REPRESENTING MODEL UNCERTAINTY OF NEURAL NETWORKS IN SPARSE INFORMATION FORM

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper addresses the problem of representing a system's belief using Multivariate Normal Distributions (MNDs) where the underlying model is based on a deep neural network (DNN). The major challenge with DNNs is the computational complexity that is needed to obtain model uncertainty using MNDs. To achieve a scalable method, we propose a novel approach that expresses the parameter posterior in sparse information form. Our inference algorithm is based on a novel Laplace Approximation scheme, which involves a diagonal correction of the Kronecker-factored eigenbasis. As this makes the inversion of the information matrix intractable - an operation that is required for full Bayesian analysis, we devise a low-rank approximation of this eigenbasis and a memory-efficient sampling scheme. We provide both a theoretical analysis and an empirical evaluation on various benchmark data sets, showing the superiority of our approach over existing methods.

1 INTRODUCTION

Whenever machine learning methods are used for safety-critical and health-related applications such as medical image analysis, autonomous driving, or mobile robotics, it is crucial to provide a precise estimation of the failure probability of the learned predictor. Therefore, most of the current learning approaches return distributions rather than single, most-likely predictions. For example, DNNs trained for classification usually use the softmax function to provide a distribution over predicted class labels. Unfortunately, this method tends to severely underestimate the true failure probability, leading to *overconfident* predictions (Guo et al., 2017). The main reason for this is that neural networks are typically trained with a principle of *maximum likelihood*, neglecting their *epistemic* or model uncertainty with point estimates of the parameters.

A recent work by Gal (2016) shows that this can be mitigated by using dropout at test time. This so-called Monte-Carlo dropout (MC-dropout) has the advantage that it is relatively easy to use and therefore very popular in practice. However, MC-dropout also has significant drawbacks. First, it requires a specific stochastic regularization during training. This limits its use on already well trained architectures, because current networks are often trained with other regularization techniques such as batch normalization. Moreover, it uses a Bernoulli distribution to represent the complex model uncertainty, which again leads to an underestimation of the predictive uncertainty.

Concurrently, several researchers proposed alternatives without these drawbacks of MC-dropout. Variational inference (Khan et al., 2018; Kingma et al., 2015; Graves, 2011) and expectation propagation (Herandez-Lobato & Adams, 2015) are such examples. However, these methods use a diagonal covariance matrix which limits their applicability as the model parameters are often highly correlated in deep architectures. Using so-called matrix normal distribution (Gupta & Nagar, 1999), Sun et al. (2017); Louizos & Welling (2017; 2016); Zhang et al. (2018); Ritter et al. (2018a) show that the correlations between the parameters can also be computed. This representation has the advantage that its covariance can be decomposed into Kronecker products of smaller matrices. However, the distribution suffers from the fact that not all matrices can be Kronecker decomposed. As a result, it usually induces crude approximations of the covariance (Bae et al., 2018).

On the other hand, MND approaches, which do not have these drawbacks, are difficult to exploit in the context of deep learning. This is due to the fact that the covariance scales quadratic to the number of parameters, resulting in intractable storage, let alone the inference. Therefore, we propose

to represent the MND in sparse information form and focus on designing an associated inference framework. As a first step, we propose a new Laplace Approximation (LA) for DNNs, in which we improve the Kronecker factored approximations of the Hessian (George et al., 2018) by "correcting" the diagonal variance in parameter space. We show that these can be computed efficiently, and that the *information matrix* of the resulting parameter posterior is more accurate in terms of the Frobenius norm. Unlike existing works, this results in estimating the model uncertainty in *information form* of the MND.

Furthermore, we propose a novel low-rank representation of the resulting Kronecker factorization, which paves the way to applications on large network structures trained on realistically sized data sets. To realize such sparsification, we propose a novel algorithm that enables a low-rank approximation of the Kronecker factored eigenvalue decomposition, and we further introduce a memory-efficient sampling scheme. As we demonstrate in experiments on benchmark data sets, our approach provides lower calibration errors and more accurate uncertainty estimates for out-of-distribution samples. Our method is still efficient in terms of memory and practical, as it does not require changes in the training procedure. A detailed theoretical analysis is also provided for further insights.

In summary, our contributions are as follows.

- A new Laplace Approximation scheme with a diagonal correction to the "eigenvalue re-scaled" approximations of the Hessian as a practical inference tool. Unlike others, our method infers the model uncertainty in *information* form of Multivariate Normal Distribution.
- A novel low-rank representation of Kronecker factored eigendecomposition for tractable sampling from Multivariate Normal Distribution. For this, we introduce a novel algorithm to enable a low rank approximation (LRA) of the Kronecker-factored and eigenvalue-decomposed matrices. We also demonstrate a memory-efficient sampling algorithm.

As a result, our approach serves as a sparse formulation of the MND in deep Bayesian Neural Networks.

1.1 Related Works

Approximation of the Hessian: Instead of diagonal approximations, e.g. (Becker & Lecun, 1989; Salimans & Kingma, 2016), several researchers have focused on various ways of approximating the Hessian, e.g. (C. Liu & Nocedal, 1989; Roux & Fitzgibbon, 2010; Hennig, 2013). Amongst them, layer-wise Kronecker Factor approximation of Grosse & Martens (2016); Martens & Grosse (2015); Botev et al. (2017); Chen et al. (2018) have demonstrated a notable scalability (Ba et al., 2017). A recent extension can be found in (George et al., 2018) where the authors propose to re-scale the eigen-values of the Kronecker factored matrices so that the diagonal variance in its eigenbasis is accurate. The work presents an interesting idea as one can prove that in terms of a Frobenius norm, the proposed approximation is more accurate than that of Martens & Grosse (2015). However, as this approximation is harmed by inaccurate estimate of eigenvectors, we propose to further correct the diagonal elements in the parameter space.

Laplace Approximation: Amongst various methods such as variational inference (Hinton & van Camp, 1993) and sampling (Neal, 1996), we focus on LA as a practical inference framework (MacKay, 1992). Recently, diagonal (Becker & Lecun, 1989) and Kronecker-factored approximations (Botev et al., 2017) to the Hessian have been applied to LA by Ritter et al. (2018a). The authors have further proposed to use LA in continual learning (Ritter et al., 2018b), and have demonstrated a competitive results by significantly outperforming its benchmarks (Kirkpatrick et al., 2017; Zenke et al., 2017). We build upon Ritter et al. (2018a) but propose to use more expressive posterior distribution than matrix normal distribution. SLANG (Mishkin et al., 2018) share similar spirit to ours in using a low-rank plus diagonal form of covariance where the authors show the effects of low-rank approximation in detail. Yet, SLANG is different to ours as they do not explore Kronecker structures. SLANG has been tested in relatively small data sets.

Sparse Information Filters: Inspired by (Thrun et al., 2004) from SLAM literature, we introduce sparse information matrix in the context of approximate inference. A main difference, however, is that DNNs have high dimensions and thus, we explore Kronecker structure.

2 Background and Notation

We model a neural network as a parameterized function $f_\theta : \mathbb{R}^{N_1} \rightarrow \mathbb{R}^{N_l}$ where $\theta \in \mathbb{R}^{N_\theta}$ are the weights and $N_\theta = N_1 + \cdots + N_l$. This function f_θ is in fact a concatenation of l layers, where each layer $\mathrm{i} \in \{1, ..., l\}$ computes $h_\mathrm{i} = W_\mathrm{i} a_{\mathrm{i}-1}$ and $a_\mathrm{i} = \phi(h_{\mathrm{i}-1})$. Here, ϕ is a nonlinear function, a_i are activations, h_i linear pre-activations, and W_i are weight matrices. The bias terms are absorbed into W_i by appending 1 to each a_i. Thus, $\theta = \begin{bmatrix} vec(W_1)^T & vec(W_2)^T & ... & vec(W_l)^T \end{bmatrix}^T$ where vec is the operator that stacks the columns of a matrix to a vector. Let $g_\mathrm{i} = \delta h_\mathrm{i}$, the gradient of h_i w.r.t θ.

Using LA, the parameter posterior can be approximated with a Gaussian, in which, the mean is given by the MAP estimate θ_{MAP} and the covariance by the Hessian of the log-likelihood $(H + \tau I)^{-1}$ (assuming a Gaussian prior with precision τ). When using standard loss functions such as MSE or cross entropy and piece-wise linear a_i (e.g RELU), a good approximation of the Hessian is given by the Fisher information matrix (IM) $\boldsymbol{I} = \mathbb{E}\left[\delta\theta\delta\theta^T\right]$ [1]. This matrix is of size $N_\theta \times N_\theta$, which means that it is intractable even for moderately sized neural networks. Therefore, we use an approximation based on a Kronecker factorization of $\boldsymbol{I}$.

To make the computation tractable, a number of approximations are applied. First, it is assumed that the weights across layers are uncorrelated, which corresponds to a block-diagonal form of $\boldsymbol{I}$ with blocks $\boldsymbol{I}_1, \boldsymbol{I}_2, \ldots, \boldsymbol{I}_l$. Then, each realisation of block $\boldsymbol{I}_i$ is represented as a Kronecker product

$$\delta\theta_\mathrm{i}\delta\theta_\mathrm{i}^T = a_{\mathrm{i}-1}a_{\mathrm{i}-1}^T \otimes g_\mathrm{i}g_\mathrm{i}^T. \tag{1}$$

This has the advantage that the representation is more compact and that the inverse can be computed efficiently using $(A \otimes G)^{-1} = A^{-1} \otimes G^{-1}$. Then, in a further approximation step, matrices $A_{\mathrm{i}-1}$ and G_i are assumed to be statistically independent, leading to

$$\boldsymbol{I}_{\mathrm{i,kfac}} = \mathbb{E}\left[a_{\mathrm{i}-1}a_{\mathrm{i}-1}^T \otimes g_\mathrm{i}g_\mathrm{i}^T\right] \approx \mathbb{E}\left[a_{\mathrm{i}-1}a_{\mathrm{i}-1}^T\right] \otimes \mathbb{E}\left[g_\mathrm{i}g_\mathrm{i}^T\right] = A_{\mathrm{i}-1} \otimes G_\mathrm{i}. \tag{2}$$

We refer to Martens & Grosse (2015) for details on the Kronecker Factored Approximate Curvature (KFAC). As demonstrated by Ba et al. (2017), KFAC even scales to big data sets such as ImageNet (Krizhevsky et al., 2012) with large DNNs. Note that $A_\mathrm{i} \in \mathbb{R}^{n_\mathrm{i} \times n_\mathrm{i}}$ and $G_\mathrm{i} \in \mathbb{R}^{m_\mathrm{i} \times m_\mathrm{i}}$, where the number of weights per layer is $N_\mathrm{i} = n_\mathrm{i} m_\mathrm{i}$. Thus, the complexity of IM is reduced significantly. Performing the LA based on equation 2 we can model the weight posterior per layer as a matrix normal distribution (proposed by Ritter et al. (2018a)). See definitions in section A.

$$p(\theta_\mathrm{i} \mid x, y) \sim \mathcal{N}(vec(W_{\mathrm{i},MAP}), A_{\mathrm{i}-1}^{-1} \otimes G_\mathrm{i}^{-1}) = \mathcal{MN}(W_{\mathrm{i},MAP}, A_{\mathrm{i}-1}^{-1}, G_\mathrm{i}^{-1}). \tag{3}$$

Typically, IM is scaled by the number of data points N and incorporates the Gaussian prior τ.

$$\mathrm{N}\boldsymbol{I}_\mathrm{i} + \tau I \approx \left(\sqrt{\mathrm{N}}A_\mathrm{i} + \sqrt{\tau}I\right) \otimes \left(\sqrt{\mathrm{N}}G_\mathrm{i} + \sqrt{\tau}I\right). \tag{4}$$

LA for DNNs is practical, because the IM is computed after training. Moreover, its symmetry approximations (Bishop, 2016) may not be severe, as DNNs have a high dimension, and no methods can accurately estimate in every directions. Yet, the approximation in equation 2 has severe limitations. For example, it assumes that the IM can be Kronecker decomposed. Equation 4 induces also approximation errors. To mitigate this, we propose an alternative approach, as shown next.

3 Methodology

3.1 Laplace Approximation with a diagonal correction

Following George et al. (2018), we first propose to use an eigenvalue correction in the Kronecker factored eigenbasis for LA. For simplify, we drop layer indices i and explanation herein applies

[1] The expectation herein is defined wrt. the paramerterized density $p_\theta(y|x)$ assuming i.i.d. samples x.

layer-wise. Let $\boldsymbol{I} = V\Lambda V^T$ be the eigendecomposition of IM per layer. From this it follows $\Lambda = \mathbb{E}\left[V^T \delta\theta \delta\theta^T V\right]$. Defining elements of layer wise matrices as $i \in \{1, 2, \cdots, N\}$,

$$\Lambda_{ii} = \mathbb{E}\left[(V^T \delta\theta)_i^2\right]. \tag{5}$$

Further, define the eigendecomposition of A and G in equation 2 as $A = U_A S_A U_A^T$ and $G = U_G S_G U_G^T$. Then, from the properties of the Kronecker product it follows

$$\boldsymbol{I}_{\text{kfac}} \approx A \otimes G = (U_A \otimes U_G)(S_A \otimes S_G)(U_A \otimes U_G)^T. \tag{6}$$

This approximation can be improved by replacing $(S_A \otimes S_G)$ with the eigenvalues Λ from equation 5, where V is set to $(U_A \otimes U_G)$. We denote it as EFB:

$$\boldsymbol{I}_{\text{efb}} = (U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T \text{ and } \boldsymbol{I}_{\text{efb}}^{-1} = (U_A \otimes U_G)\Lambda^{-1}(U_A \otimes U_G)^T. \tag{7}$$

This eigenvalue correction has many desirable properties. First, it holds $\|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F \leq \|\boldsymbol{I} - \boldsymbol{I}_{\text{kfac}}\|_F$ wrt. the Frobenius norm. Second, multiplying diagonal matrices by Kronecker products already impose less degrees of freedom, which makes the computation more accurate by correcting the diagonal in the eigenbasis. Lastly, due to the orthogonality of eigenvectors and their Kronecker products, the inverse computation only involves the diagonal matrix Λ. Thus, the resulting weight posterior using EFB can be expressed with eigenvalue-corrected matrix variate distribution (refer to section A) as shown below.

$$p(\theta \mid x, y) \sim \mathcal{N}(vec(W_{MAP}), \boldsymbol{I}_{\text{efb}}^{-1}) = \mathcal{EMN}(W_{MAP}, U_A, \Lambda^{-1}, U_G) \tag{8}$$

The regularization terms can again be incorporated, yet without inducing additional approximation using $\tau I = (U_A \otimes U_B)\tau I(U_A \otimes U_G)^T$ and exploring the orthogonality of $(U_A \otimes U_G)$:

$$\mathrm{N}\boldsymbol{I}_{\text{efb}} + \tau \boldsymbol{I} = \mathrm{N}(U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T + \tau \boldsymbol{I} = (U_A \otimes U_G)(\mathrm{N}\Lambda + \tau I)(U_A \otimes U_G)^T. \tag{9}$$

However, there is an approximation in EFB since $(U_A \otimes U_G)$ is still an approximation of the true eigenbasis V. Therefore, we propose a further improvement of the approximation by correcting the diagonal entries of the EFB in parameter space, as shown next.

Intuitively, EFB only performs a correction of the diagonal elements in the eigenbasis, but when mapping back to the parameter space this correction is again harmed by the inexact estimate of the eigenvectors. Of course, an exact estimation of the eigenvectors is infeasible, but it is important to note that the diagonals of the exact IM $\boldsymbol{I}_{ii} = \mathbb{E}\left[\delta\theta_i^2\right]$ can be computed efficiently using back-propagation. This and the fact that the off-diagonal elements of the IM are weaker with larger data sets or when using weight normalization (Neyshabur et al. (2015); Desjardins et al. (2015); Salimans & Kingma (2016)), motivates the idea to correct the approximation further as follows:

$$\begin{aligned} \boldsymbol{I}_{\text{def}} &= (U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T + D \text{ where} \\ D_{ii} &= \mathbb{E}\left[\delta\theta_i^2\right] - \sum_{k=1}^{nm} (v_{\alpha,\alpha}\sqrt{\Lambda_k})^2 \end{aligned} \tag{10}$$

Note that similar ideas of adding a diagonal correcting term are used for sparse Gaussian Processes (GPs). One prominent example is the work of Snelson & Ghahramani (2006), which also introduces an additional diagonal correction term based on the idea of so-called pseudo-input points. In our work, we correct the variance in the information form of the posterior Normal distribution, such that our approximation is at least exact on its diagonals of IM.

In equation equation 10, we have represented $(U_A \otimes U_G)\Lambda(U_A \otimes U_G)_{ii}^T$ as $\sum_{k=1}^{nm}(v_{\alpha,\alpha}\sqrt{\Lambda_k})^2$ where $V = (U_A \otimes U_G) \in \mathbb{R}^{mn \times mn}$ is a Kronecker product with diagonal elements $v_{\alpha,\alpha}$ (see definition 1 below). It follows from the properties of the Kronecker product that $\alpha = m(i-1) + k$. The derivation is shown

in section B. Note that in this given form, the Kronecker products are never directly evaluated but the diagonal matrix D can be computed recursively, making it computationally feasible.

Definition 1: *Given matrix $A \in \mathbb{R}^{m\times a}$ and $B \in \mathbb{R}^{n\times b}$, the Kronecker product of $V = A \otimes B \in \mathbb{R}^{mn\times ab}$ is given by $v_{\alpha,\beta} = a_{ij}b_{kl}$, where the indices $\alpha = n(i-1)+k$ and $\beta = b(j-1)+l$. Here, the indices of the matrices A and B are $i \in \{1,\cdots,m\}$, $j \in \{1,\cdots,a\}$, $k \in \{1,\cdots,n\}$ and $l \in \{1,\cdots,b\}$.*

Now, our LA models the weight posterior distribution in an information form of MND as shown in equation 11. With this formulation one can store its covariance with matrices of smaller sizes. IV denote here information vector (formulation is in section A).

$$p(\theta \mid x, y) \sim \mathcal{N}(vec(W_{MAP}), \boldsymbol{I}_{\text{def}}^{-1}) = \mathcal{IN}(W_{MAP}^{IV}, (U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T + D) \tag{11}$$

Now, some of the interesting theoretical properties are as follows with proofs provided in section C.

Lemma 1: *Let $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ be the real Fisher information matrix, and let $\boldsymbol{I}_{def} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{efb} \in \mathbb{R}^{N\times N}$ be the DEF and EFB estimates of it respectively. It is guaranteed to have $\left\|\boldsymbol{I} - \boldsymbol{I}_{efb}\right\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{def}\right\|_F$.*

Corollary 1: *Let $\boldsymbol{I}_{kfac} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{def} \in \mathbb{R}^{N\times N}$ be KFAC and our estimates of real Fisher information matrix $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ respectively. Then, it is guaranteed to have $\left\|\boldsymbol{I} - \boldsymbol{I}_{kfac}\right\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{def}\right\|_F$.*

We note that $\|\boldsymbol{I} - \boldsymbol{I}_{\text{kfac}}\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F$ may not mean $\left\|\boldsymbol{I}^{-1} - \boldsymbol{I}_{\text{kfac}}^{-1}\right\|_F \geq \left\|\boldsymbol{I}^{-1} - \boldsymbol{I}_{\text{efb}}^{-1}\right\|_F$ and vice versa. Yet, we can prove that our LA yield better estimates in information form of the posterior distribution than that of Ritter et al. (2018a) in terms of Frobenius norm.

Unfortunately, in the current form, it involves a matrix inversion with size N by N when sampling. For some layers in modern architectures, this is not be feasible. This problem is tackled next.

3.2 Low Rank Form of Kronecker Factored Eigenvalue Decomposition for sampling

Sampling from the posterior is crucial in Bayesian Neural Networks. For example, an important use-case of the parameter posterior is estimating the predict uncertainty for test data (x^*,y^*) by full Bayesian analysis with K_{mc} samples (equation 12).

$$p(y^*|x^*, x, y) = \int p(y^*|x^*, \theta)p(\theta|x, y)\delta\theta \approx \frac{1}{K_{mc}}\sum_{t=1}^{K_{mc}} y^*(x^*, \theta_t^s) \;\text{ for } \theta^s \sim \mathcal{N}(W_{MAP}, \boldsymbol{I}_{\text{def}}^{-1}) \tag{12}$$

However, directly sampling from equation 11 is non-trivial. This is because the size of the covariance is prohibitively too large as we use MND. Consequently, its sparse form is introduced next.

As a first step, we propose the low rank form in equation 13. Here, $\Lambda_{1:L} \in \mathbb{R}^{L\times L}$, $U_{A_{1:a}} \in \mathbb{R}^{m\times a}$ and $U_{n_{1:g}} \in \mathbb{R}^{n\times g}$ denote low rank form of corresponding eigenvalues and vectors. Naturally, it follows from definition that $L = ag$, $N = mn$ and furthermore, the persevered rank L corresponds to preserving top K and additional J eigenvalues (resulting in $L \geq K$, $L = ag = K + J$).

$$\boldsymbol{I}_{\text{def}} \approx \hat{\boldsymbol{I}}_{\text{def}} = (U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D \tag{13}$$

To explain, we highlight differences to Bishop (2016) in which top L eigenvalues and corresponding eigenvectors are preserved for LRA. In our case however, this results in intractable $(U_A \otimes U_G)_{1:L}$ which defies the purpose. Therefore, as seen in equation 13, the Kronecker structure in eigenvectors as $(U_{A_{1:a}} \otimes U_{G_{1:g}})$ is preserved. Consequently, due to the Kronecker product operation, preserving top K eigenvalues results in $L = K + J$ eigenvalues in total.

For example, let matrix E decomposed as $E = U_{1:6}\Lambda_{1:6}U_{1:6}^T \in \mathbb{R}^{6\times 6}$ with $U_{1:6} = \begin{bmatrix} u_1 & u_2 & \cdots & u_6 \end{bmatrix} \in \mathbb{R}^{6\times 6}$ and $\Lambda_{1:6} = \text{diag}(\Lambda_1, \Lambda_2, \cdots, \Lambda_6) \in \mathbb{R}^{6\times 6}$ in a descending order. In this toy example, the LRA with top 3 eigenvalues result in $E_{1:3} = U_{1:3}\Lambda_{1:3}U_{1:3}^T \in \mathbb{R}^{6\times 6}$ (see notation to above). Instead, consider now the matrix $E_{\text{kron}} = (U_{A_{1:3}} \otimes U_{G_{1:2}})\Lambda_{1:6}(U_{A_{1:3}} \otimes U_{G_{1:2}})^T \in \mathbb{R}^{6\times 6}$. Again, say we want to preserve top 3 the eigenvalues $\Lambda_{1:3}$ and corresponding eigenvectors $(U_{A_{1:3}} \otimes U_{G_{1:2}})_{1:3}$, However, as $(U_{A_{1:a}} \otimes U_{G_{1:g}})_{1:3} = \begin{bmatrix} u_{A_1} \otimes u_{G_1} & u_{A_1} \otimes u_{G_2} & u_{A_2} \otimes u_{G_1} \end{bmatrix}$, preserving the eigenvectors with the Kronecker structure results

Algorithm 1: Sparsification

Input: Marices U_A, U_G, Λ and K.
Output: Matrices: $U_{A_{1:N}}$, $U_{G_{1:M}}$, $\Lambda_{1:L}$.
Algorithm:
1. Find indices of top K eigenvalues on Λ. This results in $\alpha \in \{\alpha_1, \alpha_2, \cdots, \alpha_K\}$.
2. For each elements of α, find corresponding indices of each Kronecker factors of original matrix $S_A \otimes S_G$ (equation 6) by using definition 2: $\underline{i} = int(\frac{\alpha}{m}) + 1$ and $\underline{k} = \alpha - m(\underline{i} - 1)$. This results in indices of $\underline{i}$ and $\underline{k}$ for U_A and U_G corresponding to $S_A \otimes S_G$ or $\Lambda_{1:K} = \Lambda_\alpha$.
3. Using obtained indices $\underline{i}$ and $\underline{k}$, compute eigenvectors $U_{A_{1:a}} = U_{A\underline{i}}$ and $U_{G_{1:g}} = U_{G\underline{k}}$ which are the preserved eigenvectors in equation 13.
4. Using definition 1, find indices of top K and additional J eigenvalues using $\alpha_j = m(i-1) + k$ for all $\underline{i}$ and $\underline{k}$.
5. Preserve eigenvalues $\Lambda_{1:L} = \Lambda_{\alpha_j}$ where $\alpha_j \subseteq \alpha$. $\Lambda_{1:L}$ represents the preserved top K and additional J eigenvalues in equation 13.

Algorithm 2: Inference of IM

Input: Pre-trained Neural Network and train data.
Output: Matrices: $U_{A_{1:N}}$, $U_{G_{1:M}}$, $\Lambda_{1:L}$ and D.
Algorithm:
for *the given data points* **do** *KFAC*
  for $i := 1$ *to* l **do**
    Compute A_i and G_i (equation 2).
  end
end
Compute U_A, U_G with eigenvalue decomposition.
for *the given data points* **do** *EFB*
  for $i := 1$ *to* l **do**
    Compute $\mathbb{E}\left[\delta\theta_i^2\right]$.
    Compute Λ_i (equation 5) with U_A and U_G.
  end
end
for *all the layers* **do** *DEF*
  Compute $U_{A_{1:N}}$, $U_{G_{1:M}}$, $\Lambda_{1:L}$ (algorithm 1).
  Compute D (equation 10).
end

in having to store $U_{A_{1:2}} = \begin{bmatrix} u_{A_1} & u_{A_2} \end{bmatrix}$ and $U_{G_{1:2}} = \begin{bmatrix} u_{G_1} & u_{G_2} \end{bmatrix}$. Consequently, additional eigenvalue Λ_4 has to be saved in order to fulfill the definition of a Kronecker product $E_{kron_{1:3}} = (U_{A_{1:2}} \otimes U_{G_{1:2}})\Lambda_{1:4}(U_{A_{1:2}} \otimes U_{G_{1:2}})^T \in \mathbb{R}^{6\times 6}$. In summary, preserving top K eigenvalues results in other J eigenvalues, which ensures the memory-wise tractability in case of DNNs. Furthermore, in contrast to this toy example, finding the indices of J eigenvalues require formalism as we deal with matrices of higher dimension. This motivates for an algorithm that enables the LRA of Kronecker factored eigendecomposition, which is described below.

For this computation we introduce algorithm 1. Let us start with a definition on indexing rules of Kronecker factored diagonal matrices, which is a core of algorithm 1.

Definition 2: *Given a diagonal matrix $S_A \in \mathbb{R}^{m\times m}$ and $S_G \in \mathbb{R}^{n\times n}$, the Kronecker product of $\Lambda = S_A \otimes S_B \in \mathbb{R}^{mn\times mn}$ is given by $\Lambda_\alpha = s_{a\underline{i}} s_{b\underline{k}}$, where the indices $\alpha = n(\underline{i}-1)+\underline{k}$ with $\underline{i} \in \{1, 2, \cdots, m\}$ and $\underline{k} \in \{1, 2, \cdots, n\}$. Then, given α and m, $\underline{i} = int(\frac{\alpha}{m}) + 1$ and given α, m, and $\underline{i}$, $\underline{k} = \alpha - m(\underline{i} - 1)$. Here, $int(\cdot)$ is an operator that maps its input to lower number integer.*

For explaining algorithm 1, the toy example can be revisited. Firstly, as we preserve top 3 eigenvalues, $\alpha \in \{1, 2, 3\}$ which are indices of eigenvalues $\Lambda_{1:3}$ (line 1). Then, using line 2, $\underline{i} \in \{1, 2\}$ and $\underline{k} \in \{1, 2\}$ can be computed using definition 2. This relation holds as Λ is computed from $S_A \otimes S_G$ in equation 6, and thus, U_A and U_G are their corresponding eigenvectors respectively. In line 3, we keep $U_{A_{1:2}}$ and $U_{G_{1:2}}$) using $\underline{i}$ and $\underline{k}$. Again, in order to fulfill the Kronecker product operation, we use line 4 to find the eigenvalues $\alpha_j \in \{1, 2, 3, 4\}$, and then preserve $\Lambda_{1:4}$. As explained, this has resulted in saving top 3 and additional 1 eigenvalues. Algorithm 1 provides the generalization of this and even if eigendecomposition does not come with a descending order, the same logic applies.

Having introduced the essentials, we depict an overview of our inference scheme in algorithm 2. As IM is estimated after the training, our method can be applied to existing and already-working architectures. Moreover, EFB is computed in a different way to George et al. (2018) so that our EFB does not require batch assumption for taking expectations. Moreover, the scheme is not as expensive since eigenvalue decomposition of A_i and G_i are computed only once.

A benefit of the LRA is that now, sampling from the given covariance (equation 11 with the low rank form in equation 13) only involves the inversion of a $L \times L$ matrix (in offline settings) and matrix multiplications of smaller Kronecker factored matrices or diagonal matrices during full Bayesian analysis. To this end, we derive the analytical form of the sampler in section B which makes the sampling memory-wise feasible by exploring the Kronecker structure of $\boldsymbol{I}_{\text{def}}$. The incorporation of prior or regularization terms also follows without any additional approximation.

$$\mathrm{N}\hat{\boldsymbol{I}}_{\text{def}} + \tau I_D = (U_{A_{1:a}} \otimes U_{G_{1:g}})(\mathrm{N}\Lambda_{1:L})(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + (\mathrm{N}D + \tau I) \tag{14}$$

To our knowledge, the proposed sparse IM have not been studied before. Therefore, we theoretically motivate its design and validity for better insights. The analysis can be found below.

Lemma 2: *Let $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ be the real Fisher information matrix, and let $\boldsymbol{I}^{top}_{1:K} \in \mathbb{R}^{N\times N}$, $\boldsymbol{I}^{top}_{1:L} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{1:L} \in \mathbb{R}^{N\times N}$ be the low rank estimates of $\boldsymbol{I}$ of EKB obtained by preserving top K, L and top K plus additional J resulting in L eigenvalues. Here, we define $K < L$. Then, the approximation error of $\boldsymbol{I}_{1:L}$ is bounded as follows: $\left\|\boldsymbol{I} - \boldsymbol{I}^{top}_{1:K}\right\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{1:L}\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}^{top}_{1:L}\right\|_F$.*

Lemma 2 indicate that the given low rank form is rather conservative but a memory efficient alternative if saving top L eigenvalues leads to intractable computation.

Lemma 3: *The low rank matrix $\hat{\Sigma} = \left((U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D\right)^{-1} \in \mathbb{R}^{N\times N}$ is a non-degenerate covariance matrix if the diagonal correction matrix D and LRA $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T$ are both symmetric and positive definite. This condition is satisfied if $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T_{ii} < \mathbb{E}\left[\delta\theta_i^2\right]$ for all $i \in \{1, 2, \cdots, d\}$ and with $\Lambda_{1:L} \nsubseteq 0$.*

This Lemma comments on validity of resulting parameter posterior and proves that sparsifying the matrix can lead to a valid non-degenerate covariance if two conditions are met. As non-degenerate covariance can have a uniquely defined inverse, it is important to check these two conditions.

Lemma 4: *Let $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ be the real Fisher information matrix, and let $\hat{\boldsymbol{I}}_{def} \in \mathbb{R}^{N\times N}$, $\boldsymbol{I}_{efb} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{kfac} \in \mathbb{R}^{N\times N}$ be the low rank DEF, EFB and KFAC estimates of it respectively. Then, it is guaranteed to have $\left\|\boldsymbol{I}_{ii} - \boldsymbol{I}_{efb_{ii}}\right\|_F \geq \left\|\boldsymbol{I}_{ii} - \hat{\boldsymbol{I}}_{def_{ii}}\right\|_F = 0$ and $\left\|\boldsymbol{I}_{ii} - \boldsymbol{I}_{kfac_{ii}}\right\|_F \geq \left\|\boldsymbol{I}_{ii} - \hat{\boldsymbol{I}}_{def_{ii}}\right\|_F = 0$. Furthermore, if the eigenvalues of $\hat{\boldsymbol{I}}_{def}$ contains all non-zero eigenvalues of $\boldsymbol{I}_{def}$, it follows: $\left\|\boldsymbol{I} - \boldsymbol{I}_{efb}\right\|_F \geq \left\|\boldsymbol{I} - \hat{\boldsymbol{I}}_{def}\right\|_F$.*

Lastly theoretical property shows the optimally of the given covariance form in capturing the diagonal variance while indicating that our approach also becomes effective in estimating off-diagonal entries if IM contains many close to zero eigenvalues. Validity of this assumption has been studied by Sagun et al. (2018). Intuitively, from a graphical interpretation of IM, diagonal entries indicate information present in each nodes (parameters) and off-diagonal entries are links of these nodes. The given sparsification scheme reduces the strength of the weak links (their numerical values) while keeping the diagonal variance accurate. This is by the design of the inference procedure (depicted in algorithm 2), in which, we apply the diagonal correction after the LRA.

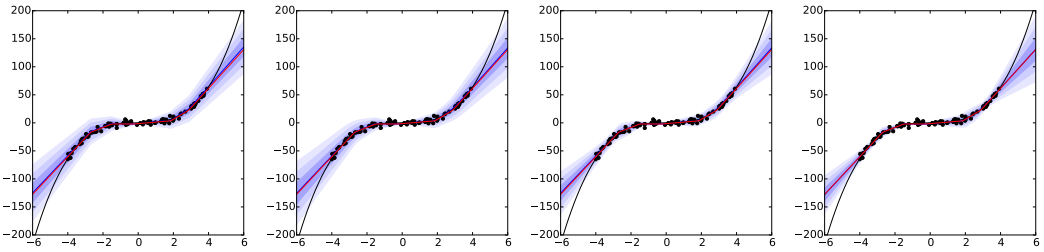
4 Experimental Results

An empirical study is presented with a toy regression, MNIST (Lecun et al., 1998) and CIFAR10 (Krizhevsky, 2009) data-sets. All experiments are implemented using Tensorflow (Abadi et al., 2016).

4.1 Predictive Uncertainty Estimation

Firstly, an evaluation on toy regression data-set is presented. This exercise has an advantage that we can not only evaluate the predictive uncertainty, but also directly compare various approximations to the Hessian (presented in section 4.2). For this a single-layered fully connected network with seven units in the first layer is considered. We have used 100 uniformly distributed points $x \sim U(-4, 4)$ and samples $y \sim N(x^3, 3^2)$. Visualization of predictive uncertainty is shown in figure 1. Ideally, predictive uncertainty should be high in the regimes where there is fewer or no training data. Both Diag and KFAC Laplace are tuned similar to Ritter et al. (2018a) by regularizing them with hyperparameters. FB Laplace denotes the use of exact block diagonal Hessian. Ours and FB Laplace for this experiment did not require any hyper-parameter tuning. All the methods show high uncertainty in the regimes far away from training data and seem to deliver reliable predictive uncertainty. Yet, Diag and KFAC Laplace predicts rather high uncertainty even within the regions that are covered by the training data. Diag and KFAC Laplace behaves similar to each other whereas ours slightly overestimate the uncertainty but produces the most comparable fit to the Full Laplace. We believe this is the direct effect of modelling the Hessian more accurately [2].

[2] we comment on this statement, and the effects of data-set size to number of parameters in section D.



Diag Laplace KFAC Laplace FB Laplace Ours (DEF)

Figure 1: Uncertainty on toy regression. The black dots and the black lines are data points (x, y). The red and blue lines show predictions of the deterministic Neural Network and the mean output respectively. Upto three standard deviations are shown with blue shades.

Next, we evaluate predictive uncertainty on classification tasks in which the proposed low-rank representation is strictly necessary. Furthermore, our goal is not to achieve the highest accuracy but evaluate predictive uncertainty. To this end, we choose classification tasks with known and unknown classes, e.g. a network is not only trained and evaluated on MNIST but also tested using notMNIST. Note that under such tests, any probabilistic methods should report their evaluations on both known and unknown classes with the same hyperparameter settings. This is because one can make a neural network to be always highly uncertain, which may seem to work well on out-of-distribution samples but are always overestimating uncertainty, even for the correctly classified samples within the distribution similar to the train data. For evaluating predictive uncertainty on known classes, Expectation Calibration Error (ECE) has been used. As we found it more intuitive, normalized entropy is reported for evaluating predictive uncertainty on unknown classes.

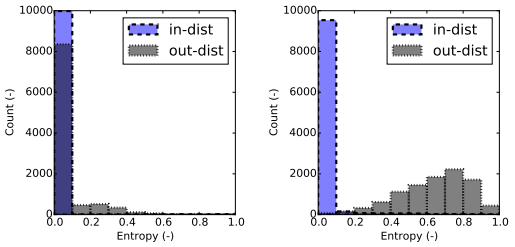


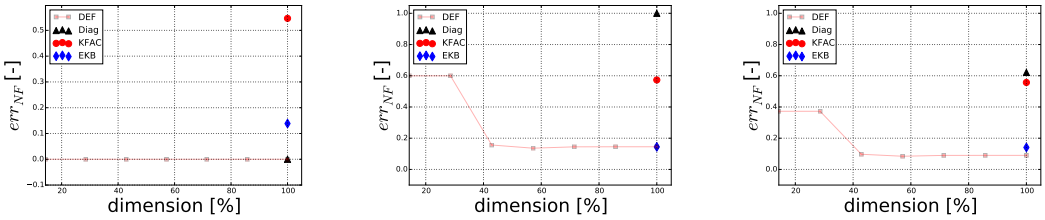
Figure 2: Normalized Entropy histogram (left: deterministic and right: ours) on MNIST vs notMNIST experiments.

On MNIST dataset, we compare our method to dropout (Gal, 2016), deep ensemble (Lakshminarayanan et al., 2017) with size 15, diagonal and KFAC Laplace (Ritter et al., 2018a). These methods have merits that no changes in the training procedure is required (in contrast to variational methods). This is crucial for a fair comparison as we can use the same experiment settings (Mukhoti et al., 2018). EFB Laplace is introduced as an ablation study. Regarding the architectures, LeNet with RELU activations and a L2 coefficient of 1e-8 has been the choice of architecture. In particular, this typically makes a neural network overconfident, and we can see the effects of model uncertainty. This architecture validates our claim as it has the parameters of size $\theta_3 \in \mathbb{R}^{3137\times1024}$ in the 3^{rd} layer. Obviously, its covariance is intractable as it is quadratic in size. The results can be found in table

Table 1: MNIST-notMNIST experiments. Accuracy and ECE are evaluated on MNIST. Entropy is evaluated on notMNIST. Lower the better for ECE. Higher the better for entropy.

	NN	**Diag**	**KFAC**	**Dropout**	**Ensemble**	**EFB**	**DEF**
Accuracy	0.993	0.9935	0.9929	0.9929	**0.9937**	0.9929	0.9927
ECE	0.395	0.0075	0.0078	0.0105	0.0635	0.012	**0.0069**
Entropy	0.055±0.133	0.555 ± 0.196	0.599 ± 0.199	0.562 ± 0.19	0.596 ± 0.133	0.618 ± 0.185	**0.635 ± 0.1904**

1. Firstly all the methods improved significantly over the deterministic one. They showed similar performance on calibration except the deep ensemble. On notMNIST experiments, Diag Laplace and dropout made the least uncertain predictions while other methods performed quite similarly. Lastly, within these experiments, DEF Laplace achieved the lowest ECE, at the same time, predicted with the highest mean entropy on out-of-distribution samples. Figure 2 shows this result. There are fewer



(a) Diagonal error (b) Off-diagonal error (c) Overall error

Figure 3: Effects of LRA in Frobenius norm of error normalized from 0 to 1. Lower the better. Laplace based methods such as EFB, Diag, KFAC and DEF are compared in terms of diagonal, off-diagonal and overall resulting approximation error to exact block diagonal hessian. Effects of LRA with DEF is shown by lowering down the dimensions. This figure shows the working principle of our method that diagonal variance of information matrix is accurately captured, while the off-diagonal approximation error largely depend on the reduced rank, and how its eigenvalues are distributed.

overconfident predictions on the data from unknown classes (out-dist) while being confident on the correctly classified samples from the known class (in-dist). To ensure a fair comparison, we have performed an extensive grid search for the parameters of Laplace based methods (see section D).

Further tests were performed on CIFAR10 (known) and SVHN (unknown). We trained a 5 layer architecture with 2 convolutional and 3 fully connected layers with batch normalization and data augmentation. The results are reported in table 2. Similar to MNIST experiments, our method resulted in a better calibration performance and out-of-distribution detection overall. Note that for Diag, KFAC and EFB Laplace, grid searches on hyperparameters were rather non-trivial here. Increasing τI had the tendency to reduce ECE on CIFAR10, but in return resulted in underestimating the uncertainty on SVHN and vice versa. DEF Laplace instead, required smallest regularization hyperparameters to strike a good balance between these two objectives. We omitted dropout as using it as a stochastic regularization instead of batch normalization would result in a different network and thus, comparison would be not meaningful. Implementation details and further results are presented in section D.

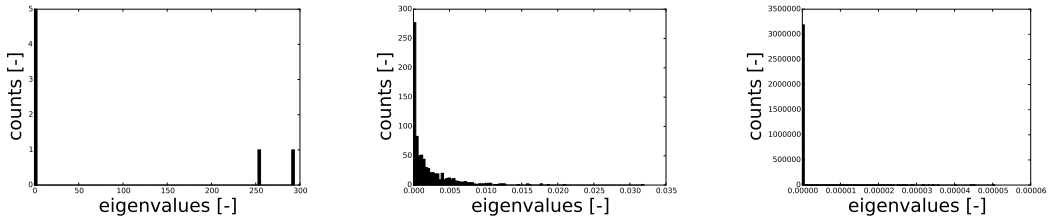
Table 2: CIFAR10-SHVN experiments. Accuracy and ECE evaluated on CIFAR10. Entropy evaluated on SHVN. Lower the better for ECE. Higher the better for entropy.

	NN	**Diag**	**KFAC**	**Ensemble**	**EFB**	**DEF**
Accuracy	0.8606	**0.8659**	0.8572	0.8651	0.8638	0.8646
ECE	0.0819	0.0358	0.0351	0.0809	0.0343	**0.0084**
Entropy	0.245 ± 0.215	0.4129 ± 0.197	0.408 ± 0.197	0.370 ± 0.192	0.417 ± 0.196	**0.4338 ± 0.186**

Importantly, we demonstrate that when projected to different success criteria, no inference methods largely win uniformly. Yet these experiments also show empirical evidence that our method works in principle and compares well to the state-of-the-art. Representing layer-wise MND in a sparse information form, and demonstrating a low rank inverse/sampling computations, we show an alternative approach of designing scalable and practical inference framework. Finally, these results also indicate that keeping the diagonals of IM accurate while sparsifying the off-diagonals can lead to outstanding performance in terms of predictive uncertainty and generalizes well to various data, models and even measures. For future works, we share the view that comparing different approximations to the true posterior is quite challenging for DNNs. Consequently, better metrics and benchmarks that show-case the benefits of model uncertainty can be an important direction of future research.

4.2 Effects of Low Rank Approximation

Next, we quantitatively study the effects of LRA by directly evaluating on the approximations of IM. This is because uncertainty estimation, despite being a crucial entity, are confounded from the problem itself and may not reveal the algorithmic insights to its full potential. For this, we revisit the



(a) Toy dataset layer 1 (b) LeNet layer 1 (c) LeNet layer 3

Figure 4: Eigenvalue histograms. Its values and counts are plotted as histograms. In particular, part (a) depicts the eigenvalue histogram from the analysis of toy experiments (figure 3). On the other hand, part (b) and (c) are two extreme cases from the classification experiments. This figure indicate that our assumption

toy regression problem and provide a direct evaluation of IM with measure on normalized Frobenius norm of error err_{NF} in the first layer of the network.

The results are shown in figure 3. Here, the reduced dimension is not proportional to the ranks (e.g. many zero or close to eigenvalues as shown in figure 4 (a)). Figure 3 (a) depicts that DEF results in accurate estimates on $\boldsymbol{I}_{ii}$ regardless of the chosen dimensions L while EFB has the more approximation error, which we believe is due to inaccurate estimates of eigenvectors. KFAC on the other hand, produces the most errors on diagonal elements, which indicate that its assumption of Kronecker factorization induces crude approximation in this experiment. Regarding the off-diagonal errors EFB also outperforms KFAC and Diag estimates. Furthermore, error profile of off-diagonal error $\boldsymbol{I}_{ij}$ also explains the principles of the LRA that as we decrease the ranks, the error increases but in preserving manner (similar to factor analysis Bishop (2016)). This reflects our theoretical analysis that it depends on how eigenvalues are distributed (depicted in figure 4 (a)).

For the classification experiments, two other extremes in eigenvalue histograms are also reported in figure 4 (b) and (c). As shown, figure 4 (b), the layer 1 of LeNet contains less close to zero eigenvalues when compared to the layer 3. This is also in line with experimental findings of Sagun et al. (2018) that when the network is over-parameterized, its Hessian tend to have many close to zero eigenvalues. From this insight, we deduce that within our experiments, the LRA is an effective tool for the Hessian of over-parameterized layer since our algorithmic assumption reasonably holds well.

These analysis reveal the methodological design principles which exploits the insights on loss landscape of neural networks, we show-case that our representation of model uncertainty is accurate in capturing the information of the parameters while removing weak correlations between the parameters. However, this also brings a limitation of our approach, which is, its dependency on how DNNs are trained, and the loss-landscape is structured. Consequently, an interesting direction of future research can be studies on how loss-landscape, approximate inference and true posterior can be together exploited to tackle the challenges in deep Bayesian Neural Networks. During our study, we found information form of Normal distribution more intuitive to understand as they tend to be rather sparse with its probabilistic interpretations (Thrun et al., 2004).

5 Conclusion

In this paper, we proposed a method to estimate model uncertainty of deep neural networks in information form of Multivariate Normal Distribution. As a first step, a practical approximate inference procedure has been imposed with Laplace Approximation, which, involves two steps namely (1) re-scale the information matrix in its eigen-basis and (2) correct the diagonals in parameter space. In order to keep the computation tractable, we have further proposed to use low rank approximation while maintaining the Kronecker structure. With the devised algorithms, we have demonstrated a memory efficient sampling computations, and a method to preserve top eigenvalues with Kronecker factored eigendecomposed matrices. Our theoretical and empirical evaluation confirms that our method compares well to the current state-of-the art, not only in terms of reliability of uncertainty estimates, but also in terms of scalability to larger networks and data.

REFERENCES

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.

Sivaram Ambikasaran and Michael O'Neil. Fast symmetric factorization of hierarchical matrices with applications. *CoRR*, abs/1405.0223, 2014.

Jimmy Ba, Roger Grosse, and James Martens. Distributed second-order optimization using kronecker-factored approximations. In *ICLR*, 2017.

Juhan Bae, Guodong Zhang, and Roger Grosse. Eigenvalue corrected noisy natural gradient. *CoRR*, abs/1811.12565, 2018.

S. Becker and Yann Lecun. Improving the convergence of back-propagation learning with second-order methods. In D. Touretzky, G. Hinton, and T. Sejnowski (eds.), *Proceedings of the 1988 Connectionist Models Summer School, San Mateo*, pp. 29–37. Morgan Kaufmann, 1989.

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2016.

Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical gauss-newton optimisation for deep learning. In *ICML*, 2017.

Dong C. Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 08 1989.

Sheng-Wei Chen, Chun-Nan Chou, and Edward Y. Chang. Bda-pch: Block-diagonal approximation of positive-curvature hessian for training neural networks. *CoRR*, abs/1802.06502, 2018.

Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, and koray kavukcuoglu. Natural Neural Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2071–2079. Curran Associates, Inc., 2015.

Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a kronecker factored eigenbasis. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 9573–9583, 2018.

Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2348–2356. Curran Associates, Inc., 2011.

Roger B. Grosse and James Martens. A kronecker-factored approximate fisher matrix for convolution layers. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 573–582, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Weinberger. On calibration of modern neural networks. In *Proc of the 34th International Conference on Machine Learning (ICML)*, 2017.

A.K. Gupta and D.K. Nagar. *Matrix Variate Distributions*. Monographs and Surveys in Pure and Applied Mathematics. Taylor & Francis, 1999. ISBN 9781584880462.

Philipp Hennig. Fast probabilistic optimization from noisy gradients. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28-1 of *Proceedings of Machine Learning Research*, pp. 62–70, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

Jose Miguel Herandez-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1861–1869. JMLR.org, 2015.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pp. 5–13, New York, NY, USA, 1993. ACM. ISBN 0-89791-611-5. doi: 10.1145/168304.168306.

Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2616–2625. PMLR, 2018.

Durk P Kingma, Tim Salimans, and Max Welling. Variational Dropout and the Local Reparameterization Trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2575–2583. Curran Associates, Inc., 2015.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114 13:3521–3526, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Canadian Institute for Advanced Research, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6405–6416, 2017.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1708–1716, New York, New York, USA, 20–22 Jun 2016. PMLR.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2218–2227, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

James Martens and Roger B. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 2408–2417, 2015.

Aaron Mishkin, Frederik Kunstner, Didrik Nielsen, Mark W. Schmidt, and Mohammad Emtiyaz Khan. SLANG: fast structured covariance approximations for bayesian deep learning with natural gradient. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montreal, Canada.*, pp. 6248–6258, 2018.

Jishnu Mukhoti, Pontus Stenetorp, and Yarin Gal. On the importance of strong baselines in bayesian deep learning. *CoRR*, abs/1811.09385, 2018.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, Berlin, Heidelberg, 1996. ISBN 0387947248.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-sgd: Path-normalized optimization in deep neural networks. *CoRR*, abs/1506.02617, 2015.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018a.

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 3742–3752, 2018b.

Nicolas Le Roux and Andrew W. Fitzgibbon. A fast natural newton method. In *ICML*, 2010.

Levent Sagun, Utku Evci, V. Ugur Güney, Yann Dauphin, and Léon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*, 2018.

Tim Salimans and Durk P Kingma. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 901–909. Curran Associates, Inc., 2016.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt (eds.), *Advances in Neural Information Processing Systems 18*, pp. 1257–1264. MIT Press, 2006.

Shengyang Sun, Changyou Chen, and Lawrence Carin. Learning Structured Weight Uncertainty in Bayesian Neural Networks. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1283–1292, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 2004.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Guodong Zhang, Shengyang Sun, David K. Duvenaud, and Roger B. Grosse. Noisy natural gradient as variational inference. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 5847–5856, 2018.

A Notations on Distributions

A.1 Matrix Normal Distribution

The matrix normal distribution is a probability density function for the random variable $X \in \mathbb{R}^{n \times p}$ in matrix form. It can be parameterized with mean $\mathrm{M} \in \mathbb{R}^{n \times p}$, scales $\mathrm{U} \in \mathbb{R}^{n \times n}$ and $\mathrm{V} \in \mathbb{R}^{p \times p}$. It is essentially a multivariate Gaussian distribution with mean vec(M) and covariance $U \otimes V$.

$$p(X|M,U,V) = \frac{\exp\left(-\frac{1}{2}\text{tr}\left[V^{-1}(X-M)^T U^{-1}(X-M)\right]\right)}{2\pi^{\frac{np}{2}}|V|^{\frac{n}{2}}|U|^{\frac{p}{2}}} \tag{15}$$

We denote this distribution with $\mathcal{MN}$ parameterized by M, U and V. Refer to Gupta & Nagar (1999) for more details.

A.2 Eigenvalue-corrected Matrix Variate Distribution

The eigenvalue corrected matrix variate distribution (introduced by Bae et al. (2018)) is a probability density function for the random variable $X \in \mathbb{R}^{n\times p}$ in matrix form again. Difference to matrix normal distribution is that it extends its representation to Kronecker factored eigenbasis which is parameterized by $\text{M} \in \mathbb{R}^{n\times p}$, scales $\text{U} \in \mathbb{R}^{n\times n}$, $\text{V} \in \mathbb{R}^{p\times p}$ and $\text{R} \in \mathbb{R}^{np\times np}$.

$$p(X|M,U,V,R) = \frac{\exp\left(-\frac{1}{2}P^T R^{-1} P\right)}{2\pi^{\frac{np}{2}}|R|^{\frac{np}{2}}}, \tag{16}$$

where $P = vec(V_U^T(X-M)V_V)$. In the main text, this distribution is denoted with $\mathcal{EMN}$.

A.3 Information Form of Multivariate Normal Distribution

Information form of Multivariate Normal Distribution is a dual representation for the well known canonical form of Multivariate Normal Distribution (equation 17). In equation 17, $x \in \mathbb{R}^n$, $\mu \in \mathbb{R}^n$ and $\Sigma \in \mathbb{R}^{n\times n}$ represent a random variable, mean and covariance respectively.

$$p(x|\mu,\Sigma) \propto \exp\left\{-\frac{1}{2}x^T\Sigma^{-1}x + \mu^T\Sigma^{-1}\mu\right\} \tag{17}$$

Equation 18 shows the Information form of Multivariate Normal Distribution.

$$p(x|b,F) \propto \exp\left\{-\frac{1}{2}x^T F x + bx\right\} \tag{18}$$

Here, $x \in \mathbb{R}^n$ represent the random variable as well. $b \in \mathbb{R}^n = \Sigma^{-1}\mu$ and $F \in \mathbb{R}^{n\times n} = \Sigma^{-1}$ are information vector (denoted IV in the main text with superscript) and matrix respectively. As this formulation is completely described by an information vector and matrix, we use $\mathcal{IN}$.

Information matrix is also widely known as precision matrix. Note that the canonical form of Multivariate Normal Distribution can be fully parameterized with mean and covariance. Likewise, information vector and matrix fully parameterize the information form of Multivariate Normal Distribution. Thrun et al. (2004) in Simultaneous Localization and Mapping (SLAM) literature provides a good overview and explanations.

B Derivations

B.1 Derivation 1: Diagonal correction without evaluating the Kronecker products

Evaluating $A \otimes G$ may not be computationally feasible. Therefore, we derive the analytical form of the diagonal elements of $(A \otimes G)\Lambda(A \otimes G)^T$ without having to fully evaluate it. Let $A \in \mathbb{R}^{n\times n}$ and $G \in \mathbb{R}^{m\times m}$ be the square matrices. $\Lambda \in \mathbb{R}^{mn\times mn}$ is a diagonal matrix by construction. $V = A \otimes G \in \mathbb{R}^{mn\times mn}$ is a Kronecker product with elements $v_{\alpha,\beta}$ with $\alpha = p(i-1)+k$ (from definition of Kronecker product). Then, the diagonal entries of $(A \otimes G)\Lambda(A \otimes G)^T$ can be computed as follows:

$$\left[(A \otimes G)\Lambda(A \otimes G)^T\right]_{ii} = \sum_{k=1}^{nm}(v_{\alpha,\alpha}\sqrt{\Lambda_k})^2 \tag{19}$$

Derivation: As a first step of the derivation, we express $(A \otimes B)\Lambda(A \otimes B)^T$ in the following form:

$$\begin{aligned}(A \otimes G)\Lambda(A \otimes G)^T &= (A \otimes G)\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}(A \otimes G)^T \\ &= \left[(A \otimes G)\Lambda^{\frac{1}{2}}\right]\left[(A \otimes G)\Lambda^{\frac{1}{2}}\right]^T \\ &= UU^T\end{aligned} \tag{20}$$

Then, $\text{diag}(UU^T)_i = \left[UU^T\right]_{ii} = \sum_{k=1}^{nm} u_{ik}^2$. Now, $(A \otimes B)\Lambda^{\frac{1}{2}} = V\Lambda^{\frac{1}{2}}$ with $\Lambda^{\frac{1}{2}}$ being a diagonal matrix. Therefore, $u_{ik} = v_{\alpha,\alpha}\sqrt{\Lambda_k}$. Substituting back results in $\left[(A \otimes B)\Lambda(A \otimes B)^T\right]_{ii} = \sum_{k=1}^{nm}(v_{\alpha,\beta}\sqrt{\Lambda_k})^2$. Formulating equation 19 and using non-square matrices (after low rank approximation) for $A_{1:a} \in \mathbb{R}^{n\times N}$ and $G_{1:g} \in \mathbb{R}^{m\times M}$ is rather trivial and hence, they are omitted.

B.2 Derivation 2: Analytical form for the efficient and cheap sampler

Let $W^l \in \mathbb{R}^{mn\times mn}$, $W_{\text{MAP}} \in \mathbb{R}^{mn\times mn}$ and $X^s \in \mathbb{R}^{mn\times mn}$ be the random variable, its mean and the samples from a standard Multivariate Normal Distribution respectively. Furthermore, let $\Lambda_{1:L} \in \mathbb{R}^{L\times L}$ and $D \in \mathbb{R}^{mn\times mn}$ are the low ranked form of the re-scaled eigen-values and the diagonal correction term respectively. $U_{A_{1:a}} \in \mathbb{R}^{m\times N}$ and $U_{G_{1:g}} \in \mathbb{R}^{n\times M}$ are the low ranked eigen-basis so that $m \geq M$, $n \geq M$ and $L = MN$.

Given the multivariate normal distribution with the covariance structure as equation 21,

$$vec(W^t) \sim \mathcal{N}(vec(W_{\text{MAP}}), \left((U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D\right)^{-1}) \tag{21}$$

an analytical form of the sampler by given by equation 22 where we have $P = \Lambda_{1:L}^{\frac{1}{2}} L R^{\frac{1}{2}} \in \mathbb{R}^{L\times L}$ with $L = (C^{-1} + V^T V)^{-1} \in \mathbb{R}^{L\times L}$, $V = D^{-\frac{1}{2}}(U_{G_{1:g}} \otimes U_{A_{1:a}})\Lambda_{1:L}^{\frac{1}{2}} \in \mathbb{R}^{mn\times L}$ and $C = A^{-T}(B - I_L)A^{-1} \in \mathbb{R}^{L\times L}$. A and B are Cholesky decomposed matrices of $V^T V \in \mathbb{R}^{L\times L}$ and $V^T V + I_L \in \mathbb{R}^{L\times L}$ respectively. $\odot$ is a notation for element-wise matrix multiplication.

$$W^t = \text{vec}(W_{\text{MAP}}) + \text{vec}\left(X^s \odot D^{-\frac{1}{2}}\right) - \text{vec}\left(\left(U_{G_{1:g}}\text{uvec}\left(P\text{vec}(U_{G_{1:g}}^T(X^s \odot \text{uvec}(D^{-\frac{1}{2}}))U_{A_{1:a}}\right)U_{A_{1:a}}^T\right) \odot \text{uvec}(D^{-\frac{1}{2}})\right) \tag{22}$$

Derivation: Firstly, sampling from a standard multivariate Gaussian for $X^s \in \mathbb{R}^{m\times m}$ is computationally cheap (see equation 23). Given a symmetrical factor for the covariance $\Sigma = AA^T$ (e.g. by Cholesky decomposition), samples can be drawn via $W^t = \text{vec}(W_{\text{MAP}}) + \text{vec}(AX^s)$. Our derivation involves finding such symmetrical factor for the given form of covariance matrix while exploring the Kronecker structure aiming for efficient and cheap computations. We denote $I_{mn} \in \mathbb{R}^{mn\times mn}$ and $I_L \in \mathbb{R}^{L\times L}$.

$$\text{vec}(X^s) \sim N(0, I_{nm}) \;\text{ or }\; X^s \sim MN(0, I_n, I_m) \tag{23}$$

Let us first reformulate the covariance as follows.

$$\begin{aligned}\Sigma &= \left((U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D\right)^{-1} \\ &= \left[D^{\frac{1}{2}}\left(D^{-\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}}\Lambda_{1:L}^{\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T D^{-\frac{1}{2}} + I_{nm}\right)D^{\frac{1}{2}}\right]^{-1} \\ &= D^{-\frac{1}{2}}\left[\left((D^{-\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}})(D^{-\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}})^T + I_{nm}\right)\right]^{-1} D^{-\frac{1}{2}} \\ &= D^{-\frac{1}{2}}\left[VV^T + I_{nm}\right]^{-1} D^{-\frac{1}{2}}\end{aligned} \tag{24}$$

Here, $V = D^{-\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}}$. Now, a symmetrical factor for $\Sigma = AA^T$ can be found by exploiting the above structure. We let W be a symmetrical factor for $VV^T + I_{nm}$ so that $A = D^{-\frac{1}{2}}W^{-1}$ is the symmetrical factor of Σ. Following the work of Ambikasaran & O'Neil (2014) the symmetrical factor W can be found using equations below.

$$\begin{aligned}W &= I_{nm} + VCV^T \\ C &= A^{-T}(B - I_L)A^{-1}\end{aligned} \tag{25}$$

Note that A and B are Cholesky decomposed matrices of V^TV and $V^TV + I_L$ respectively. Now the symmetrical factor for Σ can be expressed as follows.

$$\begin{aligned} A = D^{-\frac{1}{2}}W^{-1} &= D^{-\frac{1}{2}}(I_{nm} + VCV^T)^{-1} \\ &= D^{-\frac{1}{2}}\Big(I_{nm} - V(C^{-1} + V^TV)^{-1}V^T\Big) \end{aligned} \tag{26}$$

Woodbury's Identity were used here. Now, multiplication in $n_\theta \times n_\theta$ space is too expensive for computing $Y_{def} = \text{vec}(W_{\text{MAP}}) + A\text{vec}(X^s)$. We derive more efficient sampling scheme exploiting the Kronecker structure.

$$\begin{aligned} A\text{vec}(X^s) &= D^{-\frac{1}{2}}\Big(I_{nm} - V(C^{-1} + V^TV)^{-1}V^T\Big)\text{vec}(X^s) \\ &= D^{-\frac{1}{2}}\Big(I_{nm} - D^{-\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}}L\Lambda_{1:L}^{\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})^TD^{-\frac{1}{2}}\Big)\text{vec}(X^s) \\ &= D^{-\frac{1}{2}}\text{vec}(X^s) - D^{-1}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}^{\frac{1}{2}}L\Lambda_{1:L}^{\frac{1}{2}}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T\text{vec}(X^s \odot \text{uvec}(D^{-\frac{1}{2}})) \\ &= D^{-\frac{1}{2}}\text{vec}(X^s) - D^{-1}(U_{A_{1:a}} \otimes U_{G_{1:g}})P\text{vec}\Big(U_{G_{1:g}}^T(X^s \odot \text{uvec}(D^{-\frac{1}{2}}))U_{A_{1:a}}\Big) \\ &= \text{vec}\Big(X^s \odot \text{uvec}(D^{-\frac{1}{2}})\Big) - \text{vec}\Big(\Big(U_{G_{1:g}}\text{uvec}\Big(P\text{vec}(U_{G_{1:g}}^T(X^s \odot D^{-\frac{1}{2}})U_{A_{1:a}}\Big)U_{A_{1:a}}^T\Big) \odot \text{uvec}(D^{-1})\Big) \end{aligned} \tag{27}$$

Here, $L = (I - V(C^{-1} + V^TV)^{-1}$ and $P = \Lambda_{1:L}^{\frac{1}{2}}LR^{\frac{1}{2}}$. uvec() is inverse of vectorization operation. All the operations does not need the evaluation of Kronecker products. Inversion and matrix multiplication are performed in lower dimensional space $L = NM$. Note that inversion can be done in off-line setting (before full Bayesian analysis). A crucial component for making the sampling cheap is the Kronecker structure of our covariance matrix in low ranked form.

C Proofs

Proposition 1: *Let $\boldsymbol{I} \in R^{N\times N}$ be the real Fisher information matrix, and let $\boldsymbol{I}_{def} \in R^{N\times N}$ and $\hat{\boldsymbol{I}}_{def} \in R^{N\times N}$ be our estimates of it with rank d and k such that k < d. Their diagonal entries are equal that is $\boldsymbol{I}_{ii} = \boldsymbol{I}_{def_{ii}} = \hat{\boldsymbol{I}}_{def_{ii}}$ for all i = 1, 2, . . . , N.*

proof: The proof trivially follows from the definitions of $\boldsymbol{I} \in R^{N\times N}$, $\boldsymbol{I}_{\text{def}} \in R^{N\times N}$ and $\hat{\boldsymbol{I}}_{\text{def}} \in R^{N\times N}$. As the exact Fisher is an expectation on outer products of back-propagated gradients, its diagonal entries equal $\boldsymbol{I}_{ii} = \mathbb{E}\left[\delta\theta_i^2\right]$ for all i = 1, 2, . . . , N.

In the case of full ranked $\boldsymbol{I}_{def}$, substituting $D_{ii} = \mathbb{E}\left[\delta\theta_i^2\right] - \sum_{k=1}^{nm}(v_{\alpha,\alpha}\sqrt{\Lambda_k})^2$ with $\sum_{k=1}^{nm}(v_{\alpha,\alpha}\sqrt{\Lambda_k})^2 = (U_A \otimes U_G)\Lambda(U_A \otimes U_G)_{ii}^T$ results in equation 28 for all i = 1, 2, . . . , N.

$$\begin{aligned} \boldsymbol{I}_{def_{ii}} &= (U_A \otimes U_G)\Lambda(U_A \otimes U_G)_{ii}^T + D_{ii} \\ &= (U_A \otimes U_G)\Lambda(U_A \otimes U_G)_{ii}^T + \mathbb{E}\left[\delta\theta_i^2\right] - (U_A \otimes U_G)\Lambda(U_A \otimes U_G)_{ii}^T = \mathbb{E}\left[\delta\theta_i^2\right] \end{aligned} \tag{28}$$

Similarly, we substitute $\hat{D}_{ii} = \mathbb{E}\left[\delta\theta_i^2\right] - \sum_{k=1}^{NM}(\hat{v}_{\alpha,\alpha}\sqrt{\Lambda_{1:L}})^2$ with $\sum_{k=1}^{NM}(\hat{v}_{\alpha,\alpha}\sqrt{\Lambda_{1:L}})^2 = (U_{A_{1:N}} \otimes U_{G_{1:M}})\Lambda_{1:L}(U_{A_{1:N}} \otimes U_{G_{1:M}})_{ii}^T$ which results in equation 29 for all i = 1, 2, . . . , N.

$$\begin{aligned} \hat{\boldsymbol{I}}_{def_{ii}} &= (U_{A_{1:N}} \otimes U_{G_{1:M}})\Lambda_{1:L}(U_{A_{1:N}} \otimes U_{G_{1:M}})_{ii}^T + D_{ii} \\ &= (U_{A_{1:N}} \otimes U_{G_{1:M}})\Lambda_{1:L}(U_{A_{1:N}} \otimes U_{G_{1:M}})_{ii}^T + \mathbb{E}\left[\delta\theta_i^2\right] - (U_{A_{1:N}} \otimes U_{G_{1:M}})\Lambda_{1:L}(U_{A_{1:N}} \otimes U_{G_{1:M}})_{ii}^T \\ &= \mathbb{E}\left[\delta\theta_i^2\right] \end{aligned} \tag{29}$$

Therefore, we have $\boldsymbol{I}_{ii} = \boldsymbol{I}_{\text{def}_{ii}} = \hat{\boldsymbol{I}}_{\text{def}_{ii}}$ for all i = 1, 2, . . . , N.

Lemma 1: *Let $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ be the real Fisher information matrix, and let $\boldsymbol{I}_{def} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{efb} \in \mathbb{R}^{N\times N}$ be the DEF and EFB estimates of it respectively. It is guaranteed to have $\left\|\boldsymbol{I} - \boldsymbol{I}_{efb}\right\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{def}\right\|_F$.*

proof: Let $e^2 = \|A - B\|_F^2$ define a squared Frobenius norm of error between the two matrices $A \in \mathbb{R}^{N\times N}$ and $B \in \mathbb{R}^{N\times N}$. Now, e^2 can be formulated as,

$$\begin{aligned} e_b^2 &= \|A - B\|_F^2 \\ &= \sum_i (A-B)_{ii}^2 + \sum_i \sum_{j\neq i} (A-B)_{ij}^2 \end{aligned} \tag{30}$$

The first term of equation 30 belongs to errors of diagonal entries in B wrt A whilst the second term is due to the off-diagonal entries.

Now, it follows that,

$$\begin{gathered} \|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{\text{def}}\|_F \\ e_{efb}^2 \geq e_{def}^2 \\ \textstyle\sum_i (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ii}^2 + \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ij}^2 \geq \sum_i (\boldsymbol{I} - \boldsymbol{I}_{\text{def}})_{ii}^2 + \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{def}})_{ij}^2 \\ \textstyle\sum_i (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ii}^2 + \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ij}^2 \geq \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{def}})_{ij}^2 \\ \textstyle\sum_i (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ii}^2 + \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ij}^2 \geq \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ij}^2 \end{gathered}$$

Note that $\sum_i (\boldsymbol{I} - \boldsymbol{I}_{\text{def}})_{ii}^2 = 0$ using proposition 1. Furthermore, $\sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{def}})_{ij}^2 = \sum_i \sum_{j\neq i} (\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})_{ij}^2$ since by definition, $\boldsymbol{I}_{\text{efb}}$ and $\boldsymbol{I}_{\text{def}}$ have the same off-diagonal terms.

Corollary 1: *Let $\boldsymbol{I}_{kfac} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{def} \in \mathbb{R}^{N\times N}$ be KFAC and our estimates of real Fisher Information matrix $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ respectively. Then, it is guaranteed to have $\|\boldsymbol{I} - \boldsymbol{I}_{KFAC}\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{def}\right\|_F$.*

For interested readers, find the proof $\|\boldsymbol{I} - \boldsymbol{I}_{KFAC}\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F$ in George et al. (2018). Note that $\|\boldsymbol{I} - \boldsymbol{I}_{KFAC}\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F$ may not mean that $\left\|\boldsymbol{I}^{-1} - \boldsymbol{I}_{KFAC}^{-1}\right\|_F \geq \left\|\boldsymbol{I}^{-1} - \boldsymbol{I}_{\text{efb}}^{-1}\right\|_F$. Yet, our proposed approximation yield better estimates than KFAC and EFB in the information form of multivariate normal distribution.

Lemma 2: *Let $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ be the real Fisher information matrix, and let $\boldsymbol{I}_{1:K}^{top} \in \mathbb{R}^{N\times N}$, $\boldsymbol{I}_{1:L}^{top} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{I}_{1:L} \in \mathbb{R}^{N\times N}$ be the low rank estimates of $\boldsymbol{I}$ of EKB obtained by preserving top K, L and top K plus additional J resulting in L eigenvalues. Here, we define $K < L$. Then, the approximation error of $\boldsymbol{I}_{1:L}$ is bounded as follows: $\left\|\boldsymbol{I} - \boldsymbol{I}_{1:K}^{top}\right\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{1:L}\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{1:L}^{top}\right\|_F$.*

proof: From the definition, $(U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T = V\Lambda V^T$ is PSD as $\Lambda_{ii} = \mathbb{E}\left[(V^T\delta\theta)_i^2\right] \geq 0$ for all elements i and $VV^T = I$ with I as an identity matrix (orthogonality). Naturally, low rank approximations $(U_A \otimes U_G)_{1:L^{\text{top}}}\Lambda_{1:L^{\text{top}}}(U_A \otimes U_G)_{1:L^{\text{top}}}^T$, $(U_A \otimes U_G)_{1:K^{\text{top}}}\Lambda_{1:K^{\text{top}}}(U_A \otimes U_G)_{1:K^{\text{top}}}^T$ and $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T = (U_A \otimes U_G)_{1:L}\Lambda_{1:L}(U_A \otimes U_G)_{1:L}^T$ are again PSD by the fact that low rank approximation does not introduce negative eigenvalues.

Now, a well known fact from dimensional reduction literature is that low rank approximation preserving the top eigenvalues result in best approximation errors in terms of Frobenius norm for the given rank. Informally stating Wely's ideas on eigenvalue perturbation:

Let $B \in \mathbb{R}^{m\times n}$ with rank smaller or equal to p (one can also use complex space $\mathbb{C}$ instead of $\mathbb{R}$) and let $E = A - B$ with $A \in \mathbb{R}^{m\times n}$. Then, it follows that,

$$\|A - B\|_F^2 = \sigma_1(A-B)^2 + \cdots + \sigma_\mu(A-B)^2 \geq \sigma_{p+1}(A-B)^2 + \cdots + \sigma_\mu(A-B)^2 = \left\|A - B_{1:p}\right\|_F^2, \tag{31}$$

where $\sigma_1, \cdots \sigma_\mu$ are the singular values of A with $\mu = \min(n, m)$. The convention here is that $\sigma_i(A)$ is the ith largest singular value and $\sigma_i(A) = 0$ for $i >$ rank(A). Using this insight, and the fact that in the given settings, squared singular values are variances in new space lead to:

$$\left\|\boldsymbol{I} - \boldsymbol{I}_{1:K}^{\text{top}}\right\|_F \geq \|\boldsymbol{I} - \boldsymbol{I}_{1:L}\|_F \geq \left\|\boldsymbol{I} - \boldsymbol{I}_{1:L}^{\text{top}}\right\|_F$$

This bound provides insight that if preserving top L eigenvalues result in prohibitively too large covariance matrix, our LRA provides an alternative better than preserving top K eigenvalues given K < L. In practise, note that $\boldsymbol{I}_{1:L}$ can be memory efficient as we formulate $\boldsymbol{I}_{1:L} = (U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T = (U_A \otimes U_G)_{1:L}\Lambda_{1:L}(U_A \otimes U_G)^T_{1:L}$. Consequently, preserving smaller matrices can often be more efficient than even $\boldsymbol{I}^{\text{top}}_{1:K}$.

Lemma 3: *The low rank matrix* $\hat{\Sigma} = \left((U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D\right)^{-1} \in \mathbb{R}^{N\times N}$ *is a non-degenerate covariance matrix if the diagonal correction matrix D and LRA* $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T$ *are both symmetric and positive definite. This condition is satisfied if* $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T_{ii} < \mathbb{E}\left[\delta\theta_i^2\right]$ *for all* $i \in \{1, 2, \cdots, N\}$ *and with* $\Lambda_{1:L} \nsubseteq 0$.

proof: Let us first rewrite $\hat{\boldsymbol{I}}_{\text{def}} = (U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D$ in the following form.

$$\begin{aligned}(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D &= (U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda^{\frac{1}{2}}_{1:L}\Lambda^{\frac{1}{2}}_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T + D \\ &= \left[(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda^{\frac{1}{2}}_{1:L}\right]\left[(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda^{\frac{1}{2}}_{1:L}\right]^T + D \\ &= UU^T + D\end{aligned} \tag{32}$$

Now, if D and $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T$ is both symmetric and positive definite, it follows that for an arbitrary vector $x \in \mathbb{R}^d$, $x^T UU^T x > 0$ as eigen-values $R_i > 0$ by construction. Furthermore, $x^T Dx > 0$ also holds by the definition of positive definiteness. Therefore, we have $x^T(UU^T + D)x = x^T UU^T x + x^T Dx > 0$ which leads to the proof that $\boldsymbol{I}_{\text{def}}$ is positive definite if D and $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T$ is both symmetric and positive definite. As this results in non-degenerate IM, the canonical covariance Σ is non-degenerate as well.

Trivially following the definition of $D_{ii} = \mathbb{E}\left[\delta\theta_i^2\right] - (U_A \otimes U_G)\Lambda(U_A \otimes U_G)^T_{ii}$, $D_{ii} > 0$ for all i when $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T_{ii} < \mathbb{E}\left[\delta\theta_i^2\right]$. Again, by the definition of $\Lambda_{ii} = \mathbb{E}\left[(V^T\delta\theta)_i^2\right] \geq 0$, $\Lambda_{1:L}$ containing no zero eigenvalues result in the positive definite matrix $(U_{A_{1:a}} \otimes U_{G_{1:g}})\Lambda_{1:L}(U_{A_{1:a}} \otimes U_{G_{1:g}})^T$.

Lemma 4: *Let* $\boldsymbol{I} \in \mathbb{R}^{N\times N}$ *be the real Fisher information matrix, and let* $\hat{\boldsymbol{I}}_{def} \in \mathbb{R}^{N\times N}$, $\boldsymbol{I}_{efb} \in \mathbb{R}^{N\times N}$ *and* $\boldsymbol{I}_{kfac} \in \mathbb{R}^{N\times N}$ *be the low rank DEF, EFB and KFAC estimates of it respectively. Then, it is guaranteed to have* $\left\| diag(\boldsymbol{I}) - diag(\boldsymbol{I}_{efb_{ii}})\right\|_F \geq \left\| diag(\boldsymbol{I}_{ii}) - diag(\hat{\boldsymbol{I}}_{def_{ii}})\right\|_F = 0$ *and* $\left\| diag(\boldsymbol{I}_{ii}) - diag(\boldsymbol{I}_{kfac_{ii}})\right\|_F \geq \left\| diag(\boldsymbol{I}_{ii}) - diag(\hat{\boldsymbol{I}}_{def_{ii}})\right\|_F = 0$. *Furthermore, if the eigenvalues of* $\hat{\boldsymbol{I}}_{def}$ *contains all non-zero eigenvalues of* $\boldsymbol{I}_{def}$, *it follows:* $\left\|\boldsymbol{I} - \boldsymbol{I}_{efb}\right\|_F \geq \left\|\boldsymbol{I} - \hat{\boldsymbol{I}}_{def}\right\|_F$

proof: The first part follows from proposition 1 which states that for all the elements i, $\boldsymbol{I}_{ii} = \hat{\boldsymbol{I}}_{\text{def}_{ii}}$, $\left\| \text{diag}(\boldsymbol{I}) - \text{diag}(\boldsymbol{I}_{\text{efb}_{ii}})\right\|_F \geq \left\| \text{diag}(\boldsymbol{I}_{ii}) - \text{diag}(\hat{\boldsymbol{I}}_{\text{def}_{ii}})\right\|_F = 0$ and $\left\| \text{diag}(\boldsymbol{I}_{ii}) - \text{diag}(\boldsymbol{I}_{\text{kfac}_{ii}})\right\|_F \geq \left\| \text{diag}(\boldsymbol{I}_{ii}) - \text{diag}(\hat{\boldsymbol{I}}_{\text{def}_{ii}})\right\|_F = 0$. This results by the design of the method, in which, we correct the diagonal entries in parameter space after the LRA.

For the second part of the proof, lets recap that Lemma 2 (Wely's idea on eigenvalue perturbation) that removing zero eigenvalues does not affect the approximation error in terms of Frobenius norm. This then implies that off-diagonal elements of $\hat{\boldsymbol{I}}_{\text{def}}$ and $\boldsymbol{I}_{\text{efb}}$ are equivalent. Then,:

$$\begin{gathered}\|\boldsymbol{I} - \boldsymbol{I}_{\text{efb}}\|_F \geq \left\|\boldsymbol{I} - \hat{\boldsymbol{I}}_{\text{def}}\right\|_F \\ e^2_{efb} \geq e^2_{def} \\ \textstyle\sum_i(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ii} + \sum_i\sum_{j\neq i}(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ij} \geq \sum_i(\boldsymbol{I} - \hat{\boldsymbol{I}}_{\text{def}})^2_{ii} + \sum_i\sum_{j\neq i}(\boldsymbol{I} - \hat{\boldsymbol{I}}_{\text{def}})^2_{ij} \\ \textstyle\sum_i(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ii} + \sum_i\sum_{j\neq i}(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ij} \geq \sum_i\sum_{j\neq i}(\boldsymbol{I} - \hat{\boldsymbol{I}}_{\text{def}})^2_{ij} \\ \textstyle\sum_i(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ii} + \sum_i\sum_{j\neq i}(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ij} \geq \sum_i\sum_{j\neq i}(\boldsymbol{I} - \boldsymbol{I}_{\text{efb}})^2_{ij}\end{gathered}$$

Again, $\sum_i(\boldsymbol{I} - \hat{\boldsymbol{I}}_{\text{def}})^2_{ii} = 0$ according to proposition 1 for all the elements i.

D Implementation Details and Further Results

KFAC library from Tensorflow [3] was used to implement the Fisher estimator (Martens & Grosse, 2015) for our methods and the works of Ritter et al. (2018a). Note that empirical Fisher usually is not a good estimates as it is typically biased (Martens & Grosse, 2015) and therefore, we did not use it. KFAC library offers several estimation modes for both fully connected and convolutional layers. We have used the gradients mode for KFAC Fisher estimation (which is also crucial for our pipelines) whereas the exact mode was used for diagonal approximations. We did not use the exponential averaging for all our experiments as well as the inversion scheme in the library. However, when using it in practice, it might be useful especially if there are too many layers that one cannot access convergence of the Fisher estimation. We have used NVIDIA Tesla for grid searching the parameters of Diag and KFAC Laplace, and 1080Ti for all other experiments.

D.1 Toy Regression Dataset

Apart from the architecture choices discussed in section 4, the training details are as follows. A gradient descent optimizer from tensorflow has been used with a learning rate of 0.001 with zero prior precision or L2 regularization coefficient ($\tau = 0.2$ for KFAC, $\tau = 0.45$ for Diag, $N = 1$ and $\tau = 0$ for both FB and DEF have been used). Mean squared error (MSE) has been used as its loss function. Interestingly, the exact block-wise Hessian and their approximations for the given experimental setup contained zero values on its diagonals. This can be interpreted as zero variance in information matrix, meaning no information, resulting in information matrix being degenerate for the likelihood term. In such cases, the covariance may not be uniquely defined (Thrun et al., 2004). Therefore, we treated these variances deterministic, making the information matrix non-degenerate (motivated from Lemma 3 in section 3).

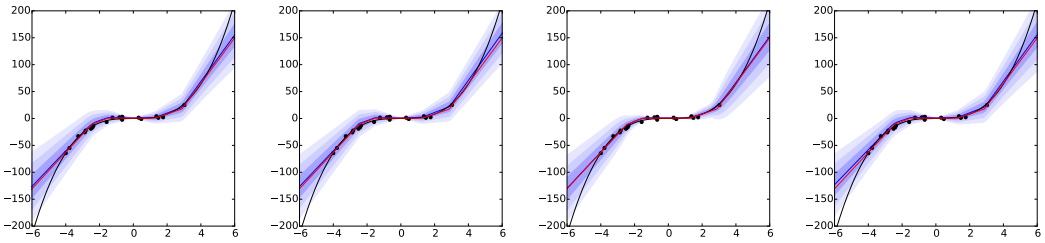
More importantly, we present a detailed analysis to avoid misunderstanding about our toy dataset experiments. As a starting remark, a main advantage of this toy regression problem is that it simplifies the understandings of on-going process, in lieu of sophisticated networks with a large number of parameters. Typically, as of Herandez-Lobato & Adams (2015), Ritter et al. (2018a), Gal (2016), or even originating back to Gaussian processes literature, this example has been used to check the predictive uncertainty by qualitatively evaluating on whether the method predicts high uncertainty in the regimes of no training data. However, a drawback exists: no quantitative analysis has been reported to our knowledge other than qualitatively comparing it to community wide accepted ground truth such as Hamiltonian Monte Carlo Sampling (Neal, 1996), and LA using KFAC and Diag seem to be sensitive to hyperparameters in this dataset which makes the comparison difficult.

This is illustrated in figure 5 where we additional introduce Random which is just a user-set τI for covariance estimation in order to demonstrate this. Qualitatively analyzing from the first look, all the methods look very similar in delivering high uncertainty estimates in the regimes of no training data. Here, we note that the same hyperparameter settings have been used for Diag, KFAC and FB Laplace whereas the user-set $\tau = 7$ has been found for Random. This agrees to the discussions of Ritter et al. (2018a) as KFAC resulted in less τ when compared to Diag Laplace.

However, we also observed that without the approximation step of equation 4 (denoted OKF), using the same hyper parameter as above resulted in visible over-prediction of uncertainty and inaccurate estimates on the prediction. This is shown in figure 6. Again, tuning the parameter to a higher precision τ, similar behavior to figure 5 can be reproduced. This can be analyzed by visualizing the covariance of KFAC and OKF. As it can be seen, in this experiment settings, figure 6 shows that equation 4 damps the magnitude of estimated covariance matrix.

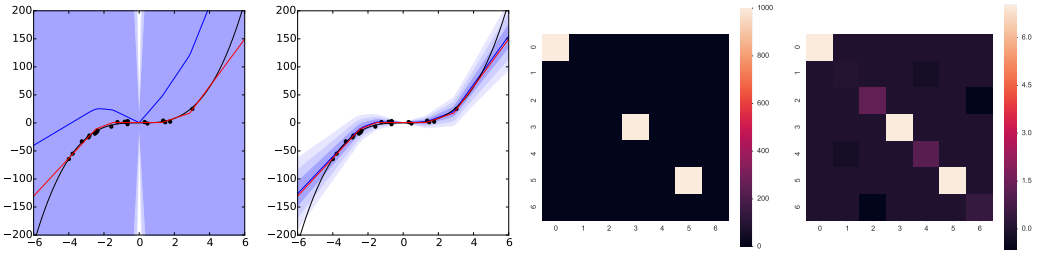
A possible explanation is that if the approximate Hessian is degenerate, then small τI places a big mass on areas of low posterior probabilities for some network parameters with no information (zero variance and correlations in the approximate Hessian). This can be seen in figure 6 part (a) where the approximate Hessian contains 3 parameters with exactly zero diagonal elements and zeros in its off-diagonal elements. If one tries to add a small $\tau = 0.001$ here, then the covariance of these parameters get close to its inverse $\tau^{-1} = 1000$ as shown in figure 6 part (c). This would in return result in over prediction of uncertainty and inaccurate predictions which explains figure 6 part (a).

[3] `https://github.com/tensorflow/kfac`



(a) KFAC Laplace (b) Diag Laplace (c) Random (d) FB Laplace

Figure 5: Toy regression uncertainty. User-set Laplace means user-set τI for covariance estimation. This shows that if one tries to tune the "regularizing" parameters, all these approximations to the true Hessian behaves similarly within this experiment. Now trained with 20 data points.

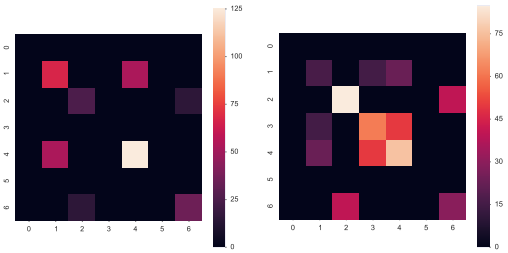


(a) OKF Laplace (b) OKF Laplace (tuned) (c) OKF inverse (d) KFAC inverse

Figure 6: Toy regression uncertainty and covariance visualization (only the first layer is shown here). OKF Laplace means using the left hand side of equation 4 without further approximation (only possible with this small model and data-set).

Another interesting experiments are studying the effects of dataset size to number of parameters. For this, we have increased the dataset size to 100 in oppose to 20. Again, we now compare the approximate Hessian by visualizing them. Notably, at using 100 data points resulted in more number of zero diagonal entries and corresponding rows and columns. This is due to over parameterization of the model which results in under determined Hessian.

These insights hint for the followings. Accurately estimating the Hessian while forcing its estimates non-degeneracy via not considering zero eigenvalues for this data and model can lead to less sensitivity to its hyperparameters or τ in particular. Secondly, further increasing or decreasing the ratio of data points to number of parameters change the approximate Hessian (similarly found for estimates of Fisher) changes its structure, and can lead to under-determined approximation (therefore, changing its loss landscape). Finally, if the Hessian is under-determined, hyperparameters τ affects the resulting predictive uncertainty (or covariance) if its magnitude significantly differs (and in case of KFAC). However, as more detailed experimental analysis is outside the scope of the paper, can be an interesting future work to further analyze the relation between the hyperparameters, their probabilistic interpretation and resulting loss landscape of neural network.



(a) the Hessian [100]. (b) the Hessian [20]

Figure 7: Visualization of the approximate Hessian with 20 and 100 data points (a and b respectively). Only the first layer shown here.

D.2 Classification Tasks

Most of the implementations for MNIST and CIFAR10 experiments were taken from Tensorflow tutorials [4] including the network architectures and training pipelines if otherwise stated in the main text. This is in line of argument that our method can be directly applied to existing, and well trained neural networks. For MNIST experiments, the architecture choices are the followings. Firstly, no down-scaling has been performed to its inputs. The architecture constitutes 2 convolutional layers followed by 2 fully connected layer (each convolutional layer is followed by a pooling layer of size 2 by 2, and a stride 2). For flattening from the second convolutional layer to the first fully connected layer, a pooling operation of 49 by 64 has been naturally used. RELU activation have been used for all the layers except the last layer which computes the softmax output. Dropout has been applied to the fully connected layer with dropout rate of 0.6 after a grid search (explained in section D.2.1). Regarding the loss functions, cross entropy loss has been used with ADAM as its optimizer and learning rate of 0.001. An important information is the size of each layers. The first layer constitutes 32 filters with 5 by 5 kernel, followed by the second layer with 64 filters and 5 by 5 kernel. The first fully connected layer then constitutes 1024 units and the last one ends with 10 units. We note that, this validates our method on memory efficiency as the third layer has a large number of parameters, and its covariance, being quadratic in its size, cannot be stored in our utilized GPUs.

Regarding the architecture selection of CIFAR10 experiments, no down-scaling of the inputs has be done. The chosen architecture is composed of 2 convolutional layers followed by 3 fully connected layers. Pooling layers of size 3 by 3 with strides 2 have been applied to outputs of the convolutional layers. Obviously, the third convolutional layer is pooled to match the input size of the following fully connected layers. Batch normalization has been applied to each outputs of convolutional layer before pooling, with bias 1, α of 0.001/9.0 and β of 0.75 (notations are different to the main text, and this follows that of tensorflow library). A weight decay factor of 0.004 has been used, and trained again with cross entropy loss, now with a stochastic gradient descent. Learning rate of 0.001 has been used. Again, the most relevant settings are: the first layer constitutes 5 by 5 kernel with 64 filters. This is then again followed by the same (but as input to CIFAR10 is RGB, the second layer naturally has more number of parameters). Units of 384, 192, and 10 have been used for the fully connected layers in an ascending order. Lastly, random cropping, flipping, brightness changes and contract have

[4] `https://www.tensorflow.org/tutorials`

been applied as the data augmentation scheme. Similar to MNIST experiments, we validate our claim that the LRA is necessary with CIFAR10 data sets.

Unlike Ritter et al. (2018a) we did not artificially augment the data for MNIST experiments because the usual training pipeline did not require it. We have augmented the data for the Fisher estimation only if the network architecture required it (e.g. one we used for CIFAR10 experiments). For our low rank approximation, we always have used the maximum rank we could fit, after removing all the zero eigenvalues. The details are listed as follows: In MNIST experiments, 450, 5185, 20625 and 4775 have been the resulting rank of our sparsification algorithm for layers of sequential order. For CIFAR10 experiments, 4800, 2112, 398, 5499 and 1920 have been the used ranks in a sequential order to the layers. It would be useful to understand the reasons behind such differences per layer. Lastly we have used 1000 Monte-Carlo samples for MNIST experiments, and 100 samples for CIFAR10 and toy regression dataset experiments.

D.2.1 Benchmark implementations

Implementation of deep ensemble (Lakshminarayanan et al., 2017) was kept rather simple by not using the adversarial training, but we combined 15 networks that were trained with different initialization. The same architecture and training procedure were used for all. Note that CIFAR10 experiments with similar convolutional architectures were not present in the works of (Lakshminarayanan et al., 2017) to the best of our knowledge. On MNIST, Louizos & Welling (2017) found similar results to ours that deep ensemble performed similar to the MC-dropout (Gal, 2016). For dropout, we have tried a grid search of dropout probabilities of 0.5 and 0.8, and have reported the best results. For the methods based on Laplace approximation, we have performed grid search on hyperparameters N of (1, 50000, 100000) and 100 values of τ were tried using known class validation set. Note that for every method, and different data-sets, each method required different values of τI to give a reasonable accuracy. The starting point of the grid-search were determined based on if the mean values of their predictions were obtained similar accuracy to the deterministic counter parts. The figure below are the examples on MNIST where minimum ece points were selected and reported.

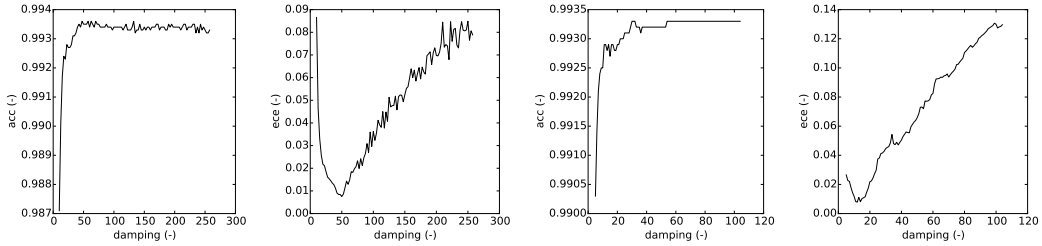


Figure 8: Grid search for diag Laplace (left two figures) and KFAC Laplace (right two) with pseudo observation term 50000 on MNIST.