upper bound the norm of the differentiated output

$$\left| \frac{\partial^m}{\prod_m \partial w_{\mu_m,\phi_m}^m} \sum_{i=1}^{d_x} \sum_{j=1}^{\gamma} \boldsymbol{x}_i A_{i,j} \prod_{k=1}^{H} w_{i,j}^{(k)} \right| \leq \frac{|\sum_{i=1}^{d_x} \sum_{j=1}^{\gamma} \boldsymbol{x}_i A_{i,j} \prod_{k=1}^{H} w_{i,j}^{(k)}|}{\epsilon^m \prod_m n_m} \tag{11}$$

Hence

$$\frac{\partial^m \boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_q}{\prod_m \partial w_{\mu_m,\phi_m}^m} \exp\left(\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_q\right) \leq \frac{|\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_q|}{\epsilon^m \prod_m n_m} \exp\left(\boldsymbol{z}(\boldsymbol{x}_i;\boldsymbol{w})_q\right) \tag{12}$$

and hence in the limit $\ell(h(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i) \to 0$, all terms in Equation 9 have norms tending to zero. Hence $\frac{\partial^2 \ell(h(\boldsymbol{x}_i;\boldsymbol{w}),\boldsymbol{y}_i)}{\partial w_l \partial w_m} \to 0$. Taking $l = m$ and summing over m we have 0 trace and using the Frobenius norm identity, i.e. taking the sum of squares over l, m we have $\sum_i^P \lambda_i^2 \to 0$ and hence $\lambda_1 \to 0$.

Remark. By writing the loss in terms of the activation σ at the output of the final layer $f(\boldsymbol{w})$, i.e $L(\boldsymbol{w}) = \sigma(f(\boldsymbol{w}))$. The Hessian may be expressed using the chain rule as

$$\boldsymbol{H}(\boldsymbol{w})_{jk} = \frac{1}{N} \sum_{n=1}^{N} \left(\sum_{c=0}^{d_y} \sum_{l=0}^{d_y} \frac{\partial^2 \sigma(f(\boldsymbol{w}))}{\partial f_l(\boldsymbol{w}) \partial f_c(\boldsymbol{w})} \frac{\partial f_l(\boldsymbol{w})}{\partial w_j} \frac{\partial f_c(\boldsymbol{w})}{\partial w_c} + \sum_{c=0}^{d_y} \frac{\partial \sigma(f(\boldsymbol{w}))}{\partial w_j} \frac{\partial^2 f_c(\boldsymbol{w})}{\partial w_j \partial w_k} \right) \tag{13}$$

Where, for the cross-entropy loss and softmax output at exactly 0 loss, $\frac{\partial f_l(\boldsymbol{w})}{\partial w_j} = 0$ and $\frac{\partial^2 \sigma(f(\boldsymbol{w}))}{\partial f_l(\boldsymbol{w}) \partial f_c(\boldsymbol{w})} = 0$. However, in practice, since the weights are finite, we never have 0 loss. Hence, unlike our proof which shows that the Hessian is given by a product of a polynomial and exponential in the weights which we expect to go to 0 in the limit of large weights and low loss, this simple result does yield information prior to the loss being exactly 0. □

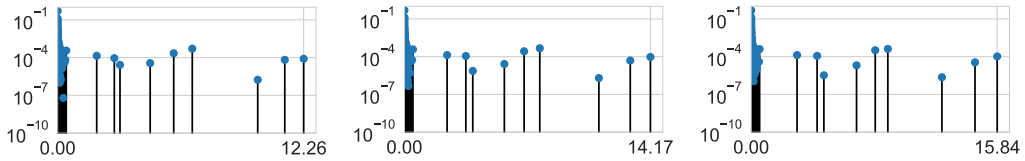
4 Weight Decay and Sharpness

Section 3 more formally demonstrates what was already hypothesised in Section 2. Larger weights, required to drive the loss to very low values, are expected to give small Hessian based measures of sharpness, despite potentially wildly over-fitting the data and generalising poorly. To show that this is relevant in practice we evaluate the effect of weight norm reducing techniques, such as $L2$ regularisation on spectral sharpness. $L2$ is regularly used to help generalisation, having been showed to reduce the effect of static noise on the target (Krogh and Hertz, 1992).

Experimental Setup: We use the deep visualisation suite (Granziol et al., 2019) package to visualise the spectrum of the Hessian and calculate the largest eigenvalues. We train all networks using SGD with momentum $\rho = 0.9$ and varying levels of L2 regularisation $\frac{\gamma}{2}||\boldsymbol{w}||^2$, $\gamma \in [0, 0.0001, 0.0005]$. For further experimental details, such as the learning rate schedule (we a linear decay schedule with a terminal learning rate of 0.01 the initial) employed and the finer details of the spectral visualisation method see Appendix A. Since adding $L2$ regularisation naturally adds γ to each eigenvalue, as $\boldsymbol{H} \to \boldsymbol{H} + \gamma I$, in our results we *do not* calculate the Hessian on the regularised loss. For simplicity we focus on the spectral norm, but include results on the Frobenius norm and trace in the Appendix. Furthermore since for certain simple architectures the inclusion of $L2$ regularisation also increases optimisation performance, wherever this is the case we will relate Hessian based measures of sharpness to the *generalisation gap*. For which we just report the difference in validation and training accuracy. For more modern networks using batch normalisation (Ioffe and Szegedy, 2015), $L2$ regularisation reduces convergence speed but negligibly alters optimisation performance, so we can relate measures of sharpness to the validation performance directly[3].

[3]since we are substracting essentially the same amount

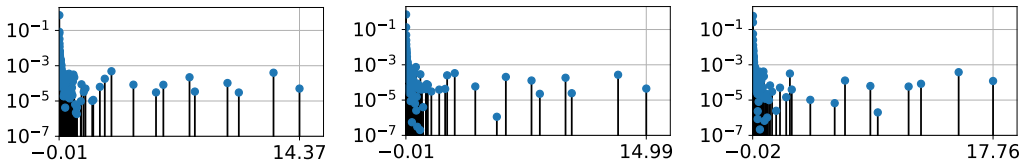
Logistic Regression on MNIST: The simplest Neural Network model, corresponding to a 0 hidden layer feed forward neural network, is the Softmax Regressor [4]. By the diagonal dominance theorem (Cover and Thomas, 2012) the Hessian of the Softmax Regressor is positive semi-definite, so the loss surface is convex. For a convex objective all local minima are global, hence we do not have the complication of different minima. We run this model on the MNIST dataset (LeCun, 1998), splitting the training set into $45,000$ training and $5,000$ validation samples. The total parameter count is 7850. We run for 1000 (we specifically use an abnormally large number of epochs to make sure that convergence is not an issue) epochs with learning rate 0.01. The validation accuracy increases incrementally with increased weight decay $[93.48, 94, 94.08]$. We plot the spectra of the final solution in Figure 1. We note here that for increasing weight decay coefficient, which corresponds to higher performing testing solutions, the spectral norm increases, from $\lambda_1 = [12.26, 14.17, 15.84]$. This shows that greater Hessian based measures of sharpness, occur for solutions with improved generalisation.



(a) $Acc = 93.48$ $\lambda = 0$ (b) $Acc = 94$ $\lambda = 10^{-4}$ (c) $Acc = 94.08$ $\lambda = 5 \times 10^{-4}$

Figure 1: Hessian spectrum for Softmax regression after 1000 epochs of SGD on the MNIST dataset, for various $L2$ regularisation coefficients λ

MLP: We now consider a single hidden layer MLP on the MNIST dataset, with a hidden layer of 100 units, parameter count 9960, trained for 50 epochs with an identical schedule and a learning rate of 0.01. We similarly find that the addition of weight decay both increases the generalisation accuracy (from $94.4 \rightarrow 96.46 \rightarrow 96.7$ as we increase the regularisation coefficient γ from $0 \rightarrow 0.0001 \rightarrow 0.0005$). Similar to the Softmax example, this also increases the spectral norm as shown in Figure 2. The training accuracy increases slightly with the introduction of regularisation, but decreases over the unregularised network when the regularisation is increased to 0.0005. The generalisation gap for the various levels of regularisation is $[0.48\%, 0.51\%, 0.04\%]$, where the smallest generalisation gap corresponds to the largest spectral norm.

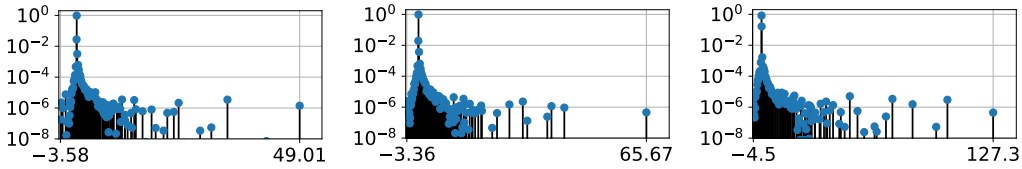


(a) $Acc = 94.4$, $\gamma = 0$ (b) $Acc = 96.46$, $\gamma = 10^{-4}$ (c) $Acc = 96.7$, $\gamma = 5 \times 10^{-4}$

Figure 2: Hessian spectrum for MLP after 50 epochs of SGD on the MNIST dataset, for various $L2$ regularisation coefficients γ

CNN: We consider a 9 layer simple convolutional neural network on the CIFAR-100 dataset (Dangel et al., 2019), with parameter count $1,387,108$ and a learning rate of $\alpha = 0.01$ for 300 epochs. We also observe that adding weight decay increases the spectral norm, as shown in Figure 3. For this network, training is also improved by the addition of a little $L2$ regularisation, but performance decreases for over regularisation, i.e. as the weight decay parameter increases from $[0, 10^{-4}, 5 \times 10^{-4}]$ the training performance is $[86.3\%, 87.9\%, 86.0\%]$. In this particular example the training accuracy is quite low, but there is still a generalisation gap $[32.4\%, 32.5\%, 31.1\%]$. Furthermore the generalisation gap is smaller for the network with the largest spectral norm, seen by comparing Figures 3c to Figures 3a and 3b.

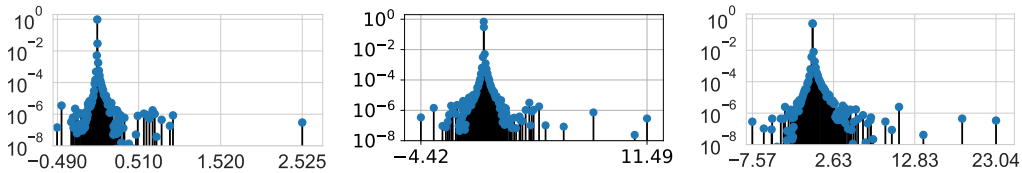
[4] multi-class equivalent of Logistic Regression



(a) $Val = 53.9$, $\gamma = 0$ (b) $Val = 55.4$, $\gamma = 10^{-4}$ (c) $Val = 54.8$, $\gamma = 5 \times 10^{-4}$

Figure 3: Hessian spectrum for CNN after 300 epochs of SGD on the CIFAR-100 dataset, for various $L2$ regularisation coefficients γ

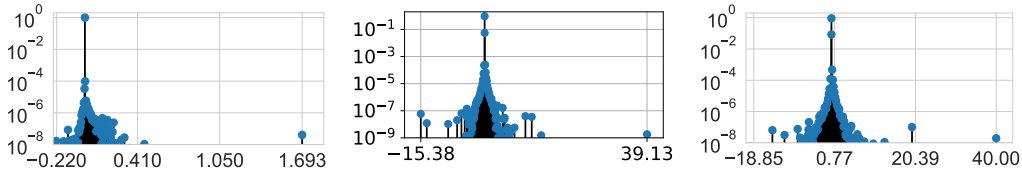
PreResNet-164 We use a pre-activated residual network on the CIFAR-100 dataset with parameter count $1,726,388$. Our training performance decreases with increased level of regularisation $[0, 10^{-4}, 5\times10^{-4}]$ from $[99.987\%, 99.985\%, 99.87\%]$ but our testing performance increases significantly $[72.78\%, 75.56\%, 76.76\%]$. The generalisation gap decreases as we increase the regularisation, yet the Hessian spectral norm continues to increase.



(a) $Val = 73.12$, $\gamma = 0$ (b) $Val = 75.96$, $\gamma = 1e^{-4}$ (c) $Val = 77.36$, $\gamma = 5e^{-4}$

Figure 4: Hessian spectrum for PreResNet-164 after 300 epochs of SGD on the CIFAR-100 dataset, for various $L2$ regularisation coefficients γ

WideResNet-28×10**:** We use a wide residual network on the CIFAR-100 dataset, with parameter count $36,546,980$, we observe the training accuracy remains roughly constant $[99.984\%, 99.984\%, 99.982\%]$ as we increase the regularisation from $[0, 10^{-4}, 5 \times 10^{-4}]$. We are now in the regime where the optimisation benefit of regularisation is negligible, but the generalisation benefit is significant. As shown in Figure 5, the Hessian spectral norm continues to increase with the increased regularisation coefficient γ. The validation set accuracies are $[75.2\%, 79.5\%, 80.6\%]$ and again we see the spectral norm increases as the generalisation gap decreases.



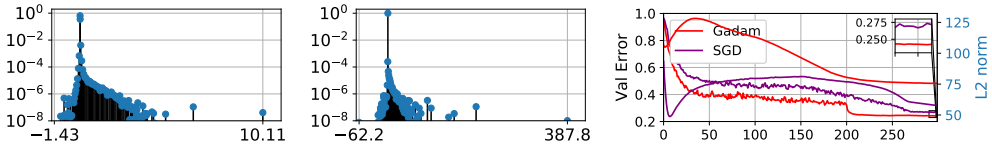
(a) $Val = 75.2$, $\lambda = 0$ (b) $Val = 79.50$, $\lambda = 1e^{-4}$ (c) $Val = 80.6$, $\lambda = 5e^{-4}$

Figure 5: Hessian spectrum for WideResNet28×10 after 300 epochs of SGD on the CIFAR-100 dataset, for various $L2$ regularisation coefficients γ, Batch Norm Train mode

Note on relationship between learning rate and spectral norm: Under a stability analysis Wu et al. (2018); Lewkowycz et al. (2020) argue that gradient descent must find $\lambda_1 \leq 2/\alpha$, whereas SGD satisfies a more restrictive condition, i.e the minima must be even flatter due to the condition of non-uniformity, specifically $\lambda_1 \leq 1 + \sqrt{1 - \alpha^2 \lambda_1(\text{Var}(\boldsymbol{H}))}$. Where $\lambda_1(\text{Var}(\boldsymbol{H}))$ is simply the largest eigenvalue of the variance of the Hessian. Although the authors Wu et al. (2018) remark that this bound is tight in deep learning experiments. We note for our ResNet experiments, with initial learning rate $\alpha = 0.1$, as we increase the weight decay, even the GD bound of 20 is over-reached. Whilst potentially the learning rate decrease in training by a factor of 100, brings the optimiser to a new region within the loss surface, the new bound of 2000 is certainly not tight.

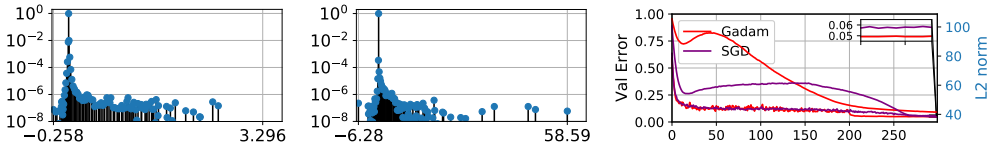
5 Sharpness and Adaptive optimisation

Given that all high performing solutions use some form of weight regularisation, we consider whether sharpness can be a useful indicator in the wild for the same neural network trained on the same dataset, but with alternative optimisers and schedules. Since out of the box adaptive gradient methods perform more poorly on the validation/test sets than SGD Wilson et al. (2017), we compare the sharpness of solutions of the Adam optimiser when combined with decoupled weight decay (Loshchilov and Hutter, 2018) and iterate averaging (Granziol et al., 2020b), which has been shown to generalise better than SGD. We use the VGG-16 with batch-normalisation on the CIFAR-10/100 datasets. We use a decoupled weight decay of 0.35/0.25 and a learning rate of $\alpha = 0.0005$. For SGD we use a weight decay γ of $3/5 \times 10^{-4}$ and a learning rate of $\alpha = 0.1$. We plot the validation accuracy curve for CIFAR-100 in Figure 6c, whilst we see that Adam clearly generalises better than SGD. As shown in Figures 6a and 6b, the spectral norm of the better performing Adam solutions is almost $40\times$ larger than the SGD solution, the Frobenius norm of Adam is 0.02 as opposed to 0.0001 for SGD. Both solutions give similar training performance, with Adam 99.81 and SGD 99.64. For CIFAR-10 although the generalisation gap is smaller, we see a similar picture, as shown in Figure 7. To investigate the fragility of other commonly employed metrics for this practical scenario, the Frobenius norm of the Adam solution is 1.55×10^{-3} as opposed to 1.43×10^{-5}. Furthermore we note from Figures 6c and 7c that not only are these solutions sharper, but they also have higher norms.



(a) $Val = 73.5$, SGD (b) $Val = 75.8$, Aadam (c) Test Error & $L2$ norm

Figure 6: Hessian spectrum for VGG-16BN after 300 epochs of SGD on the CIFAR-100 dataset, for various optimisation algorithms [SGD, Adam], batch norm train mode



(a) $Val = 94.3$, SGD (b) $Val = 95.1$, Adam (c) Test Error & $L2$ norm

Figure 7: Hessian spectrum for VGG-16BN after 300 epochs of SGD on the CIFAR-10 dataset, for various optimisation algorithms [SGD, Adam], batch norm train mode

6 Conclusion

In this paper we show that the Hessian can be written as the product of a polynomial and exponential in the weights, which in the limit of large weights, which overfit, give rise to flat minima despite poor generalisation. We show that adding $L2$ regularisation, significantly increases the spectral norm whilst improving generalisation. We also show that certain heuristics used to improve adaptive optimisation generalisation, give high weight norm sharp solutions, which generalise better than the corresponding flatter lower norm SGD solutions. This shows that whilst intuitive, pure Hessian based flatness measures are not relevant for generalisation. Solutions found in practical schedules which generalise significantly better can be much sharper.

References

Andrew J Ballard, Ritankar Das, Stefano Martiniani, Dhagash Mehta, Levent Sagun, Jacob D Stevenson, and David J Wales. Energy landscapes for machine learning. *Physical Chemistry Chemical Physics*, 19(20):12585–12603, 2017.

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Deep Frank-Wolfe for neural network optimization. *arXiv preprint arXiv:1811.07591*, 2018.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

Tony Cai, Jianqing Fan, and Tiefeng Jiang. Distributions of angles in random packing on spheres. *The Journal of Machine Learning Research*, 14(1):1837–1864, 2013.

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

Jinghui Chen and Quanquan Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks. *arXiv preprint arXiv:1806.06763*, 2018.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Felix Dangel, Frederik Kunstner, and Philipp Hennig. Backpack: Packing more into backprop. *arXiv preprint arXiv:1912.10985*, 2019.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1019–1028. JMLR. org, 2017.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via Hessian eigenvalue density. *arXiv preprint arXiv:1901.10159*, 2019.

Aditya Sharad Golatkar, Alessandro Achille, and Stefano Soatto. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In *Advances in Neural Information Processing Systems*, pages 10678–10688, 2019.

Gene H Golub and Gérard Meurant. Matrices, moments and quadrature. *Pitman Research Notes in Mathematics Series*, pages 105–105, 1994.

Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2012.

Diego Granziol, Xingchen Wan, Timur Garipov, Dmitry Vetrov, and Stephen Roberts. MLRG deep curvature. *arXiv preprint arXiv:1912.09656*, 2019.

Diego Granziol, Timur Garipov, Dmitry Vetrov, Stefan Zohren, Stephen Roberts, and Andrew Gordon Wilson. Towards understanding the true loss surface of deep neural networks using random matrix theory and iterative spectral methods, 2020a. URL `https://openreview.net/forum?id=H1gza2NtwH`.

Diego Granziol, Xingchen Wan, and Stephen Roberts. Iterate averaging helps: An alternative perspective in deep learning. *arXiv preprint arXiv:2003.01247*, 2020b.

Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. *arXiv preprint arXiv:1902.00744*, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

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

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

Stanisław Jastrzebski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the relation between the sharpest directions of DNN loss and the SGD step length. 2018.

Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The break-even point on the optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=r1g87C4KwB`.

Edwin T Jaynes. On the rationale of maximum-entropy methods. *Proceedings of the IEEE*, 70(9):939–952, 1982.

Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in neural information processing systems*, pages 586–594, 2016.

Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from Adam to SGD. *arXiv preprint arXiv:1712.07628*, 2017.

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

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

Yann LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2018.

Haihao Lu and Kenji Kawaguchi. Depth creates no bad local minima. *arXiv preprint arXiv:1702.08580*, 2017.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13132–13143, 2019.

Dhagash Mehta, Xiaojun Zhao, Edgar A Bernal, and David J Wales. Loss surface of xor artificial neural networks. *Physical Review E*, 97(5):052307, 2018.

Gérard Meurant and Zdeněk Strakoš. The Lanczos and conjugate gradient algorithms in finite precision arithmetic. *Acta Numerica*, 15:471–542, 2006.

Tristan Milne. Piecewise strong convexity of neural networks. In *Advances in Neural Information Processing Systems*, pages 12973–12983, 2019.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

Vardan Papyan. The full spectrum of deepnet hessians at scale: Dynamics with sgd training and sample size. *arXiv preprint arXiv:1811.07062*, 2018.

Barak A Pearlmutter. Fast exact multiplication by the Hessian. *Neural computation*, 6(1): 147–160, 1994.

Akshay Rangamani, Nam H Nguyen, Abhishek Kumar, Dzung Phan, Sang H Chin, and Trac D Tran. A scale invariant flatness measure for deep network minima. *arXiv preprint arXiv:1902.02434*, 2019.

Farbod Roosta-Khorasani and Uri Ascher. Improved bounds on sample size for implicit matrix trace estimators. *Foundations of Computational Mathematics*, 15(5):1187–1212, 2015.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Normalized flat minima: Exploring scale invariant definition of flat minima for neural networks using pac-bayesian analysis. *arXiv preprint arXiv:1901.04653*, 2019.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pages 4148–4158, 2017.

Lei Wu, Zhanxing Zhu, et al. Towards understanding generalization of deep learning: Perspective of loss landscapes. *arXiv preprint arXiv:1706.10239*, 2017.

Lei Wu, Chao Ma, and E Weinan. How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective. In *Advances in Neural Information Processing Systems*, pages 8279–8288, 2018.

Zhewei Yao, Amir Gholami, Qi Lei, Kurt Keutzer, and Michael W Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. In *Advances in Neural Information Processing Systems*, pages 4949–4959, 2018.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Chiyuan Zhang, Qianli Liao, Alexander Rakhlin, Brando Miranda, Noah Golowich, and Tomaso Poggio. Theory of deep learning iib: Optimization properties of sgd. *arXiv preprint arXiv:1801.02254*, 2018.

A Experiment Details

A.1 Image Classification Experiments

Hyper parameter Tuning For SGD and Gadam, we set the momentum parameter to be 0.9 whereas for Adam, we set $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 10^{-8}$, their default values. For SGD, we use a grid searched initial learning rates in the range of $[0.01, 0.03, 0.1]$ for all experiments with a fixed weight decay; for Adam and all its variants, we use grid searched initial learning rate range of $[10^{-4}, 3 \times 10^{-3}, 10^{-3}]$. After the best learning rate has been identified, we conduct a further search on the weight decay, which we find often leads to a trade off between the convergence speed and final performance. For CIFAR experiments, we search in the range of $[10^{-4}, 10^{-3}]$ whereas for ImageNet experiments, we search in the range of $[10^{-6}, 10^{-5}]$. For decoupled weight decay, we search the same range for the weight decay scaled by initial learning rate.

A.2 Experimental Details

For all experiments with SGD, we use the following learning rate schedule for the learning rate at the t-th epoch, similar to Izmailov et al. (2018):

$$\alpha_t = \begin{cases} \alpha_0, & \text{if } \frac{t}{T} \leq 0.5 \\ \alpha_0[1 - \frac{(1-r)(\frac{t}{T} - 0.5)}{0.4}] & \text{if } 0.5 < \frac{t}{T} \leq 0.9 \\ \alpha_0 r, & \text{otherwise} \end{cases} \tag{14}$$

where α_0 is the initial learning rate. In the motivating logistic regression experiments on MNIST, we used $T = 50$. $T = 300$ is the total number of epochs budgeted for all CIFAR experiments. We set $r = 0p01$ for all experiments. For experiments with iterate averaging, we use the following learning rate schedule instead:

$$\alpha_t = \begin{cases} \alpha_0, & \text{if } \frac{t}{T_{avg}} \leq 0.5 \\ \alpha_0[1 - \frac{(1-\frac{\alpha_{avg}}{\alpha_0})(\frac{t}{T} - 0.5)}{0.4}] & \text{if } 0.5 < \frac{t}{T_{avg}} \leq 0.9 \\ \alpha_{avg}, & \text{otherwise} \end{cases} \tag{15}$$

where α_{avg} refers to the (constant) learning rate after iterate averaging activation, and in this paper we set $\alpha_{avg} = \frac{1}{2}\alpha_0$. T_{avg} is the epoch after which iterate averaging is activated, and the methods to determine T_{avg} was described in the main text. This schedule allows us to adjust learning rate smoothly in the epochs leading up to iterate averaging activation through a similar linear decay mechanism in the experiments without iterate averaging, as described above.

B Lanczos algorithm

In order to empirically analyse properties of modern neural network spectra with tens of millions of parameters $N = \mathcal{O}(10^7)$, we use the Lanczos algorithm (Meurant and Strakoš, 2006), provided for deep learning by Granziol et al. (2019). It requires Hessian vector products, for which we use the Pearlmutter trick (Pearlmutter, 1994) with computational cost $\mathcal{O}(NP)$, where N is the dataset size and P is the number of parameters. Hence for m steps the total computational complexity including re-orthogonalisation is $\mathcal{O}(NPm)$ and memory cost of $\mathcal{O}(Pm)$. In order to obtain accurate spectral density estimates we re-orthogonalise at every step (Meurant and Strakoš, 2006). We exploit the relationship between the Lanczos method and Gaussian quadrature, using random vectors to allow us to learn a discrete approximation of the spectral density. A quadrature rule is a relation of the form,

$$\int_a^b f(\lambda) d\mu(\lambda) = \sum_{j=1}^{M} \rho_j f(t_j) + R[f] \tag{16}$$

for a function f, such that its Riemann-Stieltjes integral and all the moments exist on the measure $d\mu(\lambda)$, on the interval $[a, b]$ and where $R[f]$ denotes the unknown remainder. The

nodes t_j of the Gauss quadrature rule are given by the Ritz values and the weights (or mass) ρ_j by the squares of the first elements of the normalised eigenvectors of the Lanczos tridiagonal matrix (Golub and Meurant, 1994). The main properties of the Lanczos algorithm are summarized in the theorems 2,3

Theorem 2. *Let $H^{N\times N}$ be a symmetric matrix with eigenvalues $\lambda_1 \geq .. \geq \lambda_n$ and corresponding orthonormal eigenvectors $z_1, ..z_n$. If $\theta_1 \geq .. \geq \theta_m$ are the eigenvalues of the matrix T_m obtained after m Lanczos steps and $q_1, ...q_k$ the corresponding Ritz eigenvectors then*

$$\begin{aligned} \lambda_1 \geq \theta_1 \geq \lambda_1 - \frac{(\lambda_1 - \lambda_n)\tan^2(\theta_1)}{(c_{k-1}(1+2\rho_1))^2} \\ \lambda_n \leq \theta_k \leq \lambda_m + \frac{(\lambda_1 - \lambda_n)\tan^2(\theta_1)}{(c_{k-1}(1+2\rho_1))^2} \end{aligned} \tag{17}$$

where c_k is the chebyshev polyomial of order k

Proof: see (Golub and Van Loan, 2012).

Theorem 3. *The eigenvalues of T_k are the nodes t_j of the Gauss quadrature rule, the weights w_j are the squares of the first elements of the normalised eigenvectors of T_k*

Proof: See (Golub and Meurant, 1994). The first term on the RHS of equation 16 using Theorem 3 can be seen as a discrete approximation to the spectral density matching the first m moments $v^T H^m v$ (Golub and Meurant, 1994; Golub and Van Loan, 2012), where v is the initial seed vector. Using the expectation of quadratic forms, for zero mean, unit variance random vectors, using the linearity of trace and expectation

$$\mathbb{E}_v \mathrm{Tr}(v^T H^m v) = \mathrm{Tr}\mathbb{E}_v(vv^T H^m) = \mathrm{Tr}(H^m) = \sum_{i=1}^{N} \lambda_i = N \int_{\lambda\in\mathcal{D}} \lambda d\mu(\lambda) \tag{18}$$

The error between the expectation over the set of all zero mean, unit variance vectors v and the monte carlo sum used in practice can be bounded (Hutchinson, 1990; Roosta-Khorasani and Ascher, 2015). However in the high dimensional regime $N \to \infty$, we expect the squared overlap of each random vector with an eigenvector of H, $|v^T \phi_i|^2 \approx \frac{1}{N} \forall i$, with high probability. This result can be seen by computing the moments of the overlap between Rademacher vectors, containing elements $P(v_j = \pm 1) = 0.5$. Further analytical results for Gaussian vectors have been obtained (Cai et al., 2013).

C Mathematical Preliminaries

For an input/output pair $[\boldsymbol{x} \in \mathbb{R}^{d_x}, \boldsymbol{y} \in \mathbb{R}^{d_y}]$ and a given model $h(\cdot;\cdot) : \mathbb{R}^{d_x} \times \mathbb{R}^P \to \mathbb{R}^{d_y}$. Without loss of generality, we consider the family of models functions parameterized by the weight vector $\boldsymbol{w}$, i.e., $\mathcal{H} := \{h(\cdot;\boldsymbol{w}) : \boldsymbol{w} \in \mathbb{R}^P\}$, with a given loss $\ell(h(\boldsymbol{x};\boldsymbol{w}), \boldsymbol{y}) : \mathbb{R}^{d_y} \times \mathbb{R}^{d_y} \to \mathbb{R}$.

The empirical risk (often denote the *loss* in deep learning), its gradient and Hessian are given by

$$R_{emp}(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \ell(h(\boldsymbol{x}_i;\boldsymbol{w}), \boldsymbol{y}_i),\ \boldsymbol{g}_{emp}(\boldsymbol{w}) = \nabla R_{emp},\ \boldsymbol{H}_{emp}(\boldsymbol{w}) = \nabla^2 R_{emp} \tag{19}$$

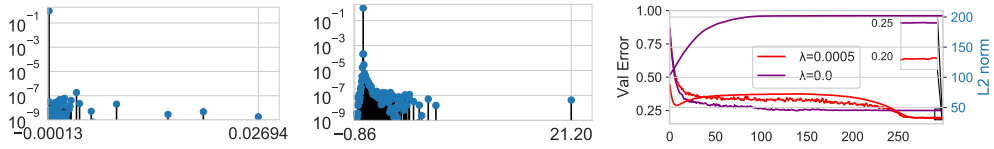
The Hessian describes the curvature at that point in weight space $\boldsymbol{w}$ and hence the risk surface can be studied through the Hessian. By the spectral theorem, we can rewrite $\boldsymbol{H}_{emp}(\boldsymbol{w}) = \sum_{i=1}^{P} \lambda_i \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T$ in terms of its eigenvalue, eigenvector pairs $[\lambda_i, \boldsymbol{\phi}_i]$. In order to characterise $\boldsymbol{H}_{emp}(\boldsymbol{w})$ by a single value, authors typically consider the spectral norm, which is given by the largest eigenvalue of $\boldsymbol{H}_{emp}(\boldsymbol{w})$ or the normalised trace, which gives the mean eigenvalue. The Hessian contains P^2 elements, so cannot be stored or eigendecomposed for all but the simplest of models. Stochastic Lanczos Quadrature can be used Meurant and Strakoš (2006), with computational complexity $\mathcal{O}(P)$ to give tight bounds on the extremal eigenvalues and good estimations of $\mathrm{Tr}(\boldsymbol{H})$ and $\mathrm{Tr}(\boldsymbol{H}^2)$, along with a moment matched

approximation of the spectrum. We use the Deep Learning implementation provided by Granziol et al. (2019). DNNs are typically trained using stochastic gradient descent with momentum, where we iteratively update the weights

$$\begin{aligned} \boldsymbol{z}_{k+1} &\leftarrow \rho \boldsymbol{z}_k + \nabla R(\boldsymbol{w}_k) \\ \boldsymbol{w}_{k+1} &\leftarrow \boldsymbol{w}_k - \alpha \boldsymbol{z}_{k+1} \end{aligned} \tag{20}$$

Where ρ is the momentum. The gradient is usually taken on a randomly selected sub-sample of size $B \ll N$. An epoch is defined as a full training pass of the data, so comprises $\approx N/B$ iterations. Often $L2$ regularisation (also termed weight decay) is added to the loss, which corresponds to $R_{emp}(\boldsymbol{w}) \rightarrow R_{emp}(\boldsymbol{w}) + \mu/2||\boldsymbol{w}||^2$.

How does batch normalisation affect curvature? During training both the mean and variance of the batch normalisation layers are adapted to the specific batch, whereas at evaluation they are fixed (to their exponentially moving average). This is done so that the transforms can function even if the prediction set is only 1 sample[5]. Previous works investigating neural network Hessians (Papyan, 2018; Ghorbani et al., 2019) do not consider this free parameter in batch-normalisation and its effect on the spectrum. From a sharpness and generalisation perspective, we would consider that it is the model that is making predictions that we should evaluate. Changing batch normalisation to the evaluation mode, we find that a somewhat different curvature profile, as shown in Figure 8. In this case the sharpness of the regularised solution in terms of the spectral norm is nearly 1000 times larger than that of the regularised, better generalising solution. The Frobenius norm, for the regularised solution is 4.9×10^{-5} as opposed to 9.8×10^{-12}, so $\mathcal{O}(10^7)$ larger.



(a) $Val = 75.2$, $\lambda = 0$ (b) $Val = 80.6$, $\lambda = 5e^{-4}$ (c) Test Error & $L2$ norm

Figure 8: Hessian spectrum for WideResNet28×10 after 300 epochs of SGD on the CIFAR-100 dataset, for various $L2$ regularisation coefficients λ, batch norm evaluation mode

[5]a 1 sample set has no variance