Low Rank Based Subspace Inference for the Laplace Approximation of Bayesian Neural Networks

Josua Faller* and **Jörg Martin***
Physikalisch-Technische Bundesanstalt. Abbestr. 2-12, 10587 Berlin, Germany

Abstract

Subspace inference for neural networks assumes that a subspace of their parameter space suffices to produce a reliable uncertainty quantification. In this work, we underpin the validity of this assumption by using low rank techniques. We derive an expression for a subspace model to a Bayesian inference scenario based on the Laplace approximation that is, in a certain sense, optimal given a specific dataset. We empirically show that a Laplace approximation constructed with a dimensionally reduced covariance matrix closely matches the full Laplace approximation obtained using the exact covariance matrix. Where feasible, this subspace model can serve as a baseline for benchmarking the performance of subspace models. In addition, we provide a scalable approximation of this subspace construction that is usable in practice and compare it to existing subspace models from the literature. In general, our approximation scheme outperforms previous work. Furthermore, we present a metric to qualitatively compare the approximation quality of different subspace models even if the exact Laplace approximation is unknown.

1 INTRODUCTION

Bayesian modelling is an elegant and flexible method to quantify uncertainties of parametric models. Treating the parameters of the model as random variables allows to incorporate model uncertainty. Bayesian neural networks (Gal, 2016; Blundell et al., 2015; Kendall and Gal, 2017; Hernández-Lobato and Adams, 2015; Maddox et al., 2019) implement this idea for neural networks (NNs). In practice, however, full posterior inference over Bayesian NNs is intractable due to the large number of parameters that define NNs. Thus, to quantify the uncertainty of a certain model, practitioners have to approximate the exact posterior distribution by a simpler one. Several methods were developed to make this approximation feasible: The posterior distribution can be approximated, e.g., by variational inference (Blundell et al., 2015; Kingma et al., 2015; Gal, 2016; Kendall and Gal, 2017; Jordan et al., 1999; Wainwright and Jordan, 2008). A different idea, that goes in fact back to the 90s, is to use the Laplace approximation (LA) (MacKay, 1992b), which has found increasing popularity in recent years due to scalable approximations (LeCun et al., 1989; Ritter et al., 2018) and its flexible usability (Daxberger et al., 2021a). Moreover, in contrast to variational-inference-based approaches, it can be applied to off-the-shelf networks without any retraining: Given a maximum a posteriori (MAP) solution, that often coincides with the minimum of canonical loss functions, the LA replaces the exact posterior by a Gaussian distribution with the MAP as the mean and the inverse of the negative Hessian of the log posterior at the MAP as covariance matrix.

However, this approximation is still infeasible for NNs since the Hessian scales quadratically in the number of parameters such that often it cannot be computed or even stored, let alone be inverted. In addition, training NNs is a high dimensional non-convex optimization problem. In practice fully trained NNs are not located in a minimum of the loss function but rather on a saddle point (Dauphin et al., 2014). Hence, the so-computed Hessian is in general not positive semi-definite (Sagun et al., 2016; Papyan, 2018). A partial solution to these issues is provided by approximating the Hessian by the generalized Gauss-Newton (GGN) matrix, which is identical to the Fisher Information matrix for common likelihoods (Schraudolph, 2002; Pascanu and Bengio, 2013; Martens, 2014). The GGN matrix is positive

*Both authors contributed equally.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

semi-definite and is constructed from objects that are feasible to compute, cf. Section 3 for details.

However, it's sheer size makes the GGN matrix still unstorable, even for medium sized networks. Thus, to make the LA feasible for NNs, additional steps are necessary to reduce the size of the Hessian and to allow for an easier computation of its inverse. Common approaches include approximations via a diagonal (LeCun et al., 1989; Salimans and Kingma, 2016; Kirkpatrick et al., 2017), last layer (Kristiadi et al., 2020), a Kronecker-factored (Ritter et al., 2018) structure or dimensional reductions from the original p-dimensional space to the s-dimensionial subspace.

The last method gained more attention through a recent series of works which argue that it might suffice to consider partially stochastic NNs (Kristiadi et al., 2020; Snoek et al., 2015; Izmailov et al., 2019; Daxberger et al., 2021b; Sharma et al., 2023) that is NNs where the Bayesian inference is performed in a lower dimensional subspace. NNs are heavily overparametrized and the idea is that a subset or well-selected linear combinations of parameters are sufficient to obtain reliable uncertainty estimates. We refer to this idea in this work as *subspace inference*. Daxberger et al. (2021b) apply this idea to make the LA for Bayesian NNs feasible by storing only a submatrix of the full GGN matrix. The submatrix is constructed using a subset of parameters that can be found via a diagonal approximation of the Hessian (Daxberger et al., 2021b), via the magnitude of the parameters (Cheng et al., 2017) or via an application of SWAG (Maddox et al., 2019).

Methods to reduce the size of the Hessian offer several computational advantages in comparison to the full LA. On the one hand, the memory complexity of the covariance matrix is reduced from $\mathcal{O}\left(p^2\right)$ to $\mathcal{O}\left(s^2\right)$, and on the other hand, the time complexity to draw Monte Carlo samples is of the order of $\mathcal{O}\left(s^3\right)$ instead of $\mathcal{O}\left(p^3\right)$.

The aim of our work is to give a systematic, generic and statistically sound approach to study the usability of subspace inference for the LA of Bayesian NNs. Similar as Daxberger et al. (2021b) we use the widespread combination of the LA with a linearization of our NN f_θ around the MAP value $\hat{\theta}$ of the parameters θ (Foong et al., 2019; Immer et al., 2021; Deng et al., 2022; Ortega et al., 2023):

$$f_{\mathrm{Lin},\theta}(X) = f_{\hat{\theta}}(X) + J_X(\theta - \hat{\theta}), \tag{1}$$

where $J_X = \nabla_\theta f_\theta(X)|_{\theta=\hat{\theta}}$. This method is known as the linearized LA. Our method differs from existing work by making the *predictive covariance* of the linearized Laplace approximation the centerpiece of our analysis. We do so for the following reasons:

1. The posterior predictive distribution is the actual objective: This was pointed out already in previous work, e.g. in (Izmailov et al., 2021): as prediction happens in function space the posterior predictive and not the posterior is the object we actually care about.
2. The predictive covariance determines the posterior predictive distribution completely under LA: As LA always starts with the MAP solution, the only object that differs from approach to approach is the variance estimate that is entirely determined by the predictive covariance. Thus, its approximation is all that matters.

This viewpoint allows us to give some precise statements of approximation quality and optimality. The contributions of our article are as follows:

1. We specify an optimality criterion for subspace LAs based on closeness to the full GGN Laplace approximation. In contrast to prior work using parameter-based heuristics, we target the *predictive covariance* directly, which determines predictive uncertainty in the linearized LA framework.
2. We prove existence and uniqueness of the optimal subspace (Theorem 1) and provide an explicit formula. Empirically, this baseline can achieve faithful approximation with $<1\%$ of parameters.
3. As the subspace from 2 can typically not be computed in practice, we provide a feasible approximation and observe that it performs in many cases superior to the subset selection of (Daxberger et al., 2021b,a).
4. We propose a trace-based criterion for comparing subspace models when the full LA is unknown, and validate empirically that it correlates with KL-divergence to the full LA.

This article is organized as follows: In Section 2 we recall recent work on the subject of the article and then evoke some background on the LA for Bayesian NNs in Section 3. In Section 4 we provide the main theoretical contributions of this work. In Section 5 several experiments to empirically verify our theoretical analysis are carried out. Additional information is provided in the Appendix.

2 RECENT WORK

Laplace Approximation. The first application of the LA using the Hessian for NNs was introduced

by MacKay (1992b). MacKay (1992a) also proposed an approximation similar to the generalized Gauss-Newton (GGN) method. The combination of scalable factorizations or diagonal Hessian approximations with the GGN approximation (Schraudolph, 2002; Martens, 2014) made the LA applicable for larger networks. In particular, the GGN approximation gained more attention due to the introduction of the Kronecker-Factored Approximate Curvature (KFAC) (Ritter et al., 2018; Botev et al., 2017; Martens and Grosse, 2015) which is scalable and outperforms the diagonal Hessian approximation. Due to underfitting issues of the LA (Foong et al., 2019), the linearized LA based on (1) was developed (Immer et al., 2021). In this work, we use the same setting.

Partially Stochastic Neural Networks. The study of partially stochastic NNs gained some attention because of their computational efficiency. But even from a statistics viewpoint, partially stochastic NNs are attractive because they can capture the uncertainty of the full model by using only a fraction of the parameters (Sharma et al., 2023; Andrade and Sato, 2024; Zhao et al., 2023). Sharma et al. (2023) developed the concept of Universal Conditional Distribution Approximators and proved that certain partially stochastic NNs can form samplers of any continuous target conditional distribution arbitrary well. Calvo-Ordonez et al. (2024) extended this idea to infinitely deep Bayesian NNs.

Lee et al. (2020) and Yang et al. (2024) apply low rank approximations for dimensional reduction in the context of the KFAC approximation. Izmailov et al. (2019) developed a low-dimensional affine subspace inference scheme. They select a linear combination of parameter vectors which span a vector space around the MAP. Since they consider low-dimensional subspaces ($s \leq 10$) different methods are available to approximately sample from the posterior distribution. However, they observed that the posterior obtained in these low-dimensional subspaces is too concentrated, so that only a tempered posterior leads to reasonable uncertainties. Dold et al. (2024) apply this idea to semi-structured models. Daxberger et al. (2021b) choose a subset of parameters to construct a subspace model. This subset is selected by the parameters that have the highest posterior variance. However, this work often requires the selection of quite a large number of parameters (s up to $4 \cdot 10^4$). Our framework is closest to this work. In contrast to previous work, we study the predictive instead of the posterior distribution to obtain a feasible parameter subspace. In addition, we observe that neither an ad hoc tempering of the posterior distribution nor thousands of parameters are needed to construct faithful lower dimensional approximations.

3 TERMINOLOGY AND BACKGROUND

Setup and Notational Remarks. We consider the supervised learning framework. We model the relation between the independent observable x and the target y by a parametric distribution $p(y|x, \theta)$ with parameters $\theta \in \mathbb{R}^p$. Different observations are, as usual, assumed to be independent and identically distributed. We denote the training set of observations as $\mathcal{D} = \{(x_i, y_i)|1 \leq i \leq N\}$, where N denotes the number of observations. We study regression and classification tasks. C represents the number of outputs $f_\theta(x) = (f_\theta^1(x), \dots, f_\theta^C(x))^\intercal \in \mathbb{R}^C$ of the NN f_θ, for both, regression and classification problems. For regression we make a Gaussian model assumption $p(y|x, \theta) = \mathcal{N}(y|f_\theta(x), \sigma^2 \mathbb{1}_C)$, where only the mean is modelled by the NN. Classification tasks with C classes are modelled by a categorical distribution $p(y|x, \theta) = \mathrm{Cat}\,(y|\phi(f_\theta(x)))$ with probability vector $\phi(f_\theta(x))$, where ϕ denotes the softmax function.

We will often consider not a single input sample to f_θ but a whole set such as $X = (x_1, \dots, x_n)$. In this case $f_\theta(X) = (f_\theta(x_1)^\intercal, \dots, f_\theta(x_n)^\intercal)^\intercal \in \mathbb{R}^{nC}$ should be read as the concatenation of the outputs. We will frequently use the Jacobian of f_θ w.r.t. its parameter $\theta \in \mathbb{R}^p$ evaluated at the MAP $\hat{\theta}$ defined in (3) below. Given a set X we concatenate the single input Jacobians along the output dimension and use the symbol

$$J_X := (\nabla_\theta f_\theta(x_1)^\intercal, \dots, \nabla_\theta f_\theta(x_n)^\intercal)^\intercal|_{\theta=\hat{\theta}} \in \mathbb{R}^{nC \times p}\,. \tag{2}$$

Bayesian Neural Networks. When taking a Bayesian view on NNs the parameter θ is considered as a random variable equipped with a prior distribution $p(\theta)$. Given the training data $\mathcal{D} = \{(x_i, y_i)|1 \leq i \leq N\}$, the posterior distribution of θ is given by $p(\theta|\mathcal{D}) \propto p(\theta)p(D|\theta) = p(\theta)\prod_{i=1}^N p(y_i|x_i, \theta)$ (with $p(y_i|x_i, \theta)$ as above). A point estimate for θ is then given by the value that is most likely under $p(\theta|\mathcal{D})$, the so-called MAP (*maximum a posteriori*) estimate, that is

$$\hat{\theta} = \arg\min_\theta \mathcal{L}_\theta(\mathcal{D})\,, \tag{3}$$

where we used the (unnormalized) negative log-posterior $\mathcal{L}_\theta(\mathcal{D}) = -\sum_{i=1}^N \ln p(y_i|x_i, \theta) - \ln p(\theta)$. In this work we will use the common choice $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1}\mathbb{1}_p)$ with precision $\lambda > 0$ for which $\mathcal{L}_\theta$ just boils down to the MSE loss (for regression) or cross-entropy loss (for classification) combined with L2 regularization.

Laplace Approximation. With $\mathcal{L}_\theta(\mathcal{D})$ as above the posterior distribution $p(\theta|\mathcal{D})$ reads as $p(\theta|\mathcal{D}) = \frac{1}{Z}p(\mathcal{D}|\theta)p(\theta) =: \frac{1}{Z}e^{-\mathcal{L}_\theta(\mathcal{D})}$ with the normalization constant $Z = \int d\theta\; p(\mathcal{D}|\theta)p(\theta)$. For complex models such

as Bayesian NNs the exact posterior is typically infeasible to compute or sample from. Expanding $\mathcal{L}_\theta(\mathcal{D})$ to second order around the MAP $\hat{\theta}$ from (3), we obtain the *Laplace approximation* of the posterior

$$p(\theta|\mathcal{D}) \simeq \mathcal{N}(\theta|\hat{\theta}, \Psi)$$

with mean $\hat{\theta}$ and covariance $\Psi = \left(\nabla^2_\theta \mathcal{L}_\theta(\mathcal{D})|_{\theta=\hat{\theta}}\right)^{-1} = (NH + \lambda \mathbb{1}_p)^{-1} \in \mathbb{R}^{p\times p}$, where we denote by $H = -\frac{1}{N}\sum_{i=1}^N \nabla^2_\theta \ln p(y_i|\theta, x_i)|_{\theta=\hat{\theta}}$ the Hessian of the averaged negative log-likelihood.

Generalized Gauss-Newton Matrix. The Hessian $H \in \mathbb{R}^{p\times p}$ from above is the second order derivative of $-\frac{1}{N}\ln p(\mathcal{D}|\theta)$ at the MAP $\hat{\theta}$. On the one hand, to compute H is infeasible, and on the other hand, even if H could be computed, it would be impractical, if not impossible, to store the $\frac{p(p+1)}{2}$ free components for typical values of p. In addition, for trained NNs the Hessian does usually not have the nice property of positive semi-definiteness that is found, e.g., in the context of convex problems, because the learned MAP $\hat{\theta}$ is, in general, not a local minimum but rather a saddle point. The difficulties of computational complexity and missing positive definiteness can be overcome by using the generalized Gauss-Newton (GGN) matrix (Schraudolph, 2002) instead of H:

$$H_{\text{GGN}} = \frac{1}{N}\sum_{i=1}^N J_{f_i}^\intercal H_{-\ln p(y_i|f_i)} J_{f_i}\,, \tag{4}$$

where $J_{f_i} = \nabla_\theta f_\theta(x_i)|_{\theta=\hat{\theta}} \in \mathbb{R}^{C\times p}$ and $H_{-\ln p(y_i|f_i)} = -\nabla^2_f \ln p(y_i|f_i)|_{f_i=f_{\hat{\theta}}(x_i)} \in \mathbb{R}^{C\times C}$ is the Hessian of the negative log-likelihood w.r.t. the model output $f_i = f_\theta(x_i)$. H_{GGN} can be interpreted as the Hessian of the linearized model (Martens, 2014; Immer et al., 2021) and is positive semi-definite if all $H_{-\ln p(y|f_i)}$ are positive-semi definite (Schraudolph, 2002), which is the case in our work. More detailed information on H_{GGN} and the relation between H_{GGN} and H is provided in Appendix H. Combining (4) with the term arising from the prior $p(\theta)$ we obtain as precision matrix of the LA $\Psi^{-1}_{\text{GGN}} = \sum_{i=1}^N J_{f_i}^\intercal H_{-\ln p(y_i|f_i)} J_{f_i} + \lambda \mathbb{1}_p$. Where feasible, the LA $p(\theta|\mathcal{D}) \simeq \mathcal{N}(\theta|\hat{\theta}, \Psi_{\text{GGN}})$ will serve as gold standard in this work. We refer to it as the *full LA* and we identify, if not stated otherwise, Ψ with its positive semi-definite estimate Ψ_{GGN}.

Approximations. While the GGN relation (4) consists of objects, J_{f_i} and $H_{-\ln p(y_i|f_i)}$, that are scalable in their computation we usually can't compute H_{GGN} or Ψ^{-1}_{GGN} as the resulting matrices have still too many dimensions for modern NNs. In particular, we can't invert Ψ^{-1}_{GGN} to obtain the posterior covariance Ψ_{GGN}. As a consequence, various approximations have been developed that modify the structure in such a way that it takes less storage and is easier to invert. An easy solution is to only keep the diagonal of Ψ_{GGN}. In the KFAC approximation the Hessian is reduced to a form where it is the Kronecker product of two smaller matrices.

Predictive Distribution. For the posterior distribution $p(\theta|\mathcal{D})$ and a set of n inputs X the posterior predictive distribution is given by

$$p(Y|X, \mathcal{D}) = \int d\theta\; p(Y|X, \theta) p(\theta|\mathcal{D}). \tag{5}$$

Under the LA and using the linearized model (1) for $p(Y|X, \mathcal{D})$ we can give an explicit formula to this distribution for regression problems

$$p(Y|X, \mathcal{D}) \simeq \mathcal{N}(Y|f_{\hat{\theta}}(X), \Sigma_X + \sigma^2 \mathbb{1}_{nC}) \tag{6}$$

$$\text{with} \quad \Sigma_X = J_X \Psi J_X^\intercal \in \mathbb{R}^{nC\times nC} \tag{7}$$

denoting the model uncertainty part of the predictive covariance. For classification tasks the predictive distribution can be approximated, for a single input x_i, by the probit approximation (Bishop, 2007)

$$p(y_i|x_i, \mathcal{D}) \simeq \text{Cat}\left(y_i|\phi\left(\frac{f_{\hat{\theta}}(x_i)}{\sqrt{1 + \frac{\pi}{8}\text{diag}\Sigma_{x_i}}}\right)\right) \tag{8}$$

with $\Sigma_{x_i} \in \mathbb{R}^{C\times C}$ being the corresponding block diagonal element of (7) and ϕ denoting the softmax function. Note that in both cases, regression and classification, the predictive distribution is essentially fixed by Σ_X from (7), which is why this object will be the linchpin of our analysis below. We will call Σ_X the *epistemic predictive covariance*.

4 THE LAPLACE APPROXIMATION FOR SUBSPACE MODELS

Subspace Models. In this work we study, as in (Izmailov et al., 2019), models that are defined on an affine subspace of the parameter space $\mathbb{R}^p$ chosen to contain the MAP $\hat{\theta}$ from (3). That is, we consider a re-parametrization

$$\theta = \hat{\theta} + P\mu\,, \tag{9}$$

where $P \in \mathbb{R}^{p\times s}$ is a matrix that we call, somewhat loosely, the projection matrix (in general it's not related to a mathematical projection) and μ is a new parameter that runs through $\mathbb{R}^s$ where $s \leq p$ is the subspace dimension. The assumption in considering Bayesian inference of NNs in a subspace is that only a fraction of the parameter space is actually needed to represent the (epistemic) uncertainty faithfully. To emphasize

that a Bayesian inference is done under the subspace model (9) we will use in the following p_P instead of p as a symbol for our probability densities.

Note that the selection of a subset of parameters on which to perform inference, as it is done by Daxberger et al. (2021b) and Sharma et al. (2023), is a special case of (9).

Bayesian Inference for μ. To perform Bayesian inference in the subspace model (9), we choose the following prior

$$p_P(\mu) = \mathcal{N}(\mu|0, (\lambda P^\intercal P)^{-1}), \tag{10}$$

where we recall that λ is the precision of $p(\theta)$. Together with the following likelihood

$$p_P(\mathcal{D}|\mu) = p(\mathcal{D}|\hat{\theta} + P\mu), \tag{11}$$

that is induced by (9), the following lemma holds:

Lemma 1. *In the setting above, consider a full rank $P \in \mathbb{R}^{p\times s}$. For the posterior $p_P(\mu|\mathcal{D}) \propto p_P(\mu)p_P(\mathcal{D}|\mu)$ with prior $p_P(\mu)$ as in (10) we have the LA*

$$p_P(\mu|\mathcal{D}) \simeq \mathcal{N}(0, (P^\intercal \Psi^{-1} P)^{-1}). \tag{12}$$

Lemma 1 follows from (11) and (10), we can deduce $-\nabla^2_\mu \ln\left(p_P(\mathcal{D}|\mu)p_P(\mu)\right)|_{\mu=0} = P^\intercal(NH + \lambda \mathbb{1}_p)P = P^\intercal\Psi^{-1}P$.

We will find in Theorem 1 below that the family of posteriors (12) is rich enough to approximate the full LA optimally in a certain sense when a suitable P is chosen. Moreover, in the case where P encodes a subset, (12) coincides with the posterior considered by Daxberger et al. (2021b).

Predictive Distributions of Subspace Models. Similar to (1) we linearize $\tilde{f}_\mu = f_{\hat{\theta}+P\mu}$ around $\mu = 0$ to obtain for a set of n inputs X

$$\tilde{f}_{\mathrm{Lin},\mu}(X) = f_{\hat{\theta}}(X) + J_X P(\mu - 0), \tag{13}$$

where we denoted, as in (1), by $J_X = \nabla_\theta f_\theta(X)|_{\theta=\hat{\theta}} \in \mathbb{R}^{nC\times p}$ the Jacobian of the full network at the MAP.

Combining (12) with (13) we obtain as above for the predictive distribution $p_P(Y|X,\mathcal{D})$

$$\begin{gathered}\mathcal{N}(Y|f_{\hat{\theta}}(X), \Sigma_{P,X} + \sigma^2 \mathbb{1}_{nC}) \text{ or} \\ \mathrm{Cat}\left(y_i|\phi\left(\frac{f_{\hat{\theta}}(x_i)}{\sqrt{1 + \frac{\pi}{8}\mathrm{diag}\Sigma_{P,x_i}}}\right)\right),\end{gathered} \tag{14}$$

for regression and classification respectively, with the notation

$$\Sigma_{P,X} = J_X P(P^\intercal\Psi^{-1}P)^{-1}P^\intercal J_X^\intercal \in \mathbb{R}^{nC\times nC} \tag{15}$$

for its epistemic predictive covariance. In (14) we use $\Sigma_{P,x_i} \in \mathbb{R}^{C\times C}$ to denote the block diagonal elements of $\Sigma_{P,X} \in \mathbb{R}^{nC\times nC}$ corresponding to the single inputs x_i.

4.1 The Subspace LA Closest to the Full LA

Consider a set of n inputs $X = (x_1, \dots, x_n)$. As already noted above, the predictive distribution $p(Y|X,\mathcal{D})$ of the full model and $p_P(Y|X,\mathcal{D})$ of the subspace model only differ by their epistemic predictive covariances Σ_P and $\Sigma_{P,X}$. Given X and a fixed subspace dimension $s \leq p$, we might therefore consider $P^* \in \mathbb{R}^{p\times s}$ optimal if it solves the following minimization problem.

$$P^* \in \operatorname*{arg\,min}_{P\in\mathbb{R}^{p\times s},\,\mathrm{rank}\,P=s} \|\Sigma_{P,X} - \Sigma_X\|_F, \tag{16}$$

where $\|\dots\|_F$ denotes the Frobenius norm. A solution to (16) is never unique. In fact, for any P^* that solves (16) we can also consider P^*Q for an arbitrary invertible $Q \in \mathbb{R}^{s\times s}$ since we have $\Sigma_{P^*Q,X} = \Sigma_{P^*,X}$, cf. (15). Note that such a change of P^* corresponds to a reparameterization $\mu \to Q^{-1}\mu$. We will prove below, however, that up to such reparametrizations the solution to (16) is unique.

For the solution of the problem (16) we will need the eigenvalue decomposition $\Sigma_X = J_X\Psi J_X = U\Lambda U^\intercal$ where U is an orthogonal matrix and $\Lambda \in \mathbb{R}^{nC\times nC}$ is a positive semi-definite diagonal matrix. We choose this eigendecomposition such that the diagonal entries of Λ are decreasing. We will use the Eckart-Young-Mirsky-Theorem (Schmidt, 1907; Eckart and Young, 1936; Mirsky, 1960) which states that the following low rank problem has an explicit solution

$$U_s\Lambda_s U_s^\intercal \in \operatorname*{arg\,min}_{A\in\mathbb{R}^{nC\times nC}:\,\mathrm{rank}\,A\leq s} \|A - \Sigma_X\|_F, \tag{17}$$

where $U_s \in \mathbb{R}^{nC\times s}$ contains the first s eigenvectors, called dominant eigenvectors from now on, and $\Lambda_s \in \mathbb{R}^{s\times s}$ is the reduced diagonal matrix obtained by taking the upper $s\times s$ block containing the s leading eigenvalues of Σ_X. While solving the low rank problem (17) is standard, solving (18) for P^* is not. In the following theorem we give an explicit expression to P^* and show that its epistemic predictive covariance solves (17).

Theorem 1 (Existence and uniqueness for (16))**.** *Consider the problem (16) with $s \leq s_{\max} = \min(nC, p)$. Suppose that $J_X \in \mathbb{R}^{nC\times p}$ has full rank. For any invertible $Q \in \mathbb{R}^{s\times s}$ the matrix*

$$P^* = \Psi J_X^\intercal U_s Q \tag{18}$$

solves (16). For any such P^* we have $\Sigma_{P^*,X} = U_s\Lambda_s U_s^\intercal$. If the diagonal elements of Λ satisfy $\sigma_s > \sigma_{s+1}$ then the solution of the minimization problem (16) is unique up to reparametrization: any minimizer of (16) is of the form (18).

The proof of Theorem 1 is provided in Appendix B. The condition $\sigma_s > \sigma_{s+1}$ on the singular values of

Algorithm 1 Subspace Construction

Require: Trained $f_{\hat{\theta}}$, dimension s, subset size n
Ensure: Projection matrix $P \in \mathbb{R}^{p \times s}$
1: Compute $\Psi_{\text{approx}} \in \{\text{KFAC, Diagonal}\}$
2: Sample $X' \subset \mathcal{D}$ with $|X'| = n$
3: Compute $J_{X'} = \nabla_\theta f_\theta(X')|_{\theta=\hat{\theta}}$
4: Compute $M = J_{X'} \Psi_{\text{approx}} J_{X'}^\top \in \mathbb{R}^{nC \times nC}$
5: SVD: $M = U \Lambda U^\top$, extract top s eigenvectors U_s
6: **Return** $P = \Psi_{\text{approx}} J_{X'}^\top U_s$

Algorithm 2 Predictive Covariance Computation

Require: Projection P (from Algorithm 1), test inputs X
Ensure: Epistemic predictive covariance $\Sigma_{P,X}$
1: Compute $J_X = \nabla_\theta f_\theta(X)|_{\theta=\hat{\theta}}$ (batchwise)
2: Compute $V^\top P$ where $\Psi_{\text{GGN}} = VV^\top$ (batchwise)
3: Compute $(P^\top \Psi_{\text{GGN}} P)^{-1} \in \mathbb{R}^{s \times s}$
4: **Return** $\Sigma_{P,X} = J_X P (P^\top \Psi_{\text{GGN}} P)^{-1} P^\top J_X^\top$

Σ_X is typically true in practice since "noisy real-world" matrices almost surely have distinct singular values (Hartfiel, 1995). The restriction to dimensions below $s_{\max} = \min(nC, p)$ and the assumption on the full rank of J_X is needed to assure that P^* has full rank which is required for $\Sigma_{P^*,X}$ in order to be well-defined. If J_X doesn't have full rank, we restrict ourselves to s below the rank of the Jacobian. This is the case for some regression problems in Section 5. In Appendix B.1 we discuss some additional consequences of Theorem 1.

4.2 A Feasible Approximation

Theorem 1 states that there is an optimal solution to problem (16) and it is, to the best knowledge of the authors, the first systematic solution to a subspace modelling for Bayesian NNs in the context of LA. However, the applicability of Theorem 1 is limited in practice, due to the following two reasons:

Epistemic Limitation. Training datasets $\mathcal{D}$ are often so large that computing the eigendecomposition of Σ_X with $X = \mathcal{D}$, and thus of U_s, is infeasible. However, even if we can pick $X = \mathcal{D}$ we actually want the subspace model to work for unseen data points, that is data points that are not contained in $\mathcal{D}$.

No Access to Ψ. The posterior covariance $\Psi = \Psi_{\text{GGN}}$ from the full LA is in general not feasible.

In practice, we will therefore use the following workflow:

1. Fix an approximation Ψ_{approx} to Ψ such as the KFAC or diagonal approximation.
2. Use a subset X' of size n of the inputs in the training set to construct $J_{X'} \Psi_{\text{approx}} J_{X'}^\intercal \in \mathbb{R}^{nC \times nC}$ and determine its s dominant eigenvectors $U_s \in \mathbb{R}^{nC \times s}$.
3. Construct P via $P = \Psi_{\text{approx}} J_{X'}^\intercal U_s$ (we will in this work fix Q to be always the identity).
4. For the X of interest (usually not contained in the training set), compute the predictive covariance $J_X P (P^\intercal \Psi P)^{-1} P^\intercal J_X^\intercal$. Note that we can really use the GGN Ψ here, since $\Psi_{\text{GGN}} = VV^\intercal$ can be written as an outer product which allows for a batch-wise computation (see Appendix H). In our experiments, we will use the full GGN for the dimensionally reduced posterior for all projectors (subset and lowrank), and we use X of size n that are randomly drawn from the test data.

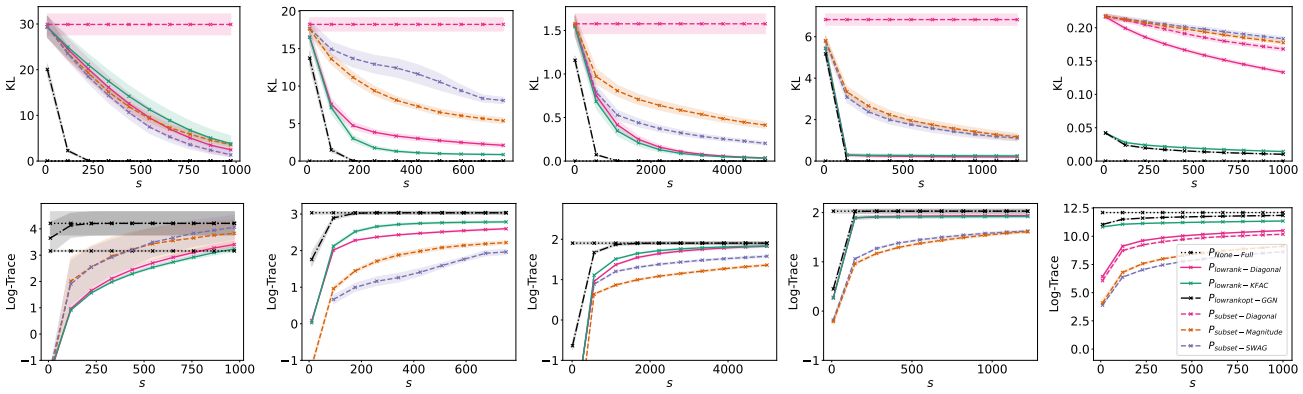
The first three steps outline the construction of our projection P. These are summarized in Algorithm 1. And the last step, is presented in Algorithm 2. Algorithm 2 is used for methods based on lowrank approximations as well as subset methods.

Limitations. As the subspace construction in Section 4.2 deviates due to $X' \neq X$ and $\Psi_{\text{approx}} \neq \Psi$ from the setting in Theorem 1 it should only be considered as an approximation. Computational bottlenecks of this algorithm are discussed in Appendix A.3. The main bottlenecks arise from s via the need to store $P \in \mathbb{R}^{p \times s}$, which is unavoidable in an approach with general affine relations, and from nC via the need of computing a $nC \times nC$ matrix and performing a SVD on it. Due to the last restriction only a subset of ImageNet was considered in Section 5. A transition to randomized algorithms or using suitable submatrices might lift such a bottleneck but was not studied in this work.

5 EXPERIMENTS

We use various `OpenML` (Vanschoren et al., 2013; Feurer et al., 2019) regression datasets as well as common classification tasks such as FashionMNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky, 2009) and a subset of ImageNet (Deng et al., 2009), called ImageNet10, that contains the ten classes listed in Appendix A, to validate different subspace methods. In addition, we study the performance of the methods on out-of-distribution (OOD) data: We chose a corrupted version of CIFAR10 (Hendrycks and Dietterich, 2019) for this analysis and provide some additional results on corrupted MNIST (Mu and Gilmer, 2019) in Appendix E.2. Details about the used NNs can be found in Appendix A. We compare the following LAs:

- The projectors $P_{\text{subset}-\text{Magnitude}}, P_{\text{subset}-\text{Diagonal}}$ and $P_{\text{subset}-\text{SWAG}}$ (dashed lines) select a subset of



(a) ENB (b) Red Wine (c) California (d) Naval Propulsion (e) Fashion-MNIST

Figure 1: Comparison of subspace models for different datasets and dimensions s. Different choices of P are marked by different colours and line types (cf. the legend in lower rightmost plot). The first row displays the KL-divergence (19) and the second the logarithm of the trace (20). Missing values in the logarithm of trace plots have a trace of zero at these values of s (e.g. $P_{\text{subset}-\text{Diagonal}}$ for all regression datasets and all considered s or $P_{\text{subset}-\text{SWAG}}$ for the lowest s in Red Wine).

parameters according to the magnitude of parameters, the diagonal GGN approximation or variances produced via SWAG. We use the term *subset methods* for these approximations from Daxberger et al. (2021b) and Daxberger et al. (2021a) because they select certain parameters to construct P.

- $P_{\text{lowrank}-\text{KFAC}}$ and $P_{\text{lowrank}-\text{Diagonal}}$ (solid lines) are constructed as in Section 4.2 and use a KFAC or a diagonal GGN approximation to estimate Ψ in Step 2 of Section 4.2. We use the term *low rank methods* for these since Theorem 1 bases its argument on a low rank approximation. A subset of the training data was used for the construction of these subspace models, cf. Appendix A.3.
- Moreover, where feasible, we show results for a $P_{\text{lowrankopt}-\text{GGN}}$ (dashed-dotted line) that is exactly constructed as in Theorem 1 by using the *test* data and Ψ_{GGN} for the construction of the subspace model. This is the subspace model that minimizes (16). $P_{\text{None}-\text{Full}} = \mathbb{1}_p$ (dotted line) is the full LA without any dimensional reduction.

All experiments are done with five different seeds and the average of the results is plotted with markers. To enhance the visualization, the markers are in some plots linearly interpolated by lines whose type indicates the methods used to approximate the LA. We further show the sample standard error via shaded areas or error bars. All plots use the same colour coding.

5.1 Faithfulness of Subspace Approximation

First, we study whether the subspace methods provide a faithful approximation of the full LA. As the posterior predictive distribution (5) is the object of genuine interest for predictions via Bayesian NNs, we assess whether the distributions in (14) are similar to the distributions (6), (8) obtained by the full LA. We consider two cases: One in which the actual distributions (6), (8) are known and the other in which they are unknown.

Evaluation of P When Full Model Is Known. We would like to find a subspace model (9) whose LA closely aligns with the full LA. To assess this, we use the Kullback-Leibler divergence between (6) (for regression) or (8) (for classification) and the according distribution from (14):

$$D_{\text{KL}}(p(Y|X,\mathcal{D})\|p_P(Y|X,\mathcal{D}))\,. \tag{19}$$

Since the probit approximation (8) only holds for single inputs, we use for classification the "block diagonal approximations" $p(Y|X,\mathcal{D}) \simeq \prod_{i=1}^N p(y_i|x_i,\Sigma_{x_i})$ and $p_P(Y|X,\mathcal{D}) \simeq \prod_{i=1}^N p_P(y_i|x_i,\Sigma_{P,x_i})$.

Evaluation of P When Full Model Is Unknown. The KL-divergence (19) quantifies the deviation of the subspace model to the full LA. As the latter is in general not feasible, we also consider the logarithm of the trace of the epistemic predictive covariances as a proxy. The predictive distributions for full and subspace models are essentially fixed by their covariances with identity between both distributions for the case $\Sigma_{P,X} = \Sigma_X$. Heuristically, $\Sigma_{P,X}$ approximates better Σ_X if it contains the dominant eigenspace, because in the directions of these eigenvectors the covariance has its largest contributions. Hence, we propose to use the following *trace criterion*: If

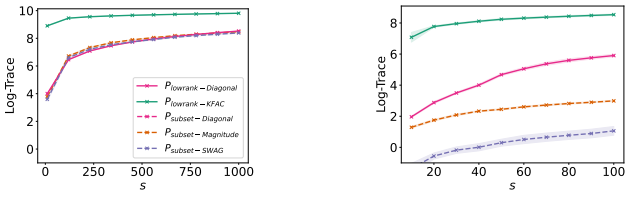
$$0 \leq \operatorname{Tr}\Sigma_{P_1,X} < \operatorname{Tr}\Sigma_{P_2,X} \leq \operatorname{Tr}\Sigma_X \tag{20}$$

metric	dataset	s	lowrank-D.	lowrank-KFAC	subset-D.	subset-M.	subset-SWAG	$\hat{\theta}$
NLL	FashionMNIST	120	0.289	**0.248**	0.292	0.292	0.292	0.293
		450	0.282	**0.249**	0.290	0.288	0.288	0.293
	CIFAR10	120	0.287	**0.263**	0.289	0.287	0.287	0.289
		450	0.281	**0.259**	0.289	0.282	0.283	0.289
	ImageNet10	50	0.274	**0.270**	0.277	0.276	0.277	0.277
		100	**0.271**	0.277	0.277	0.276	0.276	0.277
ECE	FashionMNIST	120	0.039	**0.010**	0.040	0.040	0.040	0.041
		450	0.037	**0.013**	0.040	0.039	0.039	0.041
	CIFAR10	120	0.037	**0.029**	0.038	0.037	0.037	0.038
		450	0.035	**0.027**	0.038	0.036	0.036	0.038
	ImageNet10	50	**0.039**	0.041	0.040	0.040	0.040	0.040
		100	**0.038**	0.048	0.040	0.039	0.040	0.040
Brier	FashionMNIST	120	0.135	**0.128**	0.135	0.135	0.135	0.135
		450	0.134	**0.129**	0.135	0.135	0.135	0.135
	CIFAR10	120	0.125	**0.123**	0.126	0.125	0.125	0.126
		450	0.125	**0.122**	0.126	0.125	0.125	0.126
	ImageNet10	50	0.129	**0.127**	0.129	0.129	0.129	0.129
		100	0.128	**0.127**	0.129	0.129	0.129	0.129

Table 1: Evaluation of the uncertainty quantification produced of various methods for subspace inference using the NLL, ECE metric and the Brier score for the subspace inference methods studied in his article. The last column shows the performance of the model with parameter $\hat{\theta}$ from (3).

holds, P_2 is a better projector than P_1. A larger trace value indicates that the more dominant eigenspace is captured for a given P. The proof of $\operatorname{Tr} \Sigma_{P,X} \leq \operatorname{Tr} \Sigma_X$ and an extended explanation are given in Appendix C. In Appendix B.1 we show that the P^* from Theorem 1 is also optimal under the trace criterion. Empirically, we validate the trace criterion on cases in which we can compute (6), (8) exactly.

Datasets With Feasible Baseline. Figure 1 shows the KL-divergence 19 and the logarithm of the trace criterion (20) for several datasets and the subspace models listed above for different s. All datasets in Figure 1 are such that the full LA and the baseline $P_{\text{lowrankopt}-\text{GGN}}$ are feasible. For ENB, Red Wine and Naval Propulsion the Jacobian is rank-deficient, so that only s up to the rank of the Jacobian on the training data are considered. First, we observe for all datasets that the baseline $P_{\text{lowrankopt}-\text{GGN}}$ (black dashed-dotted line) needs only a fraction of the total number of model parameters, listed in Table 6, to reach a small KL-divergence with the full model. Hence, subspace models can indeed be suitable for quantifying the uncertainty provided by a LA. However, $P_{\text{lowrankopt}-\text{GGN}}$ is usually unknown such that the ideal approximation isn't available. Comparing the feasible approximations in Figure 1 we find that low rank approximations demonstrate superior approximations compared to subset methods in general. In particular, the performance of $P_{\text{lowrank}-\text{Diagonal}}$ is stricly superior to $P_{\text{subset}-\text{Diagonal}}$. Only for ENB some subset methods obtain a slightly better performance. We speculate that the different performance on this dataset is related to the number of 'dead parameters' (with gradient almost zero), whose complement provides a natural subset to be selected. Indeed, ENB has the most number of dead parameters with 93%. More details on this investigation are given in Appendix G.



(a) CIFAR10 (b) ImageNet10

Figure 2: Evaluation the trace criterion (20) for CIFAR10 and ImageNet10 and different choices of P. The colour coding and linestyles are chosen as in Figure 1.

Datasets Without Feasible Baseline. A comparison between the first and second row of Figure 1 demonstrates that the log-trace retains the ordering of the KL-divergence. Differences are rare, and if they occur, they are small and usually contained in the sample standard deviation. Thus, we use the log-trace to assess the approximation quality of the full LA for ResNet9 and ResNet18 applied on CIFAR10 and ImageNet10, respectively. For both networks the number of parameters is so large that the computation of the full LA is infeasible. Both experiments indicate again a better performance of the low-rank based methods, especially of $P_{\text{lowrank}-\text{KFAC}}$: As revealed by the trace criterion in Figure 2 the eigenspace of $\Sigma_{P,X}$ spanned by the selected eigenvectors in parameter space is orders of

metric	corruption	lowrank-D.	lowrank-KFAC	subset-D.	subset-M.	subset-SWAG	$\hat{\theta}$
NLL	brightness	0.340	**0.313**	0.344	0.340	0.340	0.344
	elastic transform	0.522	**0.490**	0.527	0.522	0.522	0.527
	gaussian blur	0.353	**0.341**	0.354	0.352	0.353	0.354
	impulse noise	1.822	**1.660**	1.861	1.836	1.831	1.861
ECE	brightness	0.044	**0.035**	0.046	0.044	0.045	0.046
	elastic transform	0.063	**0.050**	0.065	0.064	0.064	0.065
	gaussian blur	0.031	**0.022**	0.032	0.031	0.031	0.032
	impulse noise	0.227	**0.203**	0.233	0.229	0.229	0.233
Brier	brightness	0.149	**0.145**	0.149	0.149	0.149	0.149
	elastic transform	0.230	**0.225**	0.231	0.230	0.230	0.231
	gaussian blur	0.165	**0.164**	0.166	0.165	0.165	0.166
	impulse noise	0.612	**0.595**	0.616	0.614	0.613	0.616

Table 2: Performance of various subspace inference methods on corrupted versions of CIFAR10 for $s = 200$. The method "subset-Diagonal" was omitted here as it yielded results indistinguishable from the model at the MAP $\hat{\theta}$ due to a negligible uncertainty.

magnitude higher for $P = P_{\text{lowrank}-\text{KFAC}}$ than for all other considered methods.

5.2 Uncertainty Quantification

To assess the uncertainty quantification of the studied methods, we study the common "negative log-likelihood" (NLL) criterion, which is the average of the values $-\ln p_P(y^*|x^*, \mathcal{D})$ for all pairs (x^*, y^*) from the test data. For classification, we also include the expected calibration error (ECE), which measures the discrepancy between predicted confidence and empirical accuracy across probability bins, and the Brier score with the convention outlined in Appendix D.

For the regression problems studied in this article the NLL values are essentially identical within their standard errors across methods, cf. Appendix D. Results for the classification datasets considered in this work are contained in Table 1 and also compared to the performance of the model at the MAP (3). All three metrics, NLL, ECE and Brier score, indicate a better performance of the lowrank methods.

5.3 Performance on OOD Data

OOD data analysis reveals how well different BNNs approximation methods maintain meaningful uncertainty quantification when encountering inputs that differ from the training distribution. We trained a NN on the original CIFAR10 (Krizhevsky, 2009) dataset and applied it to corrupted versions from Hendrycks and Dietterich (2019). The results are shown in Table 2 and show a better performance of lowrank methods, especially of $P_{\text{lowrank}-\text{KFAC}}$. In Appendix E.2 we also provide results for corrupted versions of MNIST from Mu and Gilmer (2019).

6 CONCLUSION

In this work we propose to look at subspace Laplace approximations of Bayesian neural networks through the lens of their predictive covariances. Using low rank techniques this approach allows us to derive an explicit relation for a subspace model that would be optimal given a set of datapoints. This subspace model can be used, where feasible, as a baseline for subspace inference.

We propose a construction of a subspace model that is scalable but still conceptually based on our theoretical insights. Starting with a preliminary estimate of the predictive covariance we can use the derived relation to get an estimate of relevant directions in parameter space. In a second step these directions can be combined with the generalized Gauss-Newton (GGN) Laplace approximation in such a way that they do not require the computation of the, typically infeasible, GGN matrix.

Studying the performance of approximating the full Laplace approximation and of the resulting uncertainty quantification, we find that this construction outperforms existing methods with respect to various metrics and is in some cases even close to the baseline constructed via the test data. Moreover, we observe that a well chosen method for subspace construction can often have more impact on the performance than an increase in the subspace dimension.

Acknowledgements

The research leading to this work was funded by the "Metrology for Artificial Intelligence in Medicine" (M4AIM) program in the frame of the QI-Digital initiative.

The authors acknowledge with gratitude the contributions of their colleague, Clemens Elster, who took part in discussions leading to this article but passed away before its completion. We will deeply miss his scientific contributions and the discussions with him as well as his personal support and advice.

References

Andrade, D. and Sato, K. (2024). On the effectiveness of partially deterministic bayesian neural networks. *Computational Statistics*.

Bishop, C. M. (2007). *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR.

Botev, A., Ritter, H., and Barber, D. (2017). Practical Gauss-Newton optimisation for deep learning. In *International Conference on Machine Learning*.

Calvo-Ordonez, S., Meunier, M., Piatti, F., and Shi, Y. (2024). Partially stochastic infinitely deep Bayesian neural networks. *arxiv preprint: 2402.03495*.

Cheng, Y., Wang, D., Zhou, P., and Tao, Z. (2017). A survey of model compression and acceleration for deep neural networks. *ArXiv*, abs/1710.09282.

Dauphin, Y. N., Pascanu, R., Gülçehre, Ç., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *arXiv preprint: 1406.2572*.

Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., and Hennig, P. (2021a). Laplace redux - effortless Bayesian deep learning. In *Neural Information Processing Systems*.

Daxberger, E., Nalisnick, E., Allingham, J. U., Antorán, J., and Hernández-Lobato, J. M. (2021b). Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, pages 2510–2521. PMLR.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Deng, Z., Zhou, F., and Zhu, J. (2022). Accelerated linearized Laplace approximation for Bayesian deep learning. *ArXiv*, abs/2210.12642.

Dold, D., Rügamer, D., Sick, B., and Dürr, O. (2024). Bayesian semi-structured subspace inference. In *International Conference on Artificial Intelligence and Statistics*, pages 1819–1827. PMLR.

Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

Feurer, M., van Rijn, J. N., Kadra, A., Gijsbers, P., Mallik, N., Ravi, S., Mueller, A., Vanschoren, J., and Hutter, F. (2019). OpenML-Python: an extensible Python API for OpenML. *arXiv*, 1911.02490.

Foong, A. Y. K., Li, Y., Hernández-Lobato, J. M., and Turner, R. E. (2019). 'in-between' uncertainty in Bayesian neural networks. *arXiv preprint: 1906.11537*.

Gal, Y. (2016). Uncertainty in deep learning.

Hardy, G., Littlewood, J., and Pólya, G. (1988). *Inequalities*. Cambridge Mathematical Library. Cambridge University Press.

Hartfiel, D. J. (1995). Dense sets of diagonalizable matrices. *Proceedings of the American Mathematical Society*, 123(6):1669–1672.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

Hendrycks, D. and Dietterich, T. G. (2019). Benchmarking neural network robustness to common corruptions and perturbations.

Hernández-Lobato, J. M. and Adams, R. (2015). Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR.

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

Horn, R. A. and Johnson, C. R. (2012). *Matrix analysis*. Cambridge university press.

Immer, A., Korzepa, M., and Bauer, M. (2021). Improving predictions of Bayesian neural nets via local linearization. *arXiv preprint: 2008.08400*.

Izmailov, P., Maddox, W. J., Kirichenko, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. (2019). Subspace inference for Bayesian deep learning. *arxiv preprint: 1907.07504*.

Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. G. (2021). What are bayesian neural network posteriors really like? In Meila, M. and

Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4629–4640. PMLR.

Jordan, M. I., Ghahramani, Z., Jaakkola, T., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233.

Kendall, A. and Gal, Y. (2017). What uncertainties do we need in Bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*.

Kingma, D. P., Salimans, T., and Welling, M. (2015). Variational dropout and the local reparameterization trick. *arXiv preprint arXiv:1506.02557*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Kristiadi, A., Hein, M., and Hennig, P. (2020). Being Bayesian, even just a bit, fixes overconfidence in relu networks. *arXiv preprint: 2002.10118*.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324.

LeCun, Y., Denker, J., and Solla, S. (1989). Optimal brain damage. *Advances in neural information processing systems*, 2.

Lee, J., Humt, M., Feng, J., and Triebel, R. (2020). Estimating model uncertainty of neural networks in sparse information form. *ArXiv*, abs/2006.11631.

MacKay, D. J. C. (1992a). The evidence framework applied to classification networks. *Neural Computation*, 4:720–736.

MacKay, D. J. C. (1992b). A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4:448–472.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. (2019). A simple baseline for Bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32.

Marshall, A. W., Olkin, I., and Arnold, B. C. (1979). Inequalities: theory of majorization and its applications.

Martens, J. (2014). New insights and perspectives on the natural gradient method. *arxiv preprint: 1412.1193*.

Martens, J. and Grosse, R. B. (2015). Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning*.

Mirsky, L. (1960). Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11(1):50–59.

Mu, N. and Gilmer, J. (2019). Mnist-c: A robustness benchmark for computer vision.

Ortega, L. A., Santana, S. R., and Hern'andez-Lobato, D. (2023). Variational linearized Laplace approximation for Bayesian deep learning. *ArXiv*, abs/2302.12565.

Papyan, V. (2018). The full spectrum of deepnet Hessians at scale: Dynamics with sgd training and sample size. *arXiv preprint: 1811.07062*.

Pascanu, R. and Bengio, Y. (2013). Revisiting natural gradient for deep networks. *arxiv preprint: 1301.3584*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library.

Ritter, H., Botev, A., and Barber, D. (2018). A scalable Laplace approximation for neural networks. In *International Conference on Learning Representations*.

Sagun, L., Bottou, L., and LeCun, Y. (2016). Eigenvalues of the Hessian in deep learning: Singularity and beyond. *arXiv preprint: 1611.07476*.

Salimans, T. and Kingma, D. P. (2016). Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *arXiv preprint: 1602.07868*.

Schmidt, E. (1907). Zur Theorie der linearen und nichtlinearen Integralgleichungen. *Mathematische Annalen*, 63(4):433–476.

Schraudolph, N. N. (2002). Fast Curvature Matrix-Vector Products for Second-Order Gradient Descent. *Neural Computation*, 14(7):1723–1738.

Sharma, M., Farquhar, S., Nalisnick, E., and Rainforth, T. (2023). Do Bayesian neural networks need to be fully stochastic? *arxiv preprint: 2211.06291*.

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M. M. A., Prabhat, and Adams, R. P. (2015). Scalable Bayesian optimization using deep neural networks. *arXiv preprint: 1502.05700*.

Székely, G. J. and Bakirov, N. K. (2003). Extremal probabilities for gaussian quadratic forms. *Probability theory and related fields*, 126(2):184–202.

Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. (2013). Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60.

Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1:1–305.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *ArXiv*, abs/1708.07747.

Yang, A. X., Robeyns, M., Wang, X., and Aitchison, L. (2024). Bayesian low-rank adaptation for large language models. In *The Twelfth International Conference on Learning Representations*.

Zhao, Z., Mair, S., Schön, T. B., and Sjölund, J. (2023). On feynman-kac training of partial bayesian neural networks. In *International Conference on Artificial Intelligence and Statistics*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and model. *Yes, cf. Section 3, 4 and Appendix A.*
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. *Yes, we included a section on computational costs, cf. Section A.5 and other computational aspects are discussed in Section 4.2 and A.3.*
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. *Yes, the source code is included in the supplementary material.*

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. *Yes, cf. Section 3 - 4.*
   (b) Complete proofs of all theoretical results. *Yes, cf. Appendix B.*
   (c) Clear explanations of any assumptions. *Yes, cf. Section 3 - 4.*

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). *Yes, source code for this will be part of the supplementary material.*
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). *Yes, cf. Section A and the source code in the supplementary material.*
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). *Yes, cf. Section 5.*
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). *Yes, cf. Section A.5.*

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. *Yes.*
   (b) The license information of the assets, if applicable. *Not Applicable.*
   (c) New assets either in the supplemental material or as a URL, if applicable. *Not Applicable.*
   (d) Information about consent from data providers/curators. *Not Applicable.*
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. *Not Applicable.*

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. *Not Applicable.*
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. *Not Applicable.*
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. *Not Applicable.*

Low Rank Based Subspace Inference for the Laplace Approximation of Bayesian Neural Networks: Supplementary Materials

A EXPERIMENTS

A.1 Architectures and Training

The code of all experiments was developed in `PyTorch` Paszke et al. (2019). The subset methods were implemented using the `laplace` library Daxberger et al. (2021a). The regression datasets were obtained from `OpenML` Vanschoren et al. (2013); Feurer et al. (2019).

For the regression problems the expected mean $E_{y\sim p(y|x,\theta)}[y]$ is estimated by multi-layer perceptrons (MLPs) with ReLU activation functions and two hidden layers that include 128 units in each layer. The bias term is used as well. A full batch training is performed in each epoch, i.e. the batch size equals the size of the training set. The input data is normalized with respect to its mean value and its standard deviation. For Naval Propulsion the output data is also normalized in the same manner. Further details of the architecture and training procedure are given in Table 3 and its caption.

Table 3: The architecture of all networks trained on regression datasets is an MLP with two hidden layers with 128 neurons each. After each hidden layer a ReLU is used. The models are trained on n_{epoch} epochs with learning rate $\alpha = \alpha_{\text{init}} \frac{b}{256}$ (b is the batch size which here equals the number of training data points). For the fraction of epochs 'warm up' the learning rate is linearly increased to α and starting from the fraction of epochs 'decay' the learning rate is linearly decreased.

dataset	α_{init}	n_{epoch}	warm up/ decay
ENB	0.004	1500	(0.1/0.5)
Red Wine	0.0004	300	(0.3/0.3)
California	0.0004	100	(0.3/0.5)
Naval Propulsion	0.0004	100	(0.3/0.5)

The architecture used for corrupted MNIST Mu and Gilmer (2019), trained on MNIST LeCun et al. (1998), and for FashionMNIST Xiao et al. (2017) is a small hand-designed convolutional network (CNN) with 2d-convolutions, max-pooling, batch normalization and ReLU activation function. Before the softmax function a linear layer is applied. The exact architecture can be found in the code linked to this article. To train the CNNs, the input data is mapped to the interval $[0, 1]$ and then normalized with "mean" and "standard deviation" 0.5. Additional details are given in Table 4 and its caption.

Table 4: The models are trained on n_{epoch} epochs with learning rate α and batch size $b = 256$. For the fraction of epochs 'warm up' the learning rate is linearely increased to α and starting from the fraction of epochs 'decay' the learning rate is linearly decreased.

dataset	α	n_{epoch}	warm up/ decay
MNIST	0.004	20	(0.1/0.3)
FashionMNIST	0.002	40	(0.1/0.5)

CIFAR10 Krizhevsky (2009) and ImageNet10 Deng et al. (2009) are classified by ResNet architectures He et al. (2015). CIFAR10 is trained from scratch with ResNet9, but for ImageNet10 the pretrained ResNet18 from

`Pytorch` with weights `IMAGENET1K_V1` is used as initialization, where the last layer is replaced by a linear layer with 10 classes. During training the images are normalized with respect to their channelwise pixel mean and pixel standard deviation. In addition, random flips are applied on both datasets. For CIFAR10 greyscale and random crops are used, too. More information is provided in Table 5 and its caption.

Table 5: The models are trained on n_{epoch} epochs with learning rate α and batch size $b = 256$. For the fraction of epochs 'warm up' the learning rate is linearly increased to α and starting from the fraction of epochs 'decay' the learning rate is linearly decreased.

dataset	α	n_{epoch}	warm up/ decay
CIFAR10	0.004	100	(0.1/0.7)
ImageNet10	0.0004	10	(0.5/0.5)

To evaluate the quality of the dimensional reduction, the size of the different models that are used for predictions are required. Table 6 lists the number of model parameters. The number of model parameters of the MLP and CNN has been chosen large enough such that the prediction performance is satisfying, but is also limited to be able to compute $P_{\text{lowrankopt}-\text{GGN}}$ and the full LA.

Table 6: Number of trainable parameters p for each model trained on the corresponding dataset, and relative values for $s = 100$ and $s = 500$.

dataset	model	p	$s = 100$	$s = 500$
ENB	MLP	17,922	0.56%	2.79%
Red Wine	MLP	18,177	0.55%	2.75%
California	MLP	17,793	0.56%	2.81%
Naval	MLP	18,690	0.53%	2.68%
MNIST	CNN	12,458	0.80%	4.01%
FashionMNIST	CNN	12,458	0.80%	4.01%
CIFAR10	ResNet9	668,234	0.01%	0.07%
ImageNet10	ResNet18	11,181,642	$< 0.01\%$	–

A.2 ImageNet10 Classes

ImageNet10 is a proper subset of ImageNet Deng et al. (2009). The selection of classes used for ImageNet10 is given in Table 7.

Table 7: These ten labels are selected from ImageNet to construct ImageNet10.

label	motifs
n01968897	pearly nautilus, nautilus, chambered nautilus
n01770081	harvestman, daddy longlegs, Phalangium opilio
n01496331	crampfish, numbfish, torpedo, electric ray
n01537544	indigo bunting, indigo finch, indigo bird, Passerina cyanea
n01818515	macaw
n02011460	bittern
n01847000	drake
n01687978	agama
n01740131	night snake, Hypsiglena torquata
n01491361	tiger shark, Galeocerdo cuvieri

A.3 Extended Discussion on the Computational Bottlenecks of our Method

For the low rank methods we construct P as described in Section 4.2 as

$$P = \Psi_{\text{approx}} J_{X'}^{\intercal} U_s \,. \tag{21}$$

All three objects in (21), Ψ_{approx}, $J_{X'}$ and U_s, are constructed from the training data. While we can take the full training data for the construction of Ψ_{approx}, both, $J_{X'}$ and U_s, are constructed from a subset X' of size n of the training data. Ideally, we would of course like to take X' to be full training data. However, doing so presents us with two difficulties:

1. The object U_s needs to be computable.

2. The computation of the product $J_{X'}^{\intercal} U_s$ needs to be feasible.

Obstacle 2 is rather straightforward to circumvent as we can compute the matrix product via mini-batches from the training data. It turns out that Obstacle 1 sets the actual limit on the subset of training data as we compute U_s via an SVD of the object $J_{X'} \Psi_{\text{approx}} J_{X'}^{\intercal} \in \mathbb{R}^{nC \times nC}$. For Red Wine and Naval we pick $n = 1000$. For ENB, the training set has only 514 data points which is why the entire training dataset is considered. For California we can analyze the subspace models until $s = 5000$ as the Jacobian of the model has full rank. To allow for this analysis we choose $n = 5000$. For the classification problems, i.e. MNIST, FashionMNIST, CIFAR10 and ImageNet10, we pick $n = 100$ so that we have $nC = 1000$ for these datasets. This choice allows for a substantially faster computation of P. Our methods demand the explicit storage of P, which limits the the maximum value of s from a computational perspective.

For our experiments, we choose to use the same size n for X' (from the training data) and X (from the test data). This is done only for convenience and is by no means a necessary assumption for the workflow from Section 4.2.

The explicit computation and storage of the projector is a limitation of methods based on linear affine transformation. For large neural networks, this is a bottleneck, because the memory complexity grows with at least $\mathcal{O}\left(s(p+s)\right)$. The subset methods of Daxberger et al. (2021b), by contrast, use heuristics to avoid the explicit construction of P. For example, they can directly construct a subspace by selecting the parameters with the highest weights. In this case, the memory complexity is only $\mathcal{O}\left(s^2\right)$. Thus, in such cases, subset methods can be applied, whereas lowrank methods cannot. The order of memory complexity $s(p+s)$ for all datasets used in this work together with typical order of s used here is listed in Table 8.

Table 8: Order of the memory load $s(p+s)$ required for the application of lowrank based subspace methods for all datasets and typical orders of s used in this article.

dataset	p	$s(s+p)$					
		$s=100$	$s=200$	$s=300$	$s=400$	$s=500$	$s=1000$
ENB	17922	$1.802 \cdot 10^6$	$3.624 \cdot 10^6$	$5.467 \cdot 10^6$	$7.329 \cdot 10^6$	$9.211 \cdot 10^6$	$1.892 \cdot 10^7$
Red Wine	18,177	$1.828 \cdot 10^6$	$3.675 \cdot 10^6$	$5.543 \cdot 10^6$	$7.431 \cdot 10^6$	$9.338 \cdot 10^6$	$1.918 \cdot 10^7$
California	17,793	$1.789 \cdot 10^6$	$3.599 \cdot 10^6$	$5.428 \cdot 10^6$	$7.277 \cdot 10^6$	$9.146 \cdot 10^6$	$1.879 \cdot 10^7$
Naval	18,690	$1.879 \cdot 10^6$	$3.778 \cdot 10^6$	$5.697 \cdot 10^6$	$7.636 \cdot 10^6$	$9.595 \cdot 10^6$	$1.969 \cdot 10^7$
MNIST	12,458	$1.256 \cdot 10^6$	$2.532 \cdot 10^6$	$3.827 \cdot 10^6$	$5.143 \cdot 10^6$	$6.479 \cdot 10^6$	$1.346 \cdot 10^7$
FashionMNIST	12,458	$1.256 \cdot 10^6$	$2.532 \cdot 10^6$	$3.827 \cdot 10^6$	$5.143 \cdot 10^6$	$6.479 \cdot 10^6$	$1.346 \cdot 10^7$
CIFAR10	668,234	$6.683 \cdot 10^7$	$1.337 \cdot 10^8$	$2.006 \cdot 10^8$	$2.675 \cdot 10^8$	$3.344 \cdot 10^8$	$6.692 \cdot 10^8$
ImageNet10	11,181,642	$1.118 \cdot 10^9$	$2.236 \cdot 10^9$	$3.355 \cdot 10^9$	$4.473 \cdot 10^9$	$5.591 \cdot 10^9$	$1.118 \cdot 10^{10}$

A.4 Prior Distribution

For all problems the prior distribution of the full parameter $\theta \in \mathbb{R}^p$ is chosen to be a centred Gaussian prior $p(\theta) = \mathcal{N}(\theta|0, \lambda^{-1})$. For each dataset one fixed prior precision is used. The precisions were obtained by applying the marginal likelihood optimization for the KFAC LA from the `laplace` software library Daxberger et al. (2021a). For all the datasets but ImageNet10 we averaged the prior precision over five seeds. Due to computational resources and the fact that the prior precision does not vary much across all sets, we estimated the prior precision for ImageNet10 for seed 1 only. The corresponding values are listed in Table 9.

Table 9: Used prior precision for the experiments in this article.

dataset	prior precision λ
ENB	2.0
Red Wine	6.5
California	8.8
Naval Propulsion	4.3
MNIST	18.9
FashionMNIST	32.7
CIFAR10	68.1
ImageNet10	13.0

A.5 Computation Costs

The main factor for the computational costs of all subspace methods depends on the approximation scheme of the LA. In general, to compute P for the subset or low rank methods needs similar time. The only exception is $P_{\text{subset}-\text{magnitude}}$ because it selects the weights depending on their magnitude which is easy to extract for trained NNs. The time to compute the projector using an Nvidia A100 graphics card is given in Table 10 for ImageNet. For the other experiments the computational time is much less, e.g. FashionMNIST needs only seconds for any projector.

Table 10: Computation costs of the projector for ImageNet10. The time to compute P for the lowrank methods (lr) or the subset (sb) methods depends on the low rank approximation or selection methods.

lowrank-KFAC	lowrank-Diagonal	subset-Diagonal	subset-Magnitude	subset-SWAG
17min	28min	28min	$<$ 1min	20min

B EXISTENCE OF AN OPTIMAL SUBSPACE MODEL FOR THE LAPLACE APPROXIMATION

In the following we prove Theorem 1 which we restate here.

Theorem. *Consider the problem* (16) *with* $s \leq s_{\max} = \min(nC, p)$. *Suppose that* $J_X \in \mathbb{R}^{nC\times p}$ *has full rank. For any invertible* $Q \in \mathbb{R}^{s\times s}$ *the matrix*

$$P^* = \Psi J_X^\intercal U_s Q \tag{22}$$

solves (16). *For any such* P^* *we have*

$$\Sigma_{P^*,X} = U_s \Lambda_s U_s^\intercal. \tag{23}$$

If the diagonal elements of Λ *satisfy* $\sigma_s > \sigma_{s+1}$ *then the solution of the minimization problem* (16) *is unique up to reparametrization: any minimizer of* (16) *is of the form* (22).

Proof. For the sake of this proof it will be convenient to work with a more general family of $P \in \mathbb{R}^{p\times s}$ than (22):

$$P = (\Psi J_X^\intercal U_s + K)\, Q\,. \tag{24}$$

Here $Q \in \mathbb{R}^{s\times s}$ is an arbitrary non-singular matrix as in the statement of the theorem and $K \in \mathbb{R}^{p\times s}$ is an arbitrary matrix that satisfies $J_X K = 0$. The proof splits in three parts:

1. We show that for any P of the form (24) we have
$$\Sigma_{P,X} = U_s \Lambda_s U_s^\intercal - U_s K^\intercal A K U_s^\intercal \tag{25}$$
where $A = (\Psi + K\Lambda_s^{-1}K^\intercal)^{-1} \in \mathbb{R}^{s\times s}$ is a positive definite matrix. For the special case $K = 0$, which corresponds to (22), we obtain in particular the identity (23), which minimizes indeed (16) due to the Eckart-Young-Mirsky theorem.
2. Assuming $\sigma_s > \sigma_{s+1}$ we show that any $P \in \mathbb{R}^{p\times s}$ that solves (16) must be of the form (24).
3. Still assuming $\sigma_s > \sigma_{s+1}$ we show that for P of form (24) $\Sigma_{P,X}$ can only minimize (16) if $K = 0$. Together with part 1-2 the claim follows.

For the first part of the proof let us now assume that P is as in (24). We then have

$$J_X P = \overbrace{J_X \Psi J_X^\intercal}^{=\Sigma_X} U_s Q + \overbrace{J_X K}^{=0} Q = \Sigma_X U_s Q = U_s \Lambda_s Q \tag{26}$$

and further

$$\begin{aligned} P^\intercal \Psi^{-1} P &= Q^\intercal (U_s^\intercal J_X \Psi + K^\intercal)\Psi^{-1}(\Psi J_X^\intercal U_s + K)Q \\ &= Q^\intercal (U_s^\intercal J_X \Psi J_X^\intercal U_s + K^\intercal \Psi^{-1} K)Q \end{aligned}$$

where we used in the last equation that $J_X K = 0$ and $K^\intercal J_X^\intercal = 0$. Using $J_X \Psi J_X^\intercal = \Sigma_X$ and $U_s^\intercal \Sigma_X U_s = \Lambda_s$ this can be further simplified to

$$P^\intercal \Psi^{-1} P = Q^\intercal (\Lambda_s + K^\intercal \Psi^{-1} K) Q\,. \tag{27}$$

Applying the Woodbury matrix identity (Wb) we obtain with (26) and (27)

$$\begin{aligned} \Sigma_{P,X} &= J_X P (P^\intercal \Psi^{-1} P)^{-1} P^\intercal J_X^\intercal \\ &= U_s \Lambda_s Q \left(Q^\intercal (\Lambda_s + K^\intercal \Psi^{-1} K) Q\right)^{-1} Q^\intercal \Lambda_s U_s^\intercal \\ &= U_s \Lambda_s (\Lambda_s + K^\intercal \Psi^{-1} K)^{-1} \Lambda_s U_s^\intercal \\ &\overset{\text{Wb}}{=} U_s \Lambda_s (\Lambda_s^{-1} - \Lambda_s^{-1} K^\intercal (\Psi + K\Lambda_s^{-1} K^\intercal)^{-1} K \Lambda_s^{-1}) \Lambda_s U_s^\intercal \\ &= U_s \Lambda_s U_s - U_s K^\intercal (\Psi + K\Lambda_s^{-1} K^\intercal)^{-1} K U_s^\intercal \\ &= U_s \Lambda_s U_s^\intercal - U_s K^\intercal A K U_s^\intercal\,, \end{aligned}$$

which concludes part 1 of the proof.

For part 2-3 of the proof we use that, according to the Eckart-Young-Mirsky theorem, the minimizer $U_s\Lambda_s U_s^\intercal$ of (16) is *unique* if $\sigma_s > \sigma_{s+1}$.

Considering part 2 of the proof, this uniqueness implies that for any P that solves (16) the projected covariance matrix has to be of the form $\Sigma_{P,X} = U_s\Lambda_s U_s^\intercal$ and in particular

$$(\Sigma_{P,X} - \Sigma_X)U_s = U_s\Lambda_s - U_s\Lambda_s = 0\,.$$

Plugging the definition of $\Sigma_{P,X}$ and Σ_X in this identity we obtain

$$J_X\left(P(P^\intercal\Psi^{-1}P)^{-1}P^\intercal J_X^\intercal U_s - \Psi J_X^\intercal U_s\right) = 0\,,$$

so that we know that the matrix

$$K = P(P^\intercal\Psi^{-1}P)^{-1}P^\intercal J_X^\intercal U_s - \Psi J_X^\intercal U_s =: PB - \Psi J_X^\intercal U_s$$

satisfies $J_X K = 0$. Note that B is non-singular because $\det(U_s^\intercal J_X P)\det(B) = \det(U_s^\intercal J_X PB) = \det(U_s^\intercal\Sigma_{P,X}U_s) = \det(\Lambda_s) \neq 0$. The choice $Q := B^{-1}$ therefore concludes part 2 of the proof.

We are left with part 3 of the proof. Assume therefore that P is of the form (24) *and* that $\Sigma_{P,X}$ minimizes (16). We have by part 1 of the proof

$$\Sigma_{P,X} = U_s\Lambda_s U_s^\intercal - U_s K^\intercal A K U_s^\intercal\,,$$

where A is positive definite. Now, under the assumption that $\sigma_s > \sigma_{s+1}$ we can, once more, use the uniqueness part of the Eckart-Young-Mirsky theorem to conclude that $\Sigma_{P,X} = U_s\Lambda_s U_s^\intercal$ which implies that

$$U_s K^\intercal A K U_s^\intercal = 0 \tag{28}$$

Multiplying both sides with U_s and using that $U_s^\intercal U_s = \mathbb{1}_s$ we obtain

$$U_s^\intercal U_s K^\intercal A K U_s^\intercal U_s = K^\intercal A K = 0$$

which implies that $K = 0$ due to the positive definiteness of A. □

Note that all we used in the proof of this Theorem were the identities (7) and (15) so that the statement of the theorem does not really require Ψ to be derived via a Laplace approximation.

B.1 Additional Remarks to Theorem 1

Optimality of P^* According to the Trace Ordering. Note, that Lemma 2 below implies that the ordered eigenvalues (in decreasing order) of any $\Sigma_{P,X}$ satisfy $\lambda_i(\Sigma_{P,X}) \leq \lambda_i(\Sigma_X)$ as follows from the min-max theorem (Horn and Johnson, 2012, Theorem 4.2.6). Since any $\Sigma_{P,X}$ has rank s and as $\Sigma_{P^*,X}$ shares the s largest eigenvalues with Σ_X we obtain

$$\lambda_i(\Sigma_{P,X}) \leq \lambda_i(\Sigma_{P^*,X}) = \lambda_i(\Sigma_X)1_{i\leq s} \tag{29}$$

so that $\Sigma_{P^*,X}$ is actually optimal w.r.t. the trace criterion (20):

$$\operatorname{Tr}\Sigma_{P,X} \leq \operatorname{Tr}\Sigma_{P^*,X}\,.$$

Implications on Coverage (for Regression Problems). Theorem 1 also implies, in a regression setting, that among any ellipsoidal region of the form $E_{P^*}(c) = \{Y : (Y - f_{\hat\theta}(X))^\intercal(\Sigma_{P,X} + \sigma\mathbb{1}_{nC})^{-1}(Y - f_{\hat\theta}(X)) \leq c\}$ the coverage of samples of the full LA is maximal for the choice $P = P^*$ for $c > 0$ large enough.

In the following, we show that this is true. Using $Z \sim \mathcal{N}(0, \mathbb{1}_{nC})$ and the invariance of the standard multivariate normal with respect to orthogonal transformations, we obtain via diagonalization of Σ_P and $\Sigma_{P,X}$

$$\begin{aligned}
&P_{Y\sim\mathcal{N}(f_{\hat\theta}(X),\Sigma_X+\sigma^2\mathbb{1}_{nC})}(Y \in E_P(c))\\
&= P(Z^\intercal(\Sigma_X + \sigma^2\mathbb{1}_{nC})^{1/2}(\Sigma_{P,X} + \sigma^2\mathbb{1}_{nC})^{-1}(\Sigma_X + \sigma^2\mathbb{1}_{nC})^{1/2}Z \leq c)\\
&= P(Z^\intercal(\Lambda + \sigma^2\mathbb{1}_{nC})^{1/2}U(\Lambda(\Sigma_{P,X}) + \sigma^2\mathbb{1}_{nC})^{-1}U^\intercal(\Lambda + \sigma^2\mathbb{1}_{nC})^{1/2}Z \leq c)\\
&\leq P(Z^\intercal A_U Z \leq c)\,,
\end{aligned}$$

where U is an orthogonal matrix (depending on the choice of P), $\Lambda(\Sigma_{P,X})$ and $\Lambda = \Lambda(\Sigma_X)$ denote the corresponding diagonal matrices with decreasing eigenvalues on the diagonal and where we wrote $A_U = (\Lambda + \sigma^2 \mathbb{1}_{nC})^{1/2} U (\Lambda(\Sigma_{P^*,X}) + \sigma^2 \mathbb{1}_{nC})^{-1} U^\intercal (\Lambda + \sigma^2 \mathbb{1}_{nC})^{1/2}$. The inequality above follows from (29). For $P = P^*$ one easily gets using $\Sigma_{P^*,X} = U_s \Lambda_s U_s^\intercal$ (Theorem 1)

$$P_{Y \sim \mathcal{N}(f_{\hat{\theta}}(X), \sigma^2 \Sigma_X)}(Y \in E_{P^*}(c)) = P(Z^\intercal A_{\mathbb{1}_{nC}} Z \leq c)\,.$$

Once we can show

$$P(Z^\intercal A_U Z \leq c) \leq P(Z^\intercal A_{\mathbb{1}_{nC}} Z \leq c) \tag{30}$$

the proof of the claim above is therefore finished. The ordering (30) follows from (Székely and Bakirov, 2003) once we know that for any $1 \leq k \leq n$

$$\sum_{i=1}^{k} \lambda_i(A_{\mathbb{1}_{nC}}) \leq \sum_{i=1}^{k} \lambda_i(A_U)\,.$$

Due to (Marshall et al., 1979, Theorem 20.A.2) we have, writing λ_i for the diagonal elements of $\Lambda = \Lambda(\Sigma_X)$,

$$\begin{aligned}\sum_{i=1}^{k} \lambda_i(A_U) &= \max_{V \in \mathbb{R}^{k \times nC} : VV^\intercal = \mathbb{1}_k} \operatorname{Tr}(V A_U V^\intercal) \\ &\geq \operatorname{Tr} \mathbb{1}_{k \times nC} A(U) \mathbb{1}_{k \times nC}^\intercal = \sum_{i,j=1}^{k} (\lambda_i + \sigma^2) |U_{ij}|^2 (\lambda_j 1_{j \leq s} + \sigma^2) = \sum_m \theta_m \sum_{i,j=1}^{k} (\lambda_i + \sigma^2) P_m (\lambda_j 1_{j \leq s} + \sigma^2)^{-1}\end{aligned} \tag{31}$$

where $\mathbb{1}_{k \times nC}$ denotes the projection on the first k components and where we used in the second line Birkhoff's theorem (Horn and Johnson, 2012, Theorem 8.7.2) to replace the doubly stochastic matrix $(|U_{ij}|^2)_{ij}$ by $\sum_m \theta_m P_m$ with $\sum_m \theta_m = 1, \theta_m \geq 0$ and permutation matrices P_m. By the rearrangement inequality (Hardy et al., 1988, Theorem 368) the object (31) becomes minimal when all P_m equal $\mathbb{1}_{nC}$ and in this case we just obtain $\sum_{i=1}^{k} \lambda_i(A_{\mathbb{1}_{nC}})$.

C TRACE CRITERION

Ideally we would like to choose a P such that the predictive distribution of the full (6), (8) and subspace model (14) are as close as possible. In Section 4 we introduced the KL divergence between these distributions. However, in practice the full LA is unknown, which makes this metric infeasible.

As an alternative criterion we propose for our purposes to use the trace $\operatorname{Tr} \Sigma_{P,X}$ instead. This criterion is feasible to compute, aligns well with the KL divergence as we show empirically in Section 5 and can be motivated by the following lemma:

Lemma 2. *For any $P \in \mathbb{R}^{p \times s}$ we have*

$$\Sigma_{P,X} \preccurlyeq \Sigma_X \tag{32}$$

in the Loewner ordering, i.e. $\Sigma_X - \Sigma_{P,X}$ is positive semi-definite. In particular we have

$$\operatorname{Tr} \Sigma_{P,X} \leq \operatorname{Tr} \Sigma_X\,. \tag{33}$$

Proof. Due to the identities (7) and (15) it suffices to show that $P \left(P^\intercal \Psi^{-1} P\right)^{-1} P^\intercal \preccurlyeq \Psi$, i.e., that the matrix

$$\Psi - P(P^\intercal \Psi^{-1} P)^{-1} P^\intercal = \Psi^{1/2} (\mathbb{1} - B) \Psi^{1/2}$$

is positive semi-definite, where we introduced $B = W \left(W^\intercal W\right)^{-1} W^\intercal$ with $W = \Psi^{-1/2} P$. It's easy to check that B is a projection ($B^2 = B$ and $B^\intercal = B$) which thus has only eigenvalues contained in $\{0, 1\}$. From this it follows that $\mathbb{1} - B$ and $\Psi^{1/2}(\mathbb{1} - B)\Psi^{1/2}$ are positive semi-definite and thus (32).

From (32) we obtain $\operatorname{Tr}(\Sigma_X - \Sigma_{P,X}) = \operatorname{Tr} \Sigma_X - \operatorname{Tr} \Sigma_{P,X} \geq 0$ from which (33) follows. □

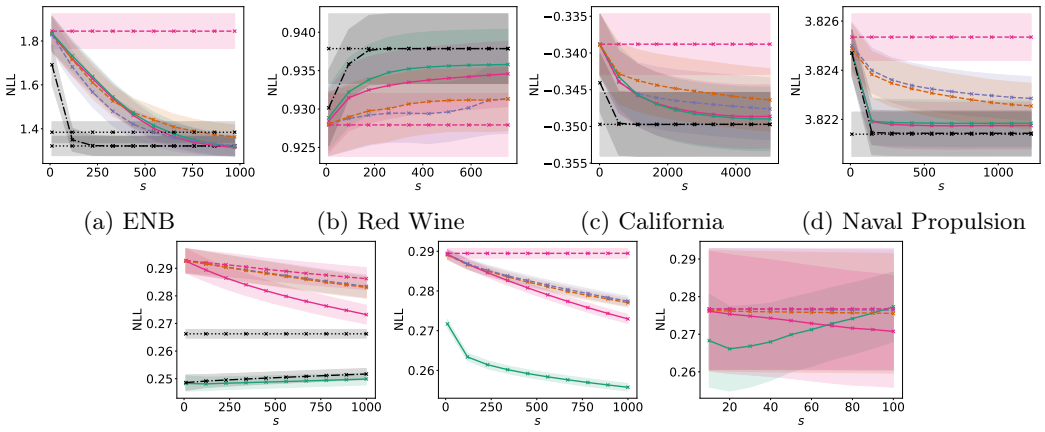
The relation (33) shows that $\mathrm{Tr}(\Sigma_X - \Sigma_{P,X}) \geq 0$ is a non-negative quantity that quantifies the closeness between Σ_X and $\Sigma_{P,X}$. Since Σ_X does not depend on P we can judge whether for two P_1, P_2 we have $\mathrm{Tr}(\Sigma_X - \Sigma_{P_1,X}) \geq \mathrm{Tr}(\Sigma_X - \Sigma_{P_2,X})$ by simply comparing whether $\mathrm{Tr}\,\Sigma_{P_1,X} \geq \mathrm{Tr}\,\Sigma_{P_2,X}$. In other words, we can take $\mathrm{Tr}\,\Sigma_{P,X}$ to rank the quality of different P. Relation (33) ensures that there is an upper bound for this quantity. Recall that the trace is the sum of all eigenvalues of a matrix. If the trace of one approximation is greater than another one, it means that this affine subspace covers an eigenspace of greater eigenvalues.

D ADDITIONAL RESULTS ON UNCERTAINTY QUANTIFICATION

Table 11 extends Table 1 by including a broader range of subspace dimensions s and reporting the standard error. For the Brier score, we use in this work the convention $\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}(1_{y_i=c} - \hat{p}_{ic})^2$ that averages over datapoints but sums over classes. Figure 3 summarizes all results of the NLL metric for all methods and datasets studied in this article with the same colour coding as in Figure 1.

metric	dataset	s	lowrank-D.	lowrank-KFAC	subset-D.	subset-M.	subset-SWAG	$\hat{\theta}$
NLL	FashionMNIST	120	0.289 (0.004)	**0.248** (0.002)	0.292 (0.004)	0.292 (0.004)	0.292 (0.004)	0.293 (0.004)
		230	0.286 (0.004)	**0.248** (0.002)	0.291 (0.004)	0.290 (0.004)	0.290 (0.004)	0.293 (0.004)
		450	0.282 (0.004)	**0.249** (0.002)	0.290 (0.004)	0.288 (0.004)	0.288 (0.004)	0.293 (0.004)
		670	0.278 (0.004)	**0.249** (0.002)	0.288 (0.004)	0.286 (0.004)	0.286 (0.004)	0.293 (0.004)
		1000	0.273 (0.003)	**0.250** (0.002)	0.286 (0.004)	0.283 (0.004)	0.283 (0.004)	0.293 (0.004)
	CIFAR10	120	0.287 (0.001)	**0.263** (0.001)	0.289 (0.001)	0.287 (0.001)	0.287 (0.001)	0.289 (0.001)
		230	0.285 (0.001)	**0.261** (0.001)	0.289 (0.001)	0.285 (0.001)	0.285 (0.001)	0.289 (0.001)
		450	0.281 (0.001)	**0.259** (0.001)	0.289 (0.001)	0.282 (0.001)	0.283 (0.001)	0.289 (0.001)
		670	0.277 (0.001)	**0.258** (0.001)	0.289 (0.001)	0.280 (0.001)	0.280 (0.001)	0.289 (0.001)
		1000	0.273 (0.001)	**0.256** (0.001)	0.289 (0.001)	0.277 (0.001)	0.277 (0.001)	0.289 (0.001)
	ImageNet10	10	0.276 (0.016)	**0.268** (0.012)	0.277 (0.016)	0.276 (0.016)	0.277 (0.016)	0.277 (0.016)
		20	0.275 (0.016)	**0.266** (0.011)	0.277 (0.016)	0.276 (0.016)	0.277 (0.016)	0.277 (0.016)
		50	0.274 (0.015)	**0.270** (0.010)	0.277 (0.016)	0.276 (0.016)	0.277 (0.016)	0.277 (0.016)
		70	**0.272** (0.015)	0.273 (0.010)	0.277 (0.016)	0.276 (0.016)	0.276 (0.016)	0.277 (0.016)
		100	**0.271** (0.015)	0.277 (0.009)	0.277 (0.016)	0.276 (0.016)	0.276 (0.016)	0.277 (0.016)
ECE	FashionMNIST	120	0.039 (0.001)	**0.010** (0.001)	0.040 (0.001)	0.040 (0.001)	0.040 (0.001)	0.041 (0.001)
		230	0.039 (0.001)	**0.012** (0.001)	0.040 (0.001)	0.040 (0.001)	0.040 (0.001)	0.041 (0.001)
		450	0.037 (0.001)	**0.013** (0.001)	0.040 (0.001)	0.039 (0.001)	0.039 (0.001)	0.041 (0.001)
		670	0.035 (0.001)	**0.015** (0.001)	0.039 (0.001)	0.039 (0.001)	0.039 (0.001)	0.041 (0.001)
		1000	0.033 (0.001)	**0.016** (0.001)	0.038 (0.001)	0.038 (0.001)	0.038 (0.001)	0.041 (0.001)
	CIFAR10	120	0.037 (0.001)	**0.029** (0.000)	0.038 (0.001)	0.037 (0.001)	0.037 (0.001)	0.038 (0.001)
		230	0.037 (0.001)	**0.029** (0.000)	0.038 (0.001)	0.037 (0.001)	0.037 (0.001)	0.038 (0.001)
		450	0.035 (0.001)	**0.027** (0.000)	0.038 (0.001)	0.036 (0.001)	0.036 (0.001)	0.038 (0.001)
		670	0.034 (0.001)	**0.026** (0.000)	0.038 (0.001)	0.035 (0.001)	0.035 (0.001)	0.038 (0.001)
		1000	0.033 (0.001)	**0.025** (0.001)	0.038 (0.001)	0.035 (0.001)	0.035 (0.001)	0.038 (0.001)
	ImageNet10	10	0.039 (0.002)	**0.036** (0.003)	0.040 (0.003)	0.040 (0.003)	0.040 (0.003)	0.040 (0.003)
		20	0.040 (0.003)	**0.036** (0.002)	0.040 (0.003)	0.041 (0.002)	0.039 (0.003)	0.040 (0.003)
		50	**0.039** (0.003)	0.041 (0.004)	0.040 (0.003)	0.040 (0.002)	0.040 (0.003)	0.040 (0.003)
		70	**0.038** (0.002)	0.044 (0.004)	0.040 (0.003)	0.039 (0.002)	0.040 (0.003)	0.040 (0.003)
		100	**0.038** (0.001)	0.048 (0.006)	0.040 (0.003)	0.039 (0.002)	0.040 (0.003)	0.040 (0.003)
Brier	FashionMNIST	120	0.135 (0.001)	**0.128** (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)
		230	0.134 (0.001)	**0.129** (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)
		450	0.134 (0.001)	**0.129** (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)
		670	0.134 (0.001)	**0.129** (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)	0.135 (0.001)
		1000	0.133 (0.001)	**0.129** (0.001)	0.135 (0.001)	0.134 (0.001)	0.134 (0.001)	0.135 (0.001)
	CIFAR10	120	0.125 (0.001)	**0.123** (0.001)	0.126 (0.001)	0.125 (0.001)	0.125 (0.001)	0.126 (0.001)
		230	0.125 (0.001)	**0.122** (0.001)	0.126 (0.001)	0.125 (0.001)	0.125 (0.001)	0.126 (0.001)
		450	0.125 (0.001)	**0.122** (0.001)	0.126 (0.001)	0.125 (0.001)	0.125 (0.001)	0.126 (0.001)
		670	0.124 (0.001)	**0.122** (0.001)	0.126 (0.001)	0.125 (0.001)	0.125 (0.001)	0.126 (0.001)
		1000	0.124 (0.001)	**0.122** (0.001)	0.126 (0.001)	0.124 (0.001)	0.124 (0.001)	0.126 (0.001)
	ImageNet10	10	0.129 (0.008)	**0.127** (0.007)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)
		20	0.129 (0.008)	**0.127** (0.006)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)
		50	0.129 (0.008)	**0.127** (0.006)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)
		70	0.128 (0.007)	**0.127** (0.006)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)
		100	0.128 (0.007)	**0.127** (0.006)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)	0.129 (0.008)

Table 11: Extension of Table 1 with additional values of s and standard errors over 5 seeds (in parentheses).



(a) ENB (b) Red Wine (c) California (d) Naval Propulsion

(e) Fashion-MNIST (f) CIFAR10 (g) ImageNet10

Figure 3: Comparison of the NLL for all subspace models and datasets studied in this article and various subspace dimension s. Different choices of P are marked by different colours and line types with the same colour coding as in Figure 1.

E ADDITIONAL MATERIAL ON CORRUPTED DATA

E.1 Corrupted CIFAR10

We study 7 corruptions from Hendrycks and Dietterich (2019), namely

- 3 types of noise: impulse, shot and speckle noise,
- 2 types of blurring: Gaussian and motion blur,
- a distortion via an elastic transform,
- an increased luminance (brightness).

We use the data available at https://doi.org/10.5281/zenodo.2535967 that contains the corruptions with different levels of severity. We do not distinguish between different levels of severity in our analysis. The entire set of results for all studied corruptions and subspace dimensions s can be found in Table 12 including standard errors obtained via 5 different seeds.

metric	corruption	s	lowrank-D.	lowrank-KFAC	subset-D.	subset-M.	subset-SWAG	$\hat{\theta}$
NLL	brightness	100	0.342 (0.002)	**0.316** (0.002)	0.344 (0.002)	0.341 (0.002)	0.342 (0.002)	0.344 (0.002)
	brightness	200	0.340 (0.002)	**0.313** (0.002)	0.344 (0.002)	0.340 (0.002)	0.340 (0.002)	0.344 (0.002)
	brightness	300	0.338 (0.002)	**0.312** (0.002)	0.344 (0.002)	0.338 (0.002)	0.338 (0.002)	0.344 (0.002)
	elastic transform	100	0.524 (0.004)	**0.492** (0.004)	0.527 (0.004)	0.524 (0.004)	0.524 (0.004)	0.527 (0.004)
	elastic transform	200	0.522 (0.004)	**0.490** (0.004)	0.527 (0.004)	0.522 (0.004)	0.522 (0.004)	0.527 (0.004)
	elastic transform	300	0.520 (0.004)	**0.488** (0.004)	0.527 (0.004)	0.520 (0.004)	0.521 (0.004)	0.527 (0.004)
	gaussian blur	100	0.353 (0.002)	**0.342** (0.002)	0.354 (0.002)	0.353 (0.002)	0.353 (0.002)	0.354 (0.002)
	gaussian blur	200	0.353 (0.002)	**0.341** (0.002)	0.354 (0.002)	0.352 (0.002)	0.353 (0.002)	0.354 (0.002)
	gaussian blur	300	0.352 (0.002)	**0.341** (0.002)	0.354 (0.002)	0.352 (0.002)	0.352 (0.002)	0.354 (0.002)
	impulse noise	100	1.841 (0.057)	**1.679** (0.048)	1.861 (0.059)	1.848 (0.057)	1.846 (0.056)	1.861 (0.059)
	impulse noise	200	1.822 (0.055)	**1.660** (0.047)	1.861 (0.059)	1.836 (0.056)	1.831 (0.055)	1.861 (0.059)
	impulse noise	300	1.804 (0.054)	**1.647** (0.045)	1.861 (0.059)	1.823 (0.056)	1.820 (0.055)	1.861 (0.059)
	motion blur	100	0.589 (0.003)	**0.560** (0.003)	0.591 (0.003)	0.589 (0.003)	0.589 (0.003)	0.591 (0.003)
	motion blur	200	0.587 (0.003)	**0.558** (0.002)	0.591 (0.003)	0.587 (0.003)	0.587 (0.003)	0.591 (0.003)
	motion blur	300	0.585 (0.003)	**0.556** (0.002)	0.591 (0.003)	0.585 (0.003)	0.586 (0.003)	0.591 (0.003)
	shot noise	100	1.531 (0.037)	**1.400** (0.033)	1.548 (0.038)	1.536 (0.037)	1.535 (0.037)	1.548 (0.038)
	shot noise	200	1.515 (0.036)	**1.383** (0.033)	1.548 (0.038)	1.526 (0.036)	1.524 (0.037)	1.548 (0.038)
	shot noise	300	1.501 (0.036)	**1.371** (0.032)	1.548 (0.038)	1.516 (0.036)	1.516 (0.036)	1.548 (0.038)
	speckle noise	100	1.481 (0.035)	**1.352** (0.032)	1.497 (0.036)	1.485 (0.035)	1.484 (0.035)	1.497 (0.036)
	speckle noise	200	1.466 (0.034)	**1.335** (0.031)	1.497 (0.036)	1.476 (0.034)	1.474 (0.034)	1.497 (0.036)
	speckle noise	300	1.452 (0.033)	**1.324** (0.030)	1.497 (0.036)	1.466 (0.034)	1.465 (0.034)	1.497 (0.036)
ECE	brightness	100	0.045 (0.000)	**0.036** (0.000)	0.046 (0.000)	0.045 (0.000)	0.045 (0.000)	0.046 (0.000)
	brightness	200	0.044 (0.000)	**0.035** (0.000)	0.046 (0.000)	0.044 (0.000)	0.045 (0.000)	0.046 (0.000)
	brightness	300	0.044 (0.000)	**0.034** (0.000)	0.046 (0.000)	0.044 (0.000)	0.044 (0.000)	0.046 (0.000)
	elastic transform	100	0.064 (0.001)	**0.051** (0.001)	0.065 (0.001)	0.064 (0.001)	0.064 (0.001)	0.065 (0.001)
	elastic transform	200	0.063 (0.001)	**0.050** (0.001)	0.065 (0.001)	0.064 (0.001)	0.064 (0.001)	0.065 (0.001)
	elastic transform	300	0.063 (0.001)	**0.049** (0.001)	0.065 (0.001)	0.063 (0.001)	0.063 (0.001)	0.065 (0.001)
	gaussian blur	100	0.032 (0.000)	**0.023** (0.000)	0.032 (0.000)	0.031 (0.000)	0.032 (0.000)	0.032 (0.000)
	gaussian blur	200	0.031 (0.000)	**0.022** (0.000)	0.032 (0.000)	0.031 (0.000)	0.031 (0.000)	0.032 (0.000)
	gaussian blur	300	0.030 (0.000)	**0.021** (0.000)	0.032 (0.000)	0.030 (0.000)	0.031 (0.000)	0.032 (0.000)
	impulse noise	100	0.230 (0.004)	**0.206** (0.004)	0.233 (0.004)	0.231 (0.004)	0.231 (0.004)	0.233 (0.004)
	impulse noise	200	0.227 (0.004)	**0.203** (0.004)	0.233 (0.004)	0.229 (0.004)	0.229 (0.004)	0.233 (0.004)
	impulse noise	300	0.225 (0.004)	**0.200** (0.004)	0.233 (0.004)	0.228 (0.004)	0.227 (0.004)	0.233 (0.004)
	motion blur	100	0.071 (0.001)	**0.060** (0.001)	0.072 (0.001)	0.071 (0.001)	0.071 (0.001)	0.072 (0.001)
	motion blur	200	0.071 (0.001)	**0.059** (0.001)	0.072 (0.001)	0.071 (0.001)	0.071 (0.001)	0.072 (0.001)
	motion blur	300	0.070 (0.001)	**0.057** (0.001)	0.072 (0.001)	0.070 (0.001)	0.070 (0.001)	0.072 (0.001)
	shot noise	100	0.198 (0.006)	**0.181** (0.006)	0.200 (0.006)	0.198 (0.005)	0.198 (0.006)	0.200 (0.006)
	shot noise	200	0.196 (0.006)	**0.178** (0.006)	0.200 (0.006)	0.197 (0.005)	0.197 (0.006)	0.200 (0.006)
	shot noise	300	0.194 (0.006)	**0.176** (0.006)	0.200 (0.006)	0.196 (0.005)	0.196 (0.006)	0.200 (0.006)
	speckle noise	100	0.189 (0.004)	**0.172** (0.005)	0.191 (0.004)	0.190 (0.004)	0.190 (0.004)	0.191 (0.004)
	speckle noise	200	0.188 (0.004)	**0.170** (0.005)	0.191 (0.004)	0.189 (0.004)	0.188 (0.004)	0.191 (0.004)
	speckle noise	300	0.186 (0.004)	**0.168** (0.005)	0.191 (0.004)	0.188 (0.004)	0.187 (0.004)	0.191 (0.004)
Brier	brightness	100	0.149 (0.001)	**0.145** (0.001)	0.149 (0.001)	0.149 (0.001)	0.149 (0.001)	0.149 (0.001)
	brightness	200	0.149 (0.001)	**0.145** (0.001)	0.149 (0.001)	0.149 (0.001)	0.149 (0.001)	0.149 (0.001)
	brightness	300	0.148 (0.001)	**0.145** (0.001)	0.149 (0.001)	0.148 (0.001)	0.148 (0.001)	0.149 (0.001)
	elastic transform	100	0.230 (0.001)	**0.226** (0.001)	0.231 (0.001)	0.230 (0.001)	0.230 (0.001)	0.231 (0.001)
	elastic transform	200	0.230 (0.001)	**0.225** (0.001)	0.231 (0.001)	0.230 (0.001)	0.230 (0.001)	0.231 (0.001)
	elastic transform	300	0.230 (0.001)	**0.225** (0.001)	0.231 (0.001)	0.230 (0.001)	0.230 (0.001)	0.231 (0.001)
	gaussian blur	100	0.166 (0.001)	**0.164** (0.001)	0.166 (0.001)	0.165 (0.001)	0.166 (0.001)	0.166 (0.001)
	gaussian blur	200	0.165 (0.001)	**0.164** (0.001)	0.166 (0.001)	0.165 (0.001)	0.165 (0.001)	0.166 (0.001)
	gaussian blur	300	0.165 (0.001)	**0.164** (0.001)	0.166 (0.001)	0.165 (0.001)	0.165 (0.001)	0.166 (0.001)
	impulse noise	100	0.614 (0.008)	**0.597** (0.008)	0.616 (0.008)	0.615 (0.008)	0.615 (0.008)	0.616 (0.008)
	impulse noise	200	0.612 (0.008)	**0.595** (0.008)	0.616 (0.008)	0.614 (0.008)	0.613 (0.008)	0.616 (0.008)
	impulse noise	300	0.611 (0.008)	**0.593** (0.008)	0.616 (0.008)	0.612 (0.008)	0.612 (0.008)	0.616 (0.008)
	motion blur	100	0.259 (0.001)	**0.255** (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)
	motion blur	200	0.259 (0.001)	**0.254** (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)
	motion blur	300	0.258 (0.001)	**0.254** (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)	0.259 (0.001)
	shot noise	100	0.509 (0.009)	**0.496** (0.009)	0.510 (0.010)	0.509 (0.009)	0.509 (0.009)	0.510 (0.010)
	shot noise	200	0.507 (0.009)	**0.495** (0.009)	0.510 (0.010)	0.508 (0.009)	0.508 (0.010)	0.510 (0.010)
	shot noise	300	0.506 (0.009)	**0.493** (0.009)	0.510 (0.010)	0.507 (0.009)	0.507 (0.009)	0.510 (0.010)
	speckle noise	100	0.490 (0.008)	**0.478** (0.008)	0.491 (0.008)	0.490 (0.008)	0.490 (0.008)	0.491 (0.008)
	speckle noise	200	0.489 (0.008)	**0.476** (0.008)	0.491 (0.008)	0.489 (0.008)	0.489 (0.008)	0.491 (0.008)
	speckle noise	300	0.488 (0.008)	**0.475** (0.008)	0.491 (0.008)	0.489 (0.008)	0.489 (0.008)	0.491 (0.008)

Table 12: Extension of Table 2 for corrupted CIFAR10 including more corruptions, several values of s and standard errors over 5 seeds (in parentheses). Smallest values in a row are marked in bold.

E.2 Corrupted MNIST

In addition to corrupted CIFAR10 we also study a corrupted version of MNIST. We use the 15 corruption types from Mu and Gilmer (2019). These include:

- Noise: shot noise and impulse noise simulate random corruptions during the imaging process.
- Blur: glass blur (viewing through frosted glass via local pixel shuffling) and motion blur (blurring along a random line).
- Affine transformations: shear, translate, scale, and rotate.
- Occlusions and overlays: stripe (inverted vertical stripe), spatter (random splotches), zigzag and dotted line (superimposed patterns with exponentially controlled brightness).
- Other: brightness (increased luminance) and canny edges (edge detection filter), fog (simulation via diamond square algorithm).

The model was trained on MNIST. We used the same architecture as for FashionMNIST. For details on the training procedure, see Section A.1. For all subspace methods we fixed the subspace dimension to be $s = 100$. Tables 13 and 14 report the result on approximation quality (compared to the full LA) and performance in uncertainty quantification. The approximation quality results show a clear preference for the lowrank methods, especially for $P_{\text{lowrank}-\text{KFAC}}$. For the evaluation of uncertainty quantification the results are more mixed but in the majority of the corruptions we obtain the best scores for lowrank methods. We suspect the mixed behavior for this dataset to be linked to a non-optimal performance of the full LA itself, so that a larger deviation from the latter can lead to a better score.

metric	corruption	lowrank-Diagonal	lowrank-KFAC	subset-Diagonal	subset-Magnitude	subset-SWAG
KL	brightness	0.868 (0.006)	**0.445** (0.020)	1.759 (0.083)	1.500 (0.063)	1.587 (0.069)
	canny edges	1.081 (0.033)	**0.481** (0.006)	3.183 (0.112)	2.338 (0.070)	2.488 (0.072)
	dotted line	0.765 (0.015)	**0.320** (0.008)	2.073 (0.046)	1.645 (0.037)	1.687 (0.037)
	fog	0.305 (0.008)	**0.212** (0.005)	0.414 (0.015)	0.386 (0.012)	0.393 (0.015)
	glass blur	0.673 (0.005)	**0.356** (0.008)	1.219 (0.026)	1.030 (0.019)	1.080 (0.023)
	impulse noise	1.224 (0.029)	**0.647** (0.011)	3.067 (0.124)	2.378 (0.064)	2.572 (0.075)
	motion blur	0.636 (0.011)	**0.335** (0.008)	1.210 (0.030)	1.065 (0.024)	1.069 (0.024)
	rotate	0.693 (0.013)	**0.283** (0.006)	1.752 (0.032)	1.453 (0.025)	1.449 (0.024)
	scale	0.916 (0.011)	**0.373** (0.010)	1.923 (0.056)	1.616 (0.038)	1.627 (0.042)
	shear	0.641 (0.010)	**0.270** (0.007)	1.607 (0.032)	1.342 (0.033)	1.352 (0.031)
	shot noise	0.796 (0.009)	**0.360** (0.008)	1.956 (0.050)	1.544 (0.039)	1.611 (0.039)
	spatter	0.636 (0.008)	**0.264** (0.007)	1.666 (0.048)	1.337 (0.037)	1.358 (0.035)
	stripe	1.405 (0.039)	**0.833** (0.038)	3.254 (0.449)	3.442 (0.169)	3.586 (0.141)
	translate	1.133 (0.029)	**0.536** (0.011)	2.980 (0.025)	2.526 (0.026)	2.482 (0.024)
	zigzag	0.963 (0.022)	**0.444** (0.010)	2.664 (0.051)	2.209 (0.043)	2.237 (0.042)
Log-Trace	brightness	10.681 (0.063)	**11.273** (0.075)	-inf	8.876 (0.043)	8.557 (0.018)
	canny edges	11.602 (0.082)	**12.208** (0.058)	-inf	9.730 (0.049)	9.488 (0.054)
	dotted line	11.572 (0.048)	**12.173** (0.022)	-inf	9.420 (0.048)	9.346 (0.032)
	fog	8.786 (0.033)	**9.371** (0.035)	-inf	7.221 (0.040)	7.015 (0.079)
	glass blur	10.286 (0.054)	**10.913** (0.041)	-inf	8.600 (0.059)	8.299 (0.023)
	impulse noise	11.337 (0.075)	**11.874** (0.059)	-inf	9.607 (0.049)	9.336 (0.033)
	motion blur	10.433 (0.054)	**10.997** (0.032)	-inf	8.344 (0.053)	8.304 (0.047)
	rotate	11.258 (0.045)	**11.909** (0.015)	-inf	9.091 (0.048)	9.075 (0.038)
	scale	10.783 (0.061)	**11.593** (0.028)	-inf	8.833 (0.080)	8.818 (0.033)
	shear	11.319 (0.039)	**11.922** (0.016)	-inf	9.147 (0.041)	9.114 (0.026)
	shot noise	11.247 (0.050)	**11.835** (0.030)	-inf	9.259 (0.049)	9.099 (0.032)
	spatter	11.466 (0.045)	**12.069** (0.020)	-inf	9.277 (0.052)	9.221 (0.033)
	stripe	11.719 (0.059)	**12.154** (0.039)	-inf	9.751 (0.046)	9.599 (0.022)
	translate	11.348 (0.049)	**11.972** (0.018)	-inf	9.288 (0.062)	9.305 (0.041)
	zigzag	11.621 (0.051)	**12.190** (0.025)	-inf	9.486 (0.042)	9.435 (0.029)

Table 13: Quality of approximation of the full Laplace approximation via the subspace models studied in this work on corrupted versions of MNIST for subspace dimension $s = 100$. We use KL-divergence 19 and logarithm of the trace 20 to measure approximation quality. The standard error, across 5 seeds, is reported in parentheses. Smallest (KL) and highest (Log-Trace) values in a row are marked in bold. For $P_{\text{subset}-\text{Diagonal}}$ the uncertainties are identical or almost identitical to zero for the considered s, which yields a Log-Trace of $-\infty$.

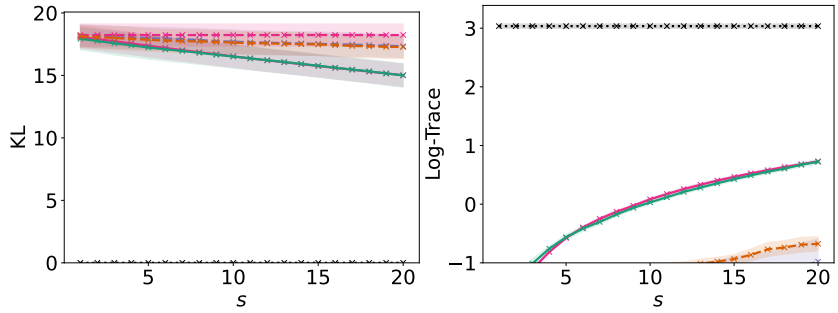
metric	corruption	lowrank-D.	lowrank-KFAC	subset-D.	subset-M.	subset-SWAG	$\hat{\theta}$
NLL	brightness	0.106 (0.018)	0.145 (0.023)	0.100 (0.019)	0.102 (0.019)	**0.099** (0.019)	0.100 (0.019)
	canny edges	1.744 (0.156)	**1.334** (0.104)	2.767 (0.303)	2.324 (0.242)	2.339 (0.237)	2.784 (0.301)
	dotted line	**0.066** (0.002)	0.086 (0.002)	0.076 (0.002)	0.071 (0.002)	0.069 (0.002)	0.075 (0.002)
	fog	0.315 (0.017)	0.350 (0.017)	0.293 (0.017)	0.300 (0.017)	0.298 (0.017)	**0.291** (0.018)
	glass blur	0.233 (0.003)	0.256 (0.002)	0.235 (0.003)	0.234 (0.003)	**0.232** (0.003)	0.235 (0.003)
	impulse noise	**0.327** (0.038)	0.336 (0.032)	0.407 (0.055)	0.369 (0.047)	0.363 (0.045)	0.407 (0.055)
	motion blur	0.194 (0.025)	0.218 (0.023)	0.192 (0.028)	0.192 (0.027)	**0.191** (0.027)	0.192 (0.028)
	rotate	0.227 (0.003)	**0.220** (0.001)	0.306 (0.004)	0.281 (0.003)	0.281 (0.002)	0.308 (0.004)
	scale	0.133 (0.006)	0.169 (0.006)	0.136 (0.006)	0.135 (0.007)	**0.131** (0.006)	0.136 (0.006)
	shear	**0.067** (0.001)	0.082 (0.002)	0.077 (0.002)	0.073 (0.002)	0.074 (0.002)	0.078 (0.002)
	shot noise	**0.118** (0.006)	0.131 (0.006)	0.138 (0.007)	0.130 (0.007)	0.128 (0.007)	0.138 (0.007)
	spatter	**0.068** (0.003)	0.081 (0.002)	0.083 (0.004)	0.077 (0.003)	0.077 (0.004)	0.083 (0.004)
	stripe	0.832 (0.244)	**0.750** (0.188)	1.460 (0.505)	1.186 (0.391)	1.250 (0.428)	1.485 (0.504)
	translate	0.940 (0.034)	**0.799** (0.030)	1.413 (0.074)	1.298 (0.060)	1.277 (0.065)	1.456 (0.077)
	zigzag	0.491 (0.027)	**0.422** (0.020)	0.770 (0.041)	0.665 (0.039)	0.654 (0.033)	0.753 (0.044)
ECE	brightness	0.025 (0.005)	0.062 (0.009)	0.008 (0.002)	0.010 (0.001)	0.009 (0.001)	**0.008** (0.001)
	canny edges	0.235 (0.019)	**0.164** (0.018)	0.292 (0.021)	0.280 (0.020)	0.271 (0.019)	0.293 (0.020)
	dotted line	**0.006** (0.001)	0.031 (0.001)	0.010 (0.000)	0.007 (0.000)	0.008 (0.000)	0.010 (0.000)
	fog	0.141 (0.010)	0.169 (0.010)	0.120 (0.009)	0.125 (0.009)	0.124 (0.009)	**0.118** (0.009)
	glass blur	0.013 (0.001)	0.049 (0.002)	0.014 (0.002)	**0.009** (0.000)	0.010 (0.001)	0.014 (0.002)
	impulse noise	**0.017** (0.006)	0.034 (0.003)	0.058 (0.009)	0.047 (0.008)	0.047 (0.008)	0.058 (0.009)
	motion blur	0.021 (0.003)	0.056 (0.002)	0.010 (0.002)	0.010 (0.001)	**0.009** (0.001)	0.009 (0.002)
	rotate	0.023 (0.001)	**0.015** (0.001)	0.044 (0.001)	0.040 (0.001)	0.040 (0.000)	0.044 (0.001)
	scale	0.010 (0.001)	0.058 (0.002)	0.012 (0.001)	0.009 (0.001)	**0.008** (0.001)	0.012 (0.001)
	shear	**0.005** (0.000)	0.025 (0.001)	0.010 (0.000)	0.008 (0.000)	0.009 (0.000)	0.010 (0.000)
	shot noise	**0.004** (0.001)	0.029 (0.001)	0.017 (0.001)	0.014 (0.001)	0.014 (0.001)	0.017 (0.001)
	spatter	**0.004** (0.000)	0.022 (0.001)	0.011 (0.000)	0.009 (0.000)	0.009 (0.001)	0.011 (0.000)
	stripe	0.107 (0.056)	**0.091** (0.051)	0.197 (0.069)	0.164 (0.062)	0.163 (0.065)	0.180 (0.064)
	translate	0.123 (0.006)	**0.057** (0.007)	0.180 (0.009)	0.173 (0.008)	0.170 (0.009)	0.184 (0.010)
	zigzag	0.067 (0.005)	**0.026** (0.003)	0.105 (0.007)	0.097 (0.007)	0.094 (0.006)	0.103 (0.007)
Brier	brightness	0.049 (0.009)	0.061 (0.011)	0.050 (0.010)	0.051 (0.010)	**0.049** (0.010)	0.050 (0.010)
	canny edges	0.592 (0.036)	**0.545** (0.033)	0.632 (0.039)	0.624 (0.038)	0.605 (0.035)	0.633 (0.038)
	dotted line	0.032 (0.001)	0.037 (0.001)	0.034 (0.001)	0.033 (0.001)	**0.032** (0.001)	0.033 (0.001)
	fog	0.128 (0.008)	0.140 (0.009)	0.121 (0.008)	0.123 (0.008)	0.122 (0.008)	**0.120** (0.008)
	glass blur	0.115 (0.001)	0.121 (0.001)	0.116 (0.002)	0.117 (0.001)	**0.115** (0.001)	0.116 (0.002)
	impulse noise	**0.160** (0.018)	0.162 (0.016)	0.173 (0.019)	0.169 (0.019)	0.162 (0.017)	0.170 (0.020)
	motion blur	0.092 (0.012)	0.097 (0.011)	0.090 (0.013)	0.091 (0.012)	0.090 (0.012)	**0.090** (0.013)
	rotate	0.109 (0.001)	**0.107** (0.001)	0.117 (0.001)	0.115 (0.001)	0.115 (0.001)	0.117 (0.001)
	scale	0.065 (0.003)	0.073 (0.003)	0.064 (0.003)	0.065 (0.004)	**0.063** (0.003)	0.064 (0.003)
	shear	**0.032** (0.001)	0.035 (0.001)	0.033 (0.001)	0.033 (0.001)	0.034 (0.001)	0.033 (0.001)
	shot noise	**0.055** (0.002)	0.057 (0.002)	0.056 (0.002)	0.057 (0.002)	0.055 (0.003)	0.056 (0.002)
	spatter	**0.034** (0.001)	0.036 (0.001)	0.035 (0.001)	0.035 (0.001)	0.035 (0.002)	0.035 (0.001)
	stripe	0.364 (0.102)	**0.358** (0.100)	0.473 (0.128)	0.403 (0.121)	0.398 (0.124)	0.415 (0.124)
	translate	0.394 (0.013)	**0.370** (0.013)	0.428 (0.017)	0.426 (0.016)	0.420 (0.017)	0.432 (0.018)
	zigzag	0.217 (0.013)	**0.205** (0.011)	0.244 (0.013)	0.234 (0.014)	0.230 (0.013)	0.237 (0.015)

Table 14: Performance of uncertainty quantification on corrupted MNIST for all subspace methods studied in this work (with $s = 100$) and the MAP solution (3) ($\hat{\theta}$). The standard error (in parantheses) is reported for 5 seeds.

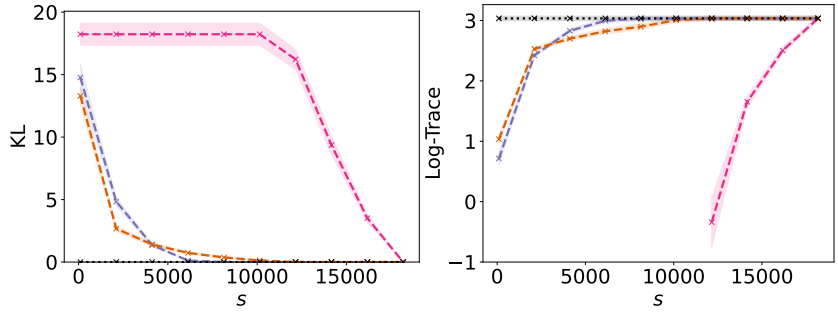
F BEHAVIOR FOR SMALLER AND LARGER s

Very Small s Values. Due to issues in computational complexity other methods in the literature, such as (Izmailov et al., 2019), use very small values of s. For completeness, we therefore report the performance of the lowrank and subset methdos in the very small s regime for the Red Wine dataset. Figures 4a and 4b show the KL-divergence 19 and logarithm of the trace criterion (20) for s between 1 and 20. We observe no fundamentaly different behavior in this regime. For these small s the approximation quality is insufficient as revealed by the KL-divergence. This is in line with the observation from Izmailov et al. (2019) that a temperature is needed in this regime to obtain reasonable uncertainties. In this work, we do not consider such modifications of the posterior.

Large Values of s. In Figure 1 it appears that $P_{\text{lowrank}-\text{Diagonal}}$ yields a KL-divergence that does not improve at all with increasing s for the Red Wine dataset. However, this is only due to the fact that we only consider s below 1000. Once we consider very large values of s the Kl divergence starts to drop finally around $s = 10,000$ as shown in Figure 4c. Note, that Figure 4c only shows subset methods as the low rank methods, introduced in this work, can only be used for s below the rank of the Jacobian (which is around 700 for Red Wine).



(a) Red Wine (small s) - KL divergence (b) Red Wine (small s) - Log-Trace



(c) Red Wine (large s) - KL-divergence (d) Red Wine (large s) - Log-Trace

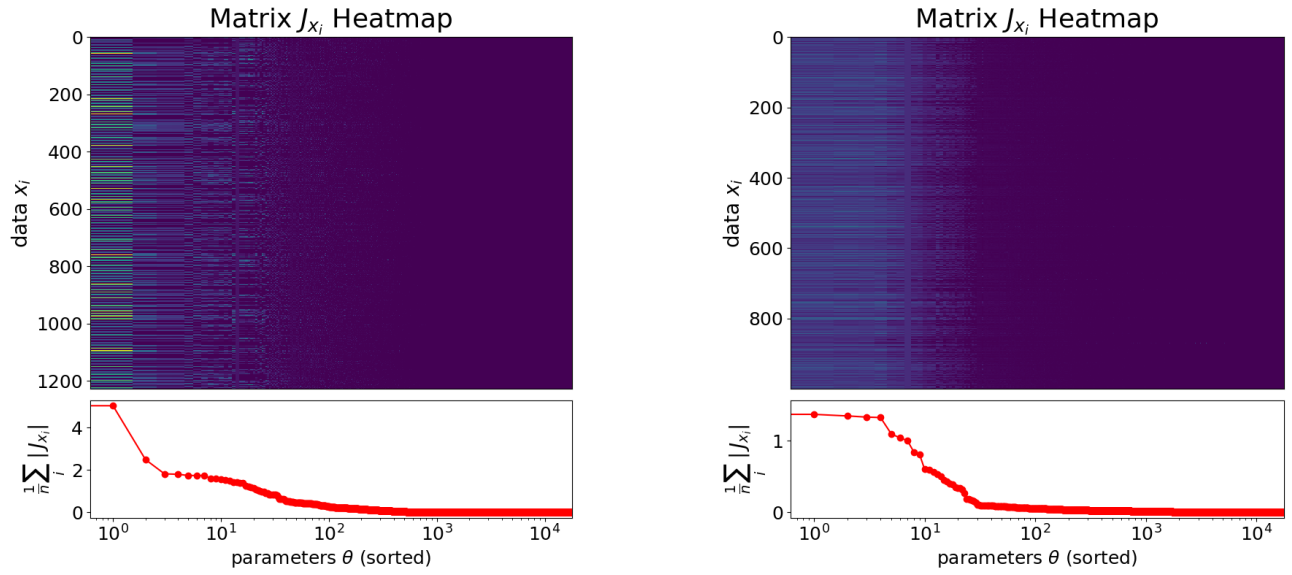
Figure 4: KL-divergence (19) (left column)) and logarithm of the trace 20 (right column) for the Red Wine dataset and very small (top row) and very large (bottom row) values of s. The colour coding is identical to the one in Figure 1. The bottom row does not show low rank methods as explained in Section F.

G DEAD PARAMETERS

Table 15: Relative number of parameters p that are insensitive to the input data for the regression datasets studied in this work. The standard deviation is taken over five seeds.

dataset	ENB	Red Wine	California	Naval
dead p	$92 \pm 1\%$	$89 \pm 2\%$	$60 \pm 2\%$	$34 \pm 4\%$

Note that our low rank methods allow for a wider class of subspace models as the subset methods, because they allow for linear combinations of the parameters instead of a simple selection. In fact, all subset solutions could be



(a) ENB

(b) California

Figure 5: The top displays a heatmap which highlights the activity of the gradients corresponding to the parameter θ_i for a subset of the test data for the regression datasets ENB (left) and Carlifornia (right). The parameters are sorted according to their sensitivity. A dark bluish colour implies that the gradient w.r.t. the data point x_i is negligible. The lower plot summarizes the magnitude of the sensitivity over all data points.

found by the low rank approximations, however, the opposite is not true. One reason why subset methods could nevertheless outperform low rank methods is that most of the parameters are irrelevant for a certain problem, i.e. have a gradient of zero w.r.t. the input. Indeed, Table 15 confirms this hypotheses, because the number of insensitive parameters positively correlates with an improved performance of the subset methods compared to the low rank methods. ENB, with most dead parameters, is the only experiment in which the subspace models are superior to the low rank subspace models according to the trace criterion (33), cf. Figure 1. For California or Naval Propulsion on the other hand, with much fewer dead parameters, low rank approximations clearly outperform subset approximations (cf. Figure 1).

This effect is visualized in Figure 5. The top displays a heatmap that highlights the sensitivity of parameters (the gradient w.r.t. the input) for a subset of the test dataset of ENB (left) and California (right). Light colours denote high sensitivity and dark colours low sensitivity. Below the average gradient of all data points w.r.t. a certain parameter is shown. For ENB both plots indicate that only a few parameters are highly responsive for most data points. If a subset method can capture these parameters, it shall perform well. In contrast, for California the sensitivity is more spread and hence, a linear combination can be more appropriate.

H FISHER INFORMATION, GENERALIZED GAUSS NEWTON AND HESSIAN

For the computations in our experiments we use the Fisher information matrix $\mathcal{I}$ instead of the generalized Gauss-Newton matrix H_{GGN} which are identical objects for the cases considered in this work Heskes (2000); Martens (2014); Dauphin et al. (2014). We summarize the relation between $\mathcal{I}$, H_{GGN} and the Hessian below in Section H.1 -H.4. For a more detailed analysis, we refer to the survey Martens (2014).

As follows from the identities in H.1 - H.4 we have $\mathcal{I} = VV^\intercal$ with a $V \in \mathbb{R}^{p \times NC}$ that can be computed via minibatches from $\mathcal{D}$ and expressions that involve first order derivatives of f_θ. This allows us to compute for any matrix $P \in \mathbb{R}^{p \times s}$ the expression

$$P^\intercal H_{\text{GGN}} P = P^\intercal \mathcal{I} P = (V^\intercal P)^\intercal V^\intercal P \in \mathbb{R}^{s \times s} \tag{34}$$

in a scalable manner if s is sufficiently small. Thus, while we often can't actually compute H_{GGN} or $\mathcal{I}$ in practice,

we can usually compute quadratic forms such as (34).

H.1 Fisher Information

H.1.1 Multivariate Regression

$$p(y_i|x_i,\theta) = \frac{1}{\sqrt{(2\pi)^C \det(\Sigma)}} e^{-\frac{1}{2}\|y_i - f(x_i,\theta)\|^2_{\Sigma^{-1}}}$$

is a common choice to model multivariate regression problems. For simplicity we assume that the covariance matrix $\Sigma \in \mathbb{R}^{C\times C}$ is independent of the parameter θ. The explicit form of the information matrix for a single input x_i is

$$\begin{aligned}\mathcal{I}_{kl}(x_i) &= \frac{1}{2}\mathbb{E}_{y\sim p(y|x_i,\theta)}\left[\partial_{\theta_k}\partial_{\theta_l}\|y_i - f_i\|^2_{\Sigma^{-1}}\right] \\ &= \sum_{c_1,c_2=1}^{C} \partial_{\theta_k} f_i^{c_1} \left(\Sigma^{-1}\right)^{c_1 c_2} \partial_{\theta_l} f_i^{c_2} \\ &= \left((\nabla_\theta f_i)^\intercal \, \Sigma^{-1} \nabla_\theta f_i\right)_{kl} .\end{aligned} \tag{35}$$

The abbreviation $f_i = f_\theta(x_i)$ is used for readability. For $\Sigma = \sigma^2 \mathbb{1}$ (as in this work), we obtain

$$\mathcal{I}_{kl}(x_i) = \sigma^{-2} \left((\nabla_\theta f_i)^\intercal \, \nabla_\theta f_i\right)_{kl} .$$

For the Fisher information matrix of the joint distribution we arrive at

$$\begin{aligned}\mathcal{I}_{kl} &= \frac{1}{2}\mathbb{E}_{(x,y)\sim p(y,x|\theta)}\left[\partial_{\theta_k}\partial_{\theta_l}\|y_i - f_i\|^2_{\sigma^{-2}\mathbb{1}}\right] \\ &\simeq \frac{\sigma^{-2}}{N}\sum_{i=1}^{N} \left((\nabla_\theta f_i)^\intercal \, \nabla_\theta f_i\right)_{kl} ,\end{aligned}$$

where in the last line $q(x)$ is approximated by $\hat{q}(x)$.

H.1.2 Softmax Classifier

For classification we consider the categorical distribution $y|x,\theta \sim \mathrm{Cat}(y|\phi(f_\theta(x))$ with probability vector

$$\phi_i^c = \phi^c(f_\theta(x_i)) = \frac{e^{f_\theta^c(x_i)}}{\sum_{\tilde{c}=1}^{C} e^{f_\theta^{\tilde{c}}(x_i)}} = \frac{e^{f_i^c}}{\sum_{\tilde{c}=1}^{C} e^{f_i^{\tilde{c}}}}.$$

The general form of the Fisher information matrix is given by

$$\begin{aligned}\mathcal{I}_{kl} &= \mathbb{E}_{(x,y)\sim p(x,y|\theta)}\left[\partial_{\theta_k} \ln p(x,y|\theta)\partial_{\theta_l} \ln p(x,y|\theta)\right] \\ &= \mathbb{E}_{y\sim p(y|x,\theta), x\sim q(x)}\left[\partial_{\theta_k} \ln p(y|x,\theta)\partial_{\theta_l} \ln p(y|x,\theta)\right] \\ &\simeq \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} \phi_i^c \partial_{\theta_k} \ln \phi_i^c \partial_{\theta_l} \ln \phi_i^c \\ &= \frac{4}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} \partial_{\theta_k}\sqrt{\phi_i^c}\,\partial_{\theta_l}\sqrt{\phi_i^c} ,\end{aligned}$$

where in the third line the empirical distribution $\hat{q}(x)$ is used to compute the expected value of the random variable x.

H.2 Relation Between Hessian and Fisher Information Matrix

Given the averaged log-likelihood $\frac{1}{N}\sum_i \ln p(y_i|f_\theta(x_i))$ of the data its Hessian w.r.t θ can be written as

$$\begin{aligned} H &= -\frac{1}{N}\sum_{i=1}^{N} \nabla_\theta^2 \ln p(y_i|f_\theta(x_i)) \\ &= \mathbb{E}_{(x,y)\sim\hat{q}(x,y)}\left[H_{-\ln p(y|x,\theta)}\right] \\ &= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{y\sim\hat{q}(y|x_i)}\left[H_{-\ln p(y|x_i,\theta)}\right] , \end{aligned} \tag{36}$$

where we wrote $H_{-\ln p(y|x,\theta)} = -\nabla_\theta^2 \ln p(y|f_\theta(x))$.

The Fisher information matrix $\mathcal{I}$ of $p(x,y|\theta)$ w.r.t the parameter θ is

$$\begin{aligned} \mathcal{I} &= \mathbb{E}_{(x,y)\sim p(x,y|\theta)}\left[\nabla_\theta \ln p(x,y|\theta)^\intercal \nabla_\theta \ln p(x,y|\theta)\right] \\ &= \mathbb{E}_{y\sim p(y|x,\theta),x\sim q(x)}\left[\nabla_\theta \ln p(y|x,\theta)^\intercal \nabla_\theta \ln p(y|x,\theta)\right] \\ &= -\mathbb{E}_{y\sim p(y|x,\theta),x\sim q(x)}\left[\nabla_\theta^2 \ln p(y|x,\theta)\right] \\ &= \mathbb{E}_{y\sim p(y|x,\theta),x\sim q(x)}\left[H_{-\ln p(y|x,\theta)}\right] . \end{aligned}$$

Since $q(x)$ is not analytically known, we shall use the empirical distribution $\hat{q}(x)$ instead.

$$\mathcal{I} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{y\sim p(y|x=x_i,\theta)}\left[H_{-\ln p(y|x,\theta)}\right] . \tag{37}$$

The equations (37) and (36) are quite similar. The difference is the distribution under which the expectation is computed. However, note that (36) and (37) are different from the empirical Fisher information matrix

$$\begin{aligned} \mathcal{I}_{\text{empirical}} &= \mathbb{E}_{y\sim\hat{q}(x,y|\theta)}\left[\nabla_\theta \ln p(x,y|\theta)^\intercal \nabla_\theta \ln p(x,y|\theta)\right] \\ &= \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \ln p(y_i|x_i,\theta)^\intercal \nabla_\theta \ln p(y_i|x_i,\theta). \end{aligned}$$

H.3 Relation Between Hessian and Generalized Gauss-Newton Matrix

The generalized Gauss-Newton matrix is often used as a substitute of the Hessian because it is positive semi-definite and easier to compute Martens (2014). For generalized linear models both quantities coincide. Let us write the Jacobian w.r.t. the log-likelihood as $\nabla_\theta \ln p(y_i|f_\theta(x_i)) = \nabla_{f_i} \ln p(y_i|f_i)\nabla_\theta f_\theta(x_i) = \nabla_{f_i} \ln p(y_i|f_i) J_{f_i}$ and $H_{f_i^c} = \nabla_\theta^2 f_\theta(x_i)^c$ for $1 \le c \le C$. Then the Hessian can be decomposed into

$$\begin{aligned} H =& \frac{-1}{N}\sum_{i=1}^{N}\Big(J_{f_i}^\intercal \nabla_{f_i}^2 \ln p(y_i|f_i) J_{f_i} \\ &+ \sum_{c=1}^{C} H_{f_i^c} \partial_{f_i^c} \ln p(y_i|f_i)\Big) \\ =& H_{\text{GGN}} - \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} H_{f_i^c}\partial_{f_i^c} \ln p(y_i|f_i) \end{aligned} \tag{38}$$

with the generalized Gauss-Newton matrix

$$\begin{aligned} H_{\text{GGN}} &= -\frac{1}{N}\sum_{i=1}^{N} J_{f_i}^\intercal \nabla_{f_i}^2 \ln p(y_i|f_i) J_{f_i} \\ &= \frac{1}{N}\sum_{i=1}^{N} J_{f_i}^\intercal H_{-\ln p(y_i|f_i)} J_{f_i} . \end{aligned} \tag{39}$$

A sufficient condition that the generalized Gauss-Newton matrix and the Hessian coincide is that the model is linear, because for linear models $H_{f_i^c} = 0$ for $1 \leq c \leq C$. In the definition of the generalized Gauss-Newton matrix a choice about where the cut between the loss and the network function has to be made. This is to some degree arbitrary, however, Schraudolph (2002) recommends to perform as much as possible of the computation in the loss such that $\ln p(y_i|f)$ is still convex to ensure positive semi-definiteness of H_{GGN}.

H.4 Relation Between Fisher Information Matrix and Generalized Gauss-Newton Matrix

Rewriting $\nabla_\theta \ln p(y_i|f_\theta(x_i)) = \nabla_{f_i} \ln p(y_i|f_i) J_{f_i}$ the Fisher information matrix is of the form

$$\begin{aligned}\mathcal{I} &= \mathbb{E}_{y\sim p(y|x,\theta), x\sim q(x)} \left[\nabla_\theta \ln p(y|x,\theta)^\intercal \nabla_\theta \ln p(y|x,\theta)\right] \\ &= \mathbb{E}_x \left[J_f^\intercal \mathbb{E}_y \left[\nabla_f \ln p(y|f)^\intercal \left(\nabla_f \ln p(y|f)\right)\right] J_f\right] \\ &:= \mathbb{E}_x \left[J_f^\intercal \mathcal{I}_{\ln p(y|f)} J_f\right] ,\end{aligned} \tag{40}$$

where we write shorthand $f = f_\theta(x)$ and

$$\begin{aligned}\mathcal{I}_{\ln p(y|f)} &= \mathbb{E}_y \left[\nabla_f \ln p(y|f)^\intercal \nabla_f \ln p(y|f)\right] \\ &= -\mathbb{E}_y \left[\nabla_f^2 \ln p(y|f)\right] \\ &= \mathbb{E}_y \left[H_{-\ln p(y|f)}\right]\end{aligned}$$

is the "Fisher information matrix of the predictive distribution".

From these two identities it easily follows that if we substitute $q(x)$ by its empirical distribution $\hat{q}(x)$, the generalized Gauss-Newton matrix (39) is identical to the Fisher information matrix (40) if $H_{-\ln p(y|f)}$ is constant in y. This is the case for squared error loss and cross-entropy loss Heskes (2000); Pascanu and Bengio (2013); Martens (2014). Indeed, for squared error loss we have

$$H_{-\ln p(y|f)} = \nabla_f^2 \frac{1}{2} \|f - y\|^2_{\Sigma^{-1}} = \Sigma^{-1}$$

and for cross-entropy loss we obtain

$$\begin{aligned}H_{\ln p(y|f);c'c''} &= \partial_{f^{c'}} \partial_{f^{c''}} \sum_{c=1}^{C} y^c \ln \phi^c \\ &= \partial_{f^{c'}} \partial_{f^{c''}} \sum_{c=1}^{C} y^c \ln \frac{e^{f^c}}{\sum_{\tilde{c}=1}^{C} e^{f^{\tilde{c}}}} \\ &= \partial_{f^{c'}} \partial_{f^{c''}} \left(\sum_{c=1}^{C} y^c f^c - \sum_c y^c \ln \sum_{\tilde{c}=1}^{C} e^{f^{\tilde{c}}}\right) \\ &= \partial_{f^{c'}} \partial_{f^{c''}} \left(\sum_{c=1}^{C} y^c f^c - \ln \sum_{\tilde{c}=1}^{C} e^{f^{\tilde{c}}}\right) \\ &= -\partial_{f^{c'}} \partial_{f^{c''}} \ln \sum_{\tilde{c}=1}^{C} e^{f^{\tilde{c}}} = -\partial_{f^{c''}} \phi^{c'} \\ &= -\delta_{c'c''} \phi^{c'} + \phi^{c'} \phi^{c''} ,\end{aligned}$$

which are both constant in y.