A Subtleties of Batch Normalisation

Most modern convolutional neural networks use batch normalisation Ioffe and Szegedy (2015). Whilst we refer the reader to Ioffe and Szegedy (2015) for a full description of batch normalisation, a subtlety which is introduced is that the output of a network for one sample depends on the samples in the mini-batch (as they are used to compute the statistics). Since these statistics are used to normalise and center the layers, they have a large effect on the curvature calculation. In our software package, we give the user two options to deal with this. We denote one **Train Mode** and the other **Evaluation Mode**. We detail them both below

A.1 Train Mode

With batch norm in train mode the output of the network for one sample depends on other samples in the mini-batch. This effect influences the result of hessian-vector product computation on a dataset. In particular, if the order of samples in the dataset is changed, then the set of mini-batches also changes and the hessian-vector product will be different. We take this effect into account and fix the order of samples in the dataset, so that hessian-vector product for different networks is computed on the same set of mini-batches. Conceptually, one can use the following interpretation of a neural network with batch norm layers in train mode: the input of such a network is not a single sample but a mini-batch of samples. Instead of thinking about a dataset of samples we should think about a dataset of mini-batches. With the fixed order of samples, we guarantee that all models are evaluated on the same dataset of mini-batches.

A.2 EvaluationMode

One can also use BN layers in eval mode with BN statistics computed over the whole dataset. With BN layers in eval mode the output of a network on sample does not depend on other samples in the mini-batch. Our code the mode of BN layers can be specified as a parameter for hessian-vector product computation.

B Learning to love Lanczos

The Lanczos Algorithm, on which our package is based, (Algorithm 1) is an iterative algorithm for learning a subset of the eigenvalues/eigenvectors of any Hermitian matrix, requiring only matrix vector products. It can be regarded as a far superior adaptation of the power iteration method, where the Krylov subspace $\mathscr{K}(\boldsymbol{H},\boldsymbol{v}) = \text{span}\{\boldsymbol{v},\boldsymbol{H}^{\boldsymbol{v}},\boldsymbol{H}^2\boldsymbol{v}...\}$ is orthogonalised using Gram-Schmidt. Beyond having improved convergence to the power iteration method (Bai et al., 1996) by storing the intermediate orthogonal vectors in the corresponding Krylov subspace, Lanczos produces estimates of the eigenvectors and eigenvalues of smaller absolute magnitude, known as Ritz vectors/values. Despite its known superiority to the power iteration method and relationship to orthogonal polynomials and hence when combined with random vectors the ability to estimate the entire spectrum of a matrix, these properties are often ignored or forgotten by practitioners. We compare against diaognal approximations for random and synthetic matrices in Appendix E.

The Lanczos method be be explicitly derived by considering the optimization of the Rayleigh quotient (Golub and Van Loan, 2012)

$$r(\boldsymbol{v}) = \frac{\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{v}} \tag{5}$$

over the entire Krylov subspace $\mathcal{K}_m(\boldsymbol{H},\boldsymbol{v})$ as opposed to power iteration which is a particular vector in the Krylov subspace $\boldsymbol{u} = \boldsymbol{H}^m\boldsymbol{v}$. Despite this, practitioners looking to learn the leading eigenvalue, very often resort to the power iteration, likely due to its implementational simplicity. We showcase the power iterations relative inferiority to Lanczos with the following convergence theorems

Theorem 1. *Let $H^{P\times P}$ be a symmetric matrix with eigenvalues $\lambda_1 \geq .. \geq \lambda_P$ and corresponding orthonormal eigenvectors $z_1, ..z_P$. If $\theta_1 \geq .. \geq \theta_m$ are the eigenvalues of the matrix T_m obtained*

λ_1/λ_2	$m=5$	$m=10$	$m=15$	$m=20$
1.5	$\frac{1.1\times10^{-4}}{3.9\times10^{-2}}$	$\frac{2\times10^{-10}}{6.8\times10^{-4}}$	$\frac{3.9\times10^{-16}}{1.2\times10^{-5}}$	$\frac{7.4\times10^{-22}}{2.0\times10^{-7}}$
1.1	$\frac{2.7\times10^{-2}}{4.7\times10^{-1}}$	$\frac{5.5\times10^{-5}}{1.8\times10^{-1}}$	$\frac{1.1\times10^{-7}}{6.9\times10^{-2}}$	$\frac{2.1\times10^{-10}}{2.7\times10^{-2}}$
1.01	$\frac{5.6\times10^{-1}}{9.2\times10^{-1}}$	$\frac{1.0\times10^{-1}}{8.4\times10^{-1}}$	$\frac{1.5\times10^{-2}}{7.6\times10^{-1}}$	$\frac{2.0\times10^{-3}}{6.9\times10^{-1}}$

Table 6: L_{k-1}/R_{k-1} For different values of spectral gap λ_1/λ_2 and iteration number m, Table from (Golub and Van Loan, 2012)

after m Lanczos steps and $q_1, ...q_m$ the corresponding Ritz eigenvectors then

$$\begin{aligned} \lambda_1 \geq \theta_1 \geq \lambda_1 - \frac{(\lambda_1 - \lambda_n)\tan^2(\theta_1)}{(c_{m-1}(1+2\rho_1))^2} \\ \lambda_P \leq \theta_k \leq \lambda_m + \frac{(\lambda_1 - \lambda_n)\tan^2(\theta_1)}{(c_{m-1}(1+2\rho_1))^2} \end{aligned} \tag{6}$$

where c_m is the Chebyshev polyomial of order k. $\cos\theta_1 = |\boldsymbol{q}_1^T \boldsymbol{z}_1|$ *&* $\rho_1 = (\lambda_1 - \lambda_2)/(\lambda_2 - \lambda_n)$

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

Theorem 2. *Assuming the same notation as in Theorem 1, after m power iteration steps the corresponding extremal eigenvalue estimate is lower bounded by*

$$\lambda_1 \geq \theta_1 \geq \lambda_1 - (\lambda_1 - \lambda_n)\tan^2(\theta_1)\left(\frac{\lambda_2}{\lambda_1}\right)^{2m-1} \tag{7}$$

From the rapid growth of orthogonal polynomials such as Chebyshev, we expect Lanczos superiority to significantly emerge for larger spectral gap and iteration number. To verify this experimentally, we collect the non identical terms in the equations 6 and 7 of the lower bounds for λ_1 derived by Lanczos and Power iteration and denote them L_{k-1} and R_{k-1} respectively. For different values of λ_1/λ_2 and iteration number m we give the ratio of these two quatities in Table 6. As can be clearly seen, the Lanczos lower bound is always closer to the true value, this improves with the iteration number m and its relative edge is reduced if the spectral gap is decreased.

B.1 The Problem of Moments: Spectral Density Estimation Using Lanczos

In this section we show that that the Lanczos Tri-Diagonal matrix corresponds to an orthogonal polynomial basis which matches the moments of $\boldsymbol{v}^T\boldsymbol{H}^m\boldsymbol{v}$ and that when $\boldsymbol{v}$ is a zero mean random vector with unit variance, this corresponds to the moment of the underlying spectral density.

Stochastic trace estimation Using the expectation of quadratic forms, for zero mean, unit variance random vectors

$$\begin{aligned} \mathbb{E}_{\boldsymbol{v}}\mathrm{Tr}(\boldsymbol{v}^T\boldsymbol{H}^m\boldsymbol{v}) &= \mathrm{Tr}\mathbb{E}_{\boldsymbol{v}}(\boldsymbol{v}\boldsymbol{v}^T\boldsymbol{H}^m) = \mathrm{Tr}(\boldsymbol{H}^m) \\ &= \sum_{i=1}^{P}\lambda_i^m = P\int_{\lambda\in\mathcal{D}}\lambda^m d\mu(\lambda) \end{aligned} \tag{8}$$

where we have used the linearity of trace and expectation. Hence in expectation over the set of random vectors, the trace of the inner product of $\boldsymbol{v}$ and $\boldsymbol{H}^m\boldsymbol{v}$ is equal to the m'th moment of the spectral density of $\boldsymbol{H}$.

Lanczos-Stieltjes The Lanczos tri-diagonal matrix $\boldsymbol{T}$ can be derived from the Moment matrix $\boldsymbol{M}$, corresponding to the discrete measure $d\alpha(\lambda)$ satisfying the moments $\mu_i = \boldsymbol{v}^T\boldsymbol{H}^i\boldsymbol{v} = \int\lambda^i d\alpha(\lambda)$ (Golub and Meurant, 1994)

$$\boldsymbol{M} = \begin{bmatrix} 1 & \boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} & \dots & \boldsymbol{v}^T\boldsymbol{H}^{m-1}\boldsymbol{v} \\ \boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v} & \boldsymbol{v}^T\boldsymbol{H}^2\boldsymbol{v} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{v}^T\boldsymbol{H}^{m-1}\boldsymbol{v} & \dots & \dots & \boldsymbol{v}^T\boldsymbol{H}^{2m-2}\boldsymbol{v} \end{bmatrix}$$

and hence for a zero mean unit variance initial seed vector, the eigenvector/eigenvalue pairs of $\boldsymbol{T}$ contain information about the spectral density of $\boldsymbol{H}$ as shown in section B.1. This is given by the following Theorem

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 normalized eigenvectors of T_k*

Proof. See Golub and Meurant (1994) □

A quadrature rule is a relation of the form,

$$\int_a^b f(\lambda)d\mu(\lambda) = \sum_{j=1}^{M} \rho_j f(t_j) + R[f] \tag{9}$$

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 first term on the RHS of equation 9 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)

For n_v starting vectors, the corresponding discrete spectral density is given as

$$p(\lambda) = \frac{1}{n_v}\sum_{l=1}^{n_v}\left(\sum_{k=1}^{m}(\tau_k^{(l)})^2\delta(\lambda - \lambda_k^{(l)})\right), \tag{10}$$

where $\tau_k^{(l)}$ corresponds to the first entry of the eigenvector of the k-th eigenvalue, λ_k, of the Lanczos tri-diagonal matrix, $\boldsymbol{T}$, for the l-th starting vector (Ubaru and Saad; Lin et al., 2016).

B.2 Computational Complexity

For large matrices, the computational complexity of the algorithm depends on the Hessian vector product, which for neural networks is $\mathcal{O}(mNP)$ where P denotes the number of parameters in the network, m is the number of Lanczos iterations and N is the number of data-points. The full re-orthogonalisation adds two matrix vector products, which is of cost $\mathcal{O}(m^2P)$, where typically $m^2 \ll N$. Each random vector used can be seen as another full run of the Lanczos algorithm, so for d random vectors the total complexity is $\mathcal{O}(dmP(N + m))$

Importance of keeping orthogonality The update equations of the Lanczos algorithm lead to a tri-diagonal matrix $\boldsymbol{T} = \mathbb{R}^{m\times m}$, whose eigenvalues represent the approximated eigenvalues of the matrix $\boldsymbol{H}$ and whose eigenvectors, when projected back into the the Krylov-subspace, $\mathscr{K}(\boldsymbol{H}, \boldsymbol{v})$, give the approximated eigenvectors of $\boldsymbol{H}$. In finite precision, it is known (Meurant and Strakoš, 2006) that the Lanczos algorithm fails to maintain orthogonality between its Ritz vectors, with corresponding convergence failure. In order to remedy this, we re-orthonormalise at each step (Bai et al., 1996) (as shown in line 9 of Algorithm 1) and observe a high degree of orthonormality between the Ritz eigenvectors. Orthonormality is also essential for achieving accurate spectral resolution as the Ritz value weights are given by the squares of the first elements of the normalised eigenvectors. For the practitioner wishing to reduce the computational cost of maintaining orthogonality, there exist more elaborate schemes (Meurant and Strakoš, 2006; Golub and Meurant, 1994).

We also explicitly debunk some key persistent myths, given below.

- We can learn the negative and interior eigenvalues by shifting and inverting the matrix sign $\boldsymbol{H} \to -\boldsymbol{H} + \mu\boldsymbol{I}$
- Lanczos learns the largest m $[\lambda_i, \boldsymbol{u}_i]$ pairs of $\boldsymbol{H} \in \mathbb{R}^{P\times P}$ with high probability (Dauphin et al., 2014)

Since these two related beliefs are prevalent, we disprove them explicitly in this section, with Theorems 4 and 5.

Theorem 4. *The shift and invert procedure $\boldsymbol{H} \to -\boldsymbol{H} + \mu\boldsymbol{I}$, changes the Eigenvalues of the Tridiagonal matrix $\boldsymbol{T}$ (and hence the Ritz values) to $\lambda_i = -\lambda_i + \mu$*

Algorithm 1 Lanczos Algorithm

```
1: Input: Hessian vector product $\{\boldsymbol{Hv}\}$, number of steps $m$
2: Output: Ritz eigenvalue/eigenvector pairs $\{\lambda_i, \boldsymbol{u}_i\}$ & quadrature weights $\tau_i$
3: Set $\boldsymbol{v} := \boldsymbol{v}/\sqrt{(\boldsymbol{v}^T\boldsymbol{v})}$
4: Set $\beta := 0$, $\boldsymbol{v}_{old} := \boldsymbol{v}$
5: Set $\boldsymbol{V}(:,1) := \boldsymbol{v}$
6: for $j$ in $1,..,m$ do
7:     $\boldsymbol{w} = \boldsymbol{Hv} - \beta\boldsymbol{v}_{old}$
8:     $\boldsymbol{T}(j,j) = \alpha = \boldsymbol{w}^T\boldsymbol{v}$
9:     $w = \boldsymbol{w} - \alpha\boldsymbol{w} - \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{w}$
10:    $\beta = \sqrt{\boldsymbol{w}^T\boldsymbol{w}}$
11:    $\boldsymbol{v}_{old} = \boldsymbol{v}$
12:    $\boldsymbol{v} = \boldsymbol{w}/\beta$
13:    $\boldsymbol{V}(:,j+1) = \boldsymbol{v}$
14:    $\boldsymbol{T}(j,j+1) = \boldsymbol{T}(j+1,1) = \beta$
15: end for
16: $\{\lambda_i, \boldsymbol{e}_i\} = eig(\boldsymbol{T})$
17: $\boldsymbol{u}_i = \boldsymbol{V}\boldsymbol{e}_i$
18: $\tau_i = (\boldsymbol{e}_i^T[1,0,0...0])^2$
```

Proof. Following the equations from Algorithm 1

$$\begin{aligned}
&\boldsymbol{w}_1^T = (-\boldsymbol{H} + \mu\boldsymbol{I})\boldsymbol{v}_1 \ \& \ \alpha_1 = \boldsymbol{v}_1^T\boldsymbol{H}\boldsymbol{v}_1 + \mu\boldsymbol{I} \\
&\boldsymbol{w}_2 = \boldsymbol{w}_1 - \alpha_1\boldsymbol{v}_1 = (\boldsymbol{H} + \mu\boldsymbol{I})\boldsymbol{v}_1 - (\boldsymbol{v}_1^T\boldsymbol{H}\boldsymbol{v}_1 + \mu\boldsymbol{I})\boldsymbol{v}_1 \\
&\boldsymbol{w}_2 = (\boldsymbol{H} - \boldsymbol{v}_1^T\boldsymbol{H}\boldsymbol{v}_1)\boldsymbol{v}_1 \ \& \ \boldsymbol{v}_2 = \boldsymbol{w}_2/||\boldsymbol{w}_2|| \\
&\alpha_2 = \boldsymbol{v}_2^T(-\boldsymbol{H} + \mu\boldsymbol{I})\boldsymbol{v}_2 = -\boldsymbol{v}_2^T\boldsymbol{H}\boldsymbol{v}_2 + \mu \\
&\beta_2 = ||\boldsymbol{w}_2||
\end{aligned} \tag{11}$$

Assuming this for $m-1$, and repeating the above steps for m we prove by induction and finally arrive at the modified tridiagonal Lanczos matrix $\tilde{\boldsymbol{T}}$

$$\begin{aligned}
&\tilde{\boldsymbol{T}} = -\boldsymbol{T} + \mu\boldsymbol{I} \\
&\tilde{\lambda}_i = -\lambda_i + \mu \ \forall 1 \leq i \leq m
\end{aligned} \tag{12}$$

$\square$

Remark. No new Eigenvalues of the matrix $\boldsymbol{H}$ are learned. Although it is clear that the addition of the identity does not change the Krylov subspace, such procedures are commonplace in code pertaining to papers attempting to find the *smallest eigenvalue*. This disproves the first misconception.

Theorem 5. *For any matrix $\boldsymbol{H} \in \mathbb{R}^{P\times P}$ such that $\lambda_1 > \lambda_2 > ..... > \lambda_P$ and $\sum_{i=1}^m \lambda_i < \sum_{i=m+1}^P \lambda_i$ in expectation over the set of random vectors $\boldsymbol{v}$ the m eigenvalues of the Lanczos Tridiagonal matrix $\boldsymbol{T}$ do not correspond to the top m eigenvalues of $\boldsymbol{H}$*

Proof. Let us consider the matrix $\tilde{\boldsymbol{H}} = \boldsymbol{H} - \frac{\lambda_{m+1}+\lambda_m}{2}\boldsymbol{I}$,

$$\begin{cases} \lambda_i > 0, & \forall i \leq m \\ \lambda_i < 0, & \forall i > m \end{cases} \tag{13}$$

Under the assumptions of the theorem, Tr$(\tilde{\boldsymbol{H}}) < 0$ and hence by Theorem 3 and Equation 8 there exist no $w_i > 0$ such that

$$\sum_{i=1}^m w_i\lambda_i^k = \frac{1}{P}\sum_{i=1}^P \lambda_i^k \forall \quad 1 \leq k \leq m \tag{14}$$

is satisfied for $k=1$, as the LHS is manifestly positive and the RHS is negative. By Theorem 4 this holds for the original matrix $\boldsymbol{H}$. $\square$

Remark. Given that Theorem 5 is satisfied over the expectation of the set of random vectors, which by the CLT is realised by Monte Carlo draws of random vectors as $d \to \infty$ the only way to really span the top m eigenvectors is to have selected a vector which lies in the m dimensional subspace of the P dimensional problem corresponding to those vectors, which would correspond to knowing those vectors *a priori*, defeating the point of using Lanczos at all.

Remark. Given the relationship between the moments of the spectral density and the expectation over the set of random vectors, one may be curious to ask how we expect the deviation to be depending on the number of random vectors actually used in practice. Whilst usually packages using Lanczos use a number of random vectors (increasing the corresponding computational cost), in Appendix C we demonstrate that we expect the difference to reduce as we increase the problem dimension P. This informs our choice of only using a single random vector in our package, we compare results for different random vectors in Appendix C and find minimal differences.

C Effect of Varying Random Vectors

Given that the proofs for the moments of Lanczos matching those of the underlying spectral density, are true over the expectation over the set of random vectors and in practice we only use a Monte Carlo average of random vectors, or in our experiments using stem plots, just a single random vector. We justify this with the following Lemma

Lemma 1. *Let $\boldsymbol{u} \in \mathbb{R}^{P\times 1}$ random vector, where $\boldsymbol{u}_i$ is zero mean and unit variance and finite 4'th moment $\mathbb{E}[\boldsymbol{u}_i^4] = m_4$. Then for $\boldsymbol{H} \in \mathbb{R}^{P\times P}$, then*

$$i)\mathbb{E}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}] = \mathrm{Tr}\,\boldsymbol{H}$$
$$ii)\mathrm{Var}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}] \leq (2+m_4)\,\mathrm{Tr}(\boldsymbol{H}^T\boldsymbol{H})$$

Proof.

$$\mathbb{E}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}] = \sum_{i,j=1}^{P}\boldsymbol{H}_{i,j}\mathbb{E}[\boldsymbol{u}_i\boldsymbol{v}_j] = \sum_{i=1}^{P}\boldsymbol{H}_{i,i} = \mathrm{Tr}\,\boldsymbol{H} \tag{15}$$

$$\begin{aligned}\mathbb{E}[||\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}||^2] &= \sum_{i,j}\sum_{k,l}\boldsymbol{H}_{i,j}\boldsymbol{H}^T_{k,l}\mathbb{E}[\boldsymbol{u}_i\boldsymbol{u}_j^T\boldsymbol{u}_k\boldsymbol{u}_l^T]\\ &\sum_{i,j}\sum_{k,l}\boldsymbol{H}_{i,j}\boldsymbol{H}^T_{k,l}[\delta_{i,j}\delta_{k,l}+\delta_{i,l}\delta_{j,k}+\delta_{i,k}\delta_{j,l}+m_4\delta_{i,j,k,l}]\\ &= (\mathrm{Tr}\,\boldsymbol{H})^2 + (2+m_4)\,\mathrm{Tr}(\boldsymbol{H}^2)\end{aligned} \tag{16}$$

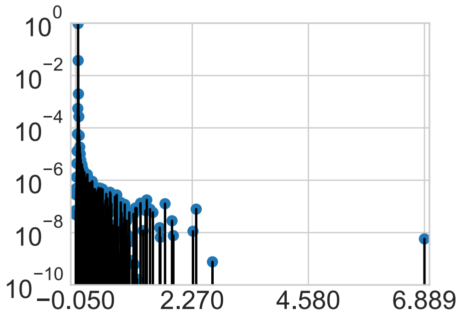
□

Remark. Let us consider the signal to noise ratio for some positive definite $\boldsymbol{H} \succ c\boldsymbol{I}$

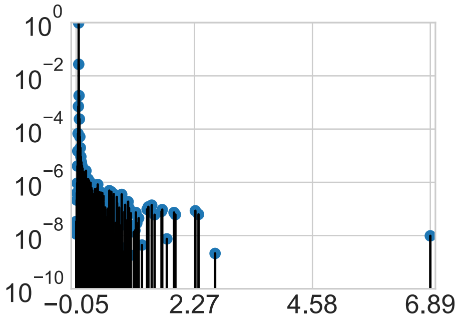
$$\begin{aligned}\left(\frac{\sqrt{\mathrm{Var}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}]}}{\mathbb{E}[\boldsymbol{u}^T\boldsymbol{H}\boldsymbol{u}]}\right)^2 &\propto \frac{1}{1+\frac{\sum_{i\neq j}^{P}\lambda_i\lambda_j}{\sum_k^P\lambda_k^2}} = \frac{1}{1+\frac{P-1\langle\lambda_i\lambda_j\rangle}{\langle\lambda_k^2\rangle}}\\ &\leq \frac{1}{1+\frac{P-1}{\kappa^2}}\end{aligned} \tag{17}$$

where $\langle..\rangle$ denotes the arithmetic average. For the extreme case of all eigenvalues being identical, the condition number $\kappa = 1$ and hence this reduces to $1/P \to 0$ in the $P \to \infty$ limit, whereas for a rank-1 matrix, this ratio remains 1. For the MP density, which well models neural network spectra, κ is not a function of P as $P \to \infty$ and hence we also expect this benign dimensional scaling to apply.

We verify this high dimensional result experimentally, by running spectral visualisation but using two different random vectors. We plot the results in Figure 5. We find both figures 5a & 5b to be close to visually indistinguishable. There are minimal differences in the extremal eigenvalues, with former giving $\{\lambda_1, \lambda_n\} = \{6.8885, -0.0455\}$ and the latter $\{6.8891, -0.0456\}$, but the degeneracy at 0, bulk, triplet of outliers at 2.27 and the large outlier at 6.89 is unchanged.



(a) Vector 2



(b) Vector 3

Figure 5: VGG16 Epoch 300 end of training Lanczos stem plot for different random vectors

C.1 Why we don't kernel smooth

Concurrent work, which has also used Lanczos with the Pearlmutter trick to learn the Hessian (Yao et al., 2018; Ghorbani et al., 2019), typically uses n_v random vectors and then uses kernel smoothing, to give a final density. In this section we argue that beyond costing a factor of n_v more computationally, that the extra compute extended in order to get more accurate moment estimates, which we already argued in Section C are asymptotically error free, is wasted due to the kernel smoothing (Granziol et al., 2019). The smoothed spectral density takes the form:

$$\tilde{p}(\lambda) = \int k_\sigma(\lambda - \lambda')p(\lambda')d\lambda' = \sum_{i=1}^{n} w_i k_\sigma(\lambda - \lambda_i) \tag{18}$$

We make some assumptions regarding the nature of the kernel function, $k_\sigma(\lambda - \lambda_i)$, in order to prove our main theoretical result about the effect of kernel smoothing on the moments of the underlying spectral density. Both of our assumptions are met by (the commonly employed) Gaussian kernel.

Assumption 1. *The kernel function $k_\sigma(\lambda - \lambda_i)$ is supported on the real line $[-\infty, \infty]$.*

Assumption 2. *The kernel function $k_\sigma(\lambda - \lambda_i)$ is symmetric and permits all moments.*

Theorem 6. *The m-th moment of a Dirac mixture $\sum_{i=1}^{n} w_i \delta(\lambda - \lambda_i)$, which is smoothed by a kernel k_σ satisfying assumptions 1 and & 2, is perturbed from its unsmoothed counterpart by an amount $\sum_{i=1}^{n} w_i \sum_{j=1}^{r/2} \binom{r}{2j} \mathbb{E}_{k_\sigma(\lambda)}(\lambda^{2j})\lambda_i^{m-2j}$, where $r = m$ if m is even and $m-1$ otherwise. $\mathbb{E}_{k_\sigma(\lambda)}(\lambda^{2j})$ denotes the $2j$-th central moment of the kernel function $k_\sigma(\lambda)$.*

Proof. The moments of the Dirac mixture are given as,

$$\langle \lambda^m \rangle = \sum_{i=1}^{n} w_i \int \delta(\lambda - \lambda_i)\lambda^m d\lambda = \sum_{i=1}^{n} w_i \lambda_i^m. \tag{19}$$

The moments of the modified smooth function (Equation equation 18) are

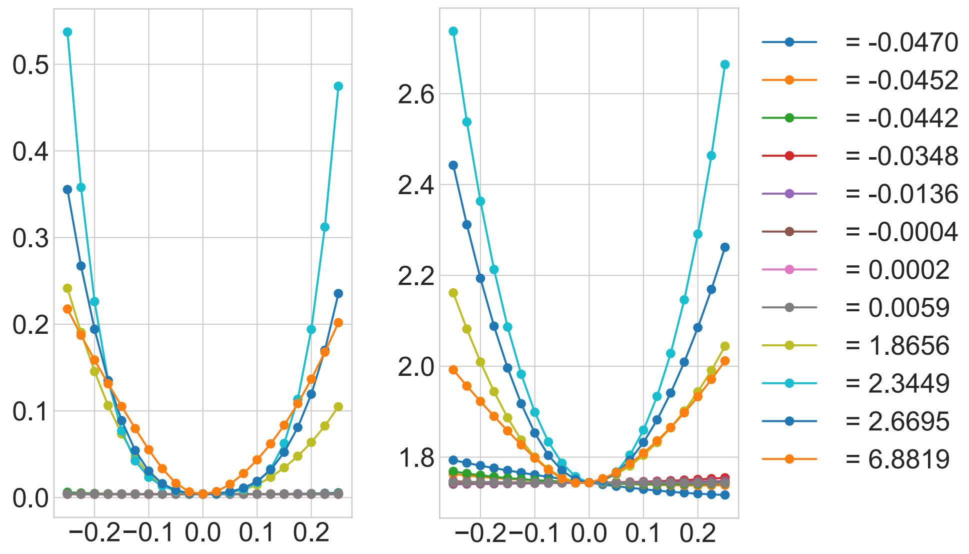
$$\begin{aligned} \langle \tilde{\lambda}^m \rangle &= \sum_{i=1}^{n} w_i \int k_\sigma(\lambda - \lambda_i)\lambda^m d\lambda \\ &= \sum_{i=1}^{n} w_i \int k_\sigma(\lambda')(\lambda' + \lambda_i)^m d\lambda' \\ &= \langle \lambda^m \rangle + \sum_{i=1}^{n} w_i \sum_{j=1}^{r/2} \binom{r}{2j} \mathbb{E}_{k_\sigma(\lambda)}(\lambda^{2j})\lambda_i^{m-2j}. \end{aligned} \tag{20}$$

We have used the binomial expansion and the fact that the infinite domain is invariant under shift reparametarization and the odd moments of a symmetric distribution are 0. □

Remark. The above proves that kernel smoothing alters moment information, and that this process becomes more pronounced for higher moments. Furthermore, given that $w_i > 0$, $\mathbb{E}_{k_\sigma(\lambda)}(\lambda^{2j}) > 0$ and (for the GGN $lambda_i > 0$, the corrective term is manifestly positive, so the smoothed moment estimates are biased.

D LOCAL LOSS LANDSCAPE

The Lanczos algorithm with enforced orthogonality initialised with a random vector gives a moment matched discrete approximation to the Hessian spectrum. However this information is local to the point in weight space $\boldsymbol{w}$ and the quadratic approximation may break down within the near vicinity. To investigate this, we use the **loss landscape visualisation** function of our package: We display this for the VGG-16 on CIFAR-100 in Figure 8. We see for the training loss 6a that the eigenvector corresponding to the largest eigenvalue $\lambda = 6.88$ only very locally corresponds to the sharpest increase in loss for the training, with other extremal eigenvectors, corresponding to the eigenvalues $\lambda = \{2.67, 2.35\}$ overtaking it in loss change relatively rapidly. Interestingly for the testing loss, all the extremal eigenvectors change the loss much more rapidly, contradicting previous assertions that the test loss is a "shifted" version of the training loss (He et al., 2019; Izmailov et al., 2018). We do however note some small asymmetry between the changes in loss along the opposite ends of the eigenvectors. The flat directions remain flat locally and some of the eigen-vectors corresponding to negative values correspond to decreases in test loss. We include the code in **??**



(a) Training Loss

(b) Testing Loss

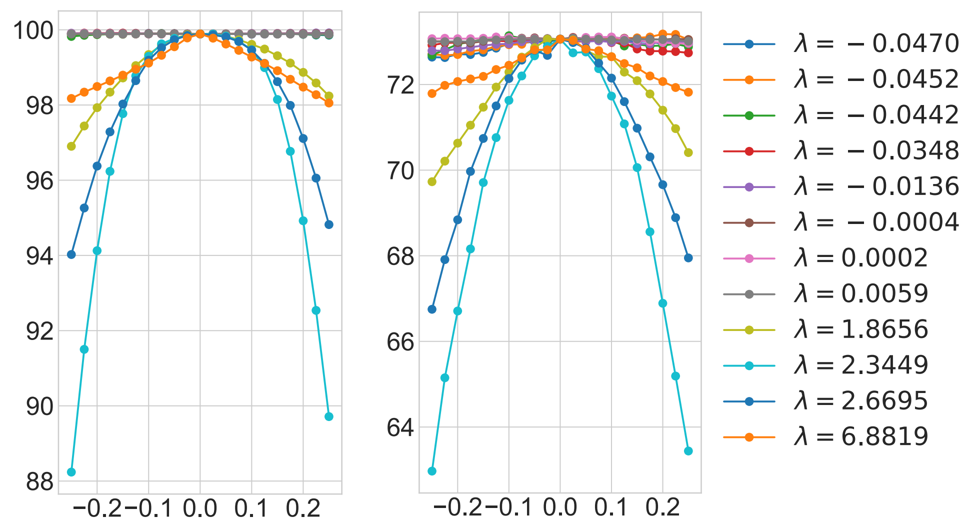
Figure 6: VGG-16 CIFAR-100 Loss surface visualised along 6 negative and position eigenvalues

CIFAR-10 DATASET

To showcase the ability of our software to handle multiple datasets we display the Hessian of the VGG-16 trained in an identical fashion as its CIFAR-100 counterpart of CIFAR-10 in Figure **??**, along with the a plot of a selection of Ritz vectors traversing the training loss surface in Figure 8a and testing loss surface in Figure 8b along with also the training accuracy surface (Figure 9a) and testing accuracy surface (Figure 9b).

E EXAMPLES ON SMALL RANDOM MATRICES

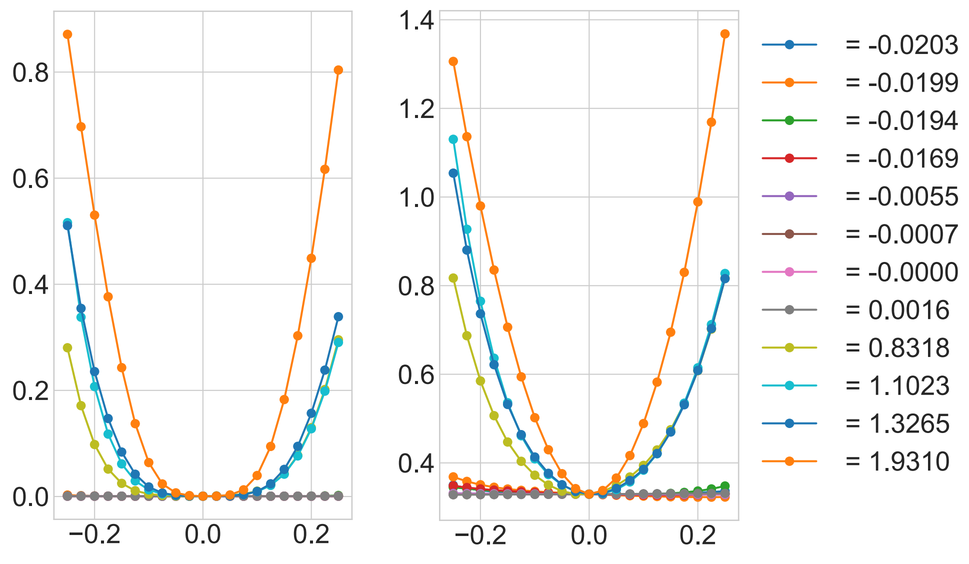
In this section, we use some examples on small random matrices to showcase the power of our package that uses the Lanczos algorithm with random vectors to learn the spectral density. Here, we look at known random matrices with elements drawn from specific distributions which converge



(a) Training Accuracy

(b) Testing Accuracy

Figure 7: VGG-16 CIFAR-100 Accuracy surface visualised along 6 negative and position eigenvalues



(a) Training Loss

(b) Testing Loss

Figure 8: VGG-16 CIFAR-10 Loss surface visualised along 6 negative and position eigenvalues

to known spectral densities in the asymptotic limit. Here we consider **Wigner Ensemble** (Wigner, 1993) and the **Marcenko Pastur** (Marchenko and Pastur, 1967), both of which are extensively used in simulations or theoretical analyses of deep neural network spectra (Pennington and Bahri, 2017; Choromanska et al., 2015a; Anonymous, 2020).

E.1 Wigner Matrices

Wigner matrices can be defined in Definition E.1, and their distributions of eigenvalues are governed by the semi-circle distribution law (Theorem 7).

Definition E.1. *Let $\{Y_i\}$ and $\{Z_{ij}\}_{1\leq i\leq j}$ be two real-valued families of zero mean, i.i.d random variables, Furthermore suppose that $\mathbb{E}Z_{12}^2 = 1$ and for each $k \in \mathbb{N}$*

$$max(E|Z_{12}^k, E|Y_1|^k) < \infty \tag{21}$$

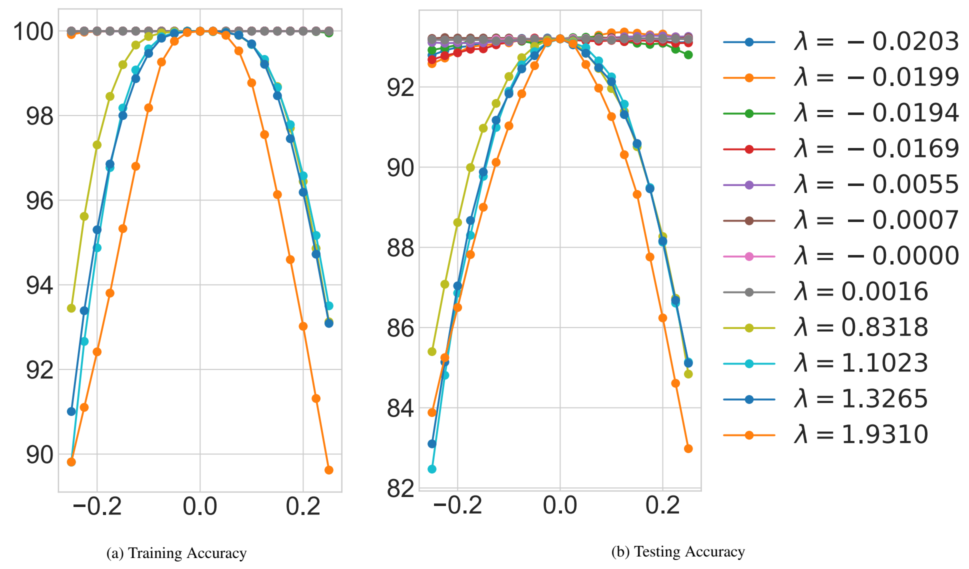


Figure 9: VGG-16 CIFAR-10 Accuracy surface visualised along negative and positive eigenvalues

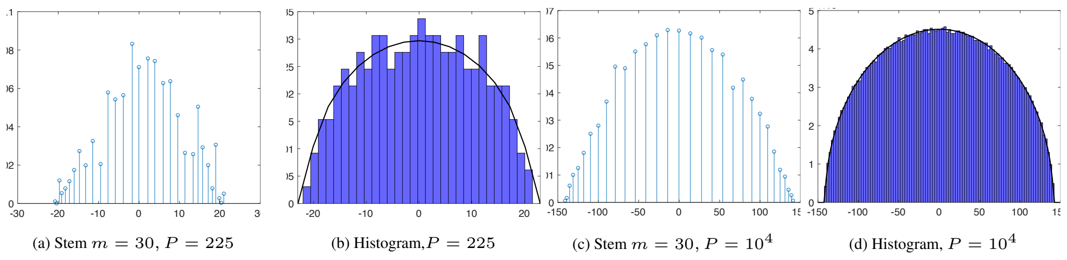


Figure 10: Lanczos stem plot for a single random vector with $m = 30$ steps compared to actual eigenvalue histogram for matrices of the form $\boldsymbol{H} \in \mathbb{R}^{P\times P}$, where each element is a drawn from a normal distribution with unit variance, converging to the Wigner semi circle.

Consider a $P \times P$ symmetric matrix M_P, whose entries are given by

$$\begin{cases} M_P(i,i) = Y_i \\ M_P(i,j) = Z_{ij} = M_P(j,i), \quad \textit{if } x \geq 1 \end{cases} \tag{22}$$

The Matrix M_P is known as a real symmetric Wigner matrix.

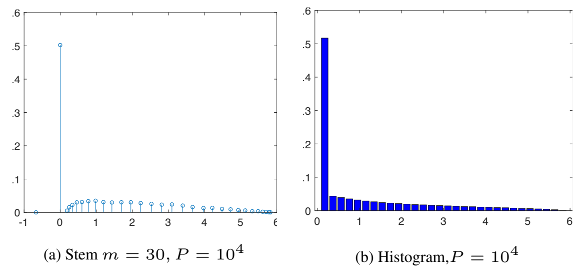
Theorem 7. *Let $\{M_P\}_{P=1}^{\infty}$ be a sequence of Wigner matrices, and for each P denote $X_P = M_P/\sqrt{P}$. Then μ_{X_P}, converges weakly, almost surely to the semi circle distribution,*

$$\sigma(x)dx = \frac{1}{2\pi}\sqrt{4 - x^2}\mathbf{1}_{|x|\leq 2} \tag{23}$$

For our experiments, we generate random matrices $\boldsymbol{H} \in \mathbb{R}^{P\times P}$ with elements drawn from the distribution $\mathcal{N}(0,1)$ for $P = \{225, 10000\}$ and plot histogram of the spectra found by eigendecomposition, along with the predicted Wigner density (scaled by a factor of $\sqrt{P}$) in Figures 10b & 10d and compare them along with the discrete spectral density approximation learned by lanczos in $m = 30$ steps using a single random vector $d = 1$ in Figures 10a & 10c. It can be seen that even for a small number of steps $m \ll P$ and a single random vector, Lanczos impressively captures not only the support of the eigenvalue spectral density but also its shape. We note as discussed in section B.2 that the 30 Ritz values here do not span the top 30 eigenvalues even approximately.

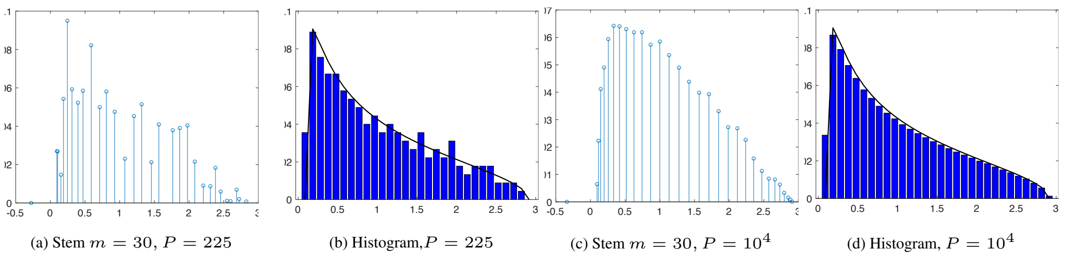
E.2 Marcenko-Pastur

An equally important limiting law for the limiting spectral density of many classes of matrices constrained to be positive definite, such as covariance matrices, is the Marcenko-Pastur law (Marchenko


(a) Stem $m = 30$, $P = 10^4$

(b) Histogram, $P = 10^4$

Figure 11: Lanczos stem plot for a single random vector with $m = 30$ steps compared to actual eigenvalue histogram for matrices of the form $\boldsymbol{H} \in \mathbb{R}^{P\times P}$, where $\boldsymbol{H} = \boldsymbol{X}\boldsymbol{X}^T/k$, where each element of $\boldsymbol{X}^{P\times k}$, $k = 0.5P$ is a drawn from a normal distribution with unit variance, converging to the Marcenko-Pastur distribution with $q = 0.5$.


(a) Stem $m = 30$, $P = 225$

(b) Histogram, $P = 225$

(c) Stem $m = 30$, $P = 10^4$

(d) Histogram, $P = 10^4$

Figure 12: Lanczos stem plot for a single random vector with $m = 30$ steps compared to actual eigenvalue histogram for matrices of the form $\boldsymbol{H} \in \mathbb{R}^{P\times P}$, where $\boldsymbol{H} = \boldsymbol{X}\boldsymbol{X}^T/k$, where each element of $\boldsymbol{X}^{P\times k}$, $k = 2P$ is a drawn from a normal distribution with unit variance, converging to the Marcenko-Pastur distribution with $q = 2$.

and Pastur, 1967). Formally, given a matrix $\boldsymbol{X} \in \mathbb{R}^{P\times T}$ with i.i.d zero mean entires with variance $\sigma^2 < \infty$. Let $\lambda_1 \geq \lambda_2, ... \geq \lambda_P$ be eigenvalues of $\boldsymbol{Y}_n = \frac{1}{T}\boldsymbol{X}\boldsymbol{X}^T$. The random measure $\mu_P(A) = \frac{1}{P}\#\{\lambda_j \in A\}, A \in \mathbb{R}$

Theorem 8. *Assume that $P, N \to \infty$ and the ratio $P/N \to q \in (0, \infty)$ (this is known as the Kolmogorov limit) then $\mu_P \to \mu$ in distribution where*

$$\begin{cases} (1-\frac{1}{q})\mathbb{1}_{0\in A} + \nu_{1/q}(A), & \text{if } q > 1 \\ \nu_q(A), & \text{if } 0 \leq q \leq 1 \end{cases} \tag{24}$$

$$d\nu_q = \frac{\sqrt{(\lambda_+ - x)(x - \lambda_-)}}{\lambda x 2\pi\sigma^2}, \lambda_\pm = \sigma^2(1 \pm \sqrt{q})^2 \tag{25}$$

Here, we construct a random matrix $\boldsymbol{X} \in \mathbb{P} \times \mathbb{T}$ with independently drawn elements from the distribution $\mathcal{N}(0, 1)$ and then form the matrix $\frac{1}{T}\boldsymbol{X}\boldsymbol{X}^T$, which is known to converge to the Marcenko-Pastur distribution. We use $P = \{225, 10000\}$ and $T = 2P$ and plot the associated histograms from full eigendecomposition in Figures 12b & 12d along with their $m = 30, d = 1$ Lanczos stem counterparts in Figures 12a & 12c. Similarly we see a faithful capturing not just of the support, but also of the general shape. We note that both for Figure 10 and Figure 12, the smoothness of the discrete spectral density for a single random vector increases significantly, even relative to the histogram.

We also run the same experiment for $P = 10000$ but this time with $T = 0.5P$ so that exactly half of the eigenvalues will be 0. We compare the Histogram of the eigenvalues in Figure 11b against its $m = 30, d = 1$ Lanczos stem plot in Figure 11a and find both the density at the origin, along with the bulk and support to be faithfully captured.

E.3 Comparison to Diagonal Approximations

As a proxy for deep neural network spectra, often the diagonal of the matrix (Bishop, 2006) or the diagonal of a surrogate matrix, such as the Fisher information, or that implied by the values of the Adam Optimizer (Chaudhari et al., 2016) is used. We plot the true eigenvalue estimates for random matrices pertaining to both the Marcenko-Pastur (Fig. 14a) and the Wigner density (Fig. 14b) in blue, along with the Lanczos estimate in red and the diagonal approximation in yellow. We see here

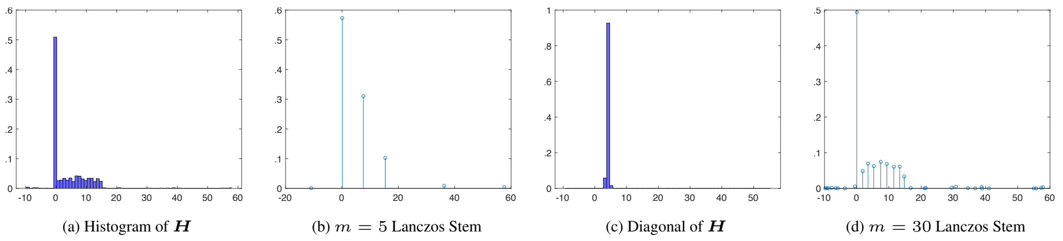

(a) Histogram of $\boldsymbol{H}$ (b) $m = 5$ Lanczos Stem (c) Diagonal of $\boldsymbol{H}$ (d) $m = 30$ Lanczos Stem

Figure 13: Generated matrices $\boldsymbol{H} \in \mathbb{R}^{1000 \times 1000}$ with known eigenspectrum and Lanczos stem plots for different values of $m = \{5, 15, 30\}$

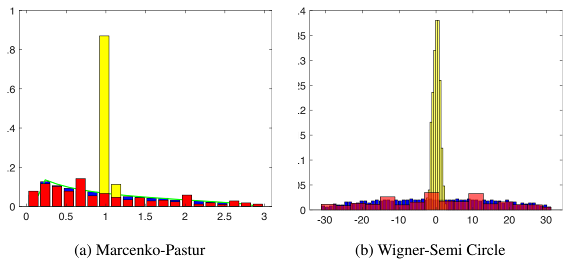

(a) Marcenko-Pastur (b) Wigner-Semi Circle

Figure 14: Two randomly generated matrices $\boldsymbol{H} \in \mathbb{R}^{500 \times 500}$ with the histogram of the true eigenvalues in blue, the Lanczos estimate $m = 30, d = 1$ in red and the diagonal approximation in yellow

that the diagonal approximation in both cases, fails to adequately the support or accurately model the spectral density, whereas the lanczos estimate is nearly indistinguishable from the true binned eigenspectrum. This is of-course obvious from the mathematics of the un-normalised Wigner matrix. The diagonal elements are simply draws from the normal distribution $\mathcal{N}(0, 1)$ and so we expect the diagonal histogram plot to approximately follow this distribution (with variance 1). However the second moment of the Wigner Matrix can be given by the Frobenius norm identity

$$\mathbb{E}\left(\frac{1}{P}\sum_{i}^{P}\lambda_i^2\right) = \mathbb{E}\left(\frac{1}{P}\sum_{i,j=1}^{P}\boldsymbol{H}_{i.j}^2\right) = \mathbb{E}\left(\frac{1}{P}\chi^2_{P^2}\right) = P \tag{26}$$

Similarly for the Marcenko-Pastur distribution, We can easily see that each element of $\boldsymbol{H}$ follows a chi-square distribution of $1/T\chi^2_T$, with mean 1 and variance $2/T$.

E.4 Synthetic Example

The curvature eigenspectrum of neural network often features a large spike at zero, a right-skewed bulk and some outliers (Sagun et al., 2016; 2017).[3] In order to simulate the spectrum of a neural network, we generate a Matrix $\boldsymbol{H} \in \mathbb{R}^{1000 \times 1000}$ with 470 eigenvalues drawn from the uniform distribution from $[0, 15]$, 20 drawn from the uniform $[0, 60]$ and 10 drawn from the uniform $[-10, 0]$. The matrix is rotated through a rotation matrix U, i.e $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$ where $\boldsymbol{D}$ is the diagonal matrix consisting of the eigenvalues and the columns are gaussian random vectors which are orthogonalised using Gram-Schmidt orthogonalisation. The resulting eigenspectrum is given in a histogram in Figure 13a and then using the same random vector, successive Lanczos stem plots for different number of iterations $m = [5, 30]$ are shown in Figure 13. Figure 13b, for a low number of steps, the degeneracy at $\lambda = 0$ is learned, as are the largest and smallest eigenvalues, some information is retained about the bulk density, but some of the outlier eigenvalues around $\lambda \approx 20$ and $\lambda \approx 30$ are completely missed out, along with all the negative outliers except the largest. For $m = 30$ even the shape of the bulk is accurately represented, as shown in Figure 13d. Here, we would like to emphasise that learning the outliers is important in the neural network context, as they relate to important properties of the network and the optimisation process (Ghorbani et al., 2019).

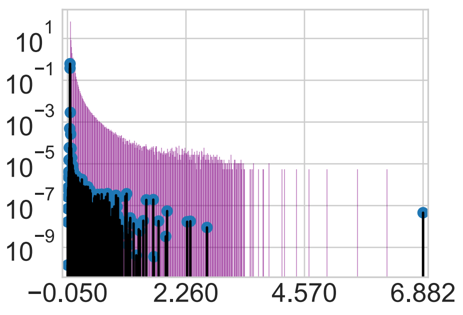
On the other hand, we note that the diagonal estimate in Figure 13c gives absolutely no spectral information, with no outliers shown (maximal and minimal diagonal elements being 5.3 and 3.3

[3]Some examples of this can be found in later sections on real-life neural network experiments - see Figures 15 and 5.

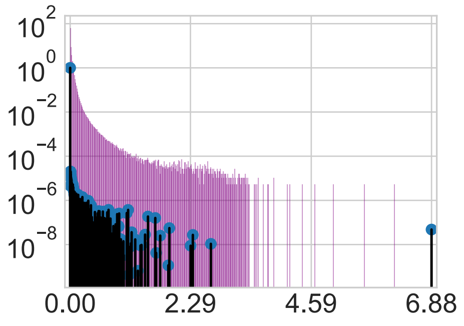
respectively and it also gets the spectral mass at 0 wrong. This builds on section E.3, as furthering the case against making diagonal approximations in general. In neural networks, the diagonal approximation is similar to positing no correlations between the weights. This is a very harsh assumption and usually a more reasonable assumption is to posit that the correlations between weights in the same layer are larger than between different layers, leading to a block diagonal approximation (Martens, 2016), however often when the layers have millions of parameters, full diagonal approximations are still used. (Bishop, 2006; Chaudhari et al., 2016).

E.5 Comparison to Backpack

For 16-layer VGG network, on the CIFAR-100 dataset the GGN diagonal computations Dangel et al. (2019) require over 125GB of GPU memory. Hence we use their Monte Carlo approximation to the GGN diagonal against both our GGN-Hessian-Lanczos spectral visualizations. We plot a histogram of the Monte Carlo approximation of the diagonal GGN (Diag-GGN) against both the Lanczos GGN (Lanc-GGN) and Lanczos Hessian (Lanc-Hess) in Figure 15. Note that as the Lanc-GGN and Lanc-Hess are displayed as stem plots (with the discrete spectral density summing to 1 as opposed to the histogram area summing to 1). We note that the Gauss-Newton approximation quite closely resembles its Hessian counterpart, capturing the majority of the bulk and the outlier eigenvectors at $\lambda_1 \approx 6.88$ and the triad near $\lambda_i \approx 2.29$. The Hessian does still have significant spectral mass on the negative axis, around 37%. However most of this is captured by a Ritz value at -0.0003, with this removed, the negative spectral mass is only 0.05%. However the Diag-GGN gives a very poor spectral approximation. It vastly overestimates the bulk region, which extends well beyond $\lambda \approx 1$ implied by Lanczos and adds many spurious outliers between 3 and the misses the largest outlier of 6.88. *Computational Cost* Using a single NVIDIA GeForce GTX 1080 Ti GPU, the Gauss-Newton takes an average 26.5 seconds for each Lanczos iteration with the memory useage 2850Mb. Using the Hessian takes an average of 27.9 seconds for each Lanczos iteration with 2450Mb memory usage.



(a) Diag-GGN & Lanc-GGN



(b) Diag-GGN & Lanc-Hess

Figure 15: Diagonal Generalised Gauss-Newton monte carlo approximation (Diag-GGN) against $m = 100$ Lanczos using Gauss-Newton vector products (Lanc-GGN) or Hessian vector products (Lanc-Hess)