Bayes without Underfitting: Fully Correlated Deep Learning Posteriors via Alternating Projections

Marco Miani[†]
Technical University of Denmark
mmia@dtu.dk

Hrittik Roy[†]
Technical University of Denmark
hroy@dtu.dk

Søren Hauberg
Technical University of Denmark
sohau@dtu.dk

Abstract

Bayesian deep learning all too often underfits so that the Bayesian prediction is less accurate than a simple point estimate. Uncertainty quantification then comes at the cost of accuracy. For linearized models, the null space of the generalized Gauss-Newton matrix corresponds to parameters that preserve the training predictions of the point estimate. We propose to build Bayesian approximations in this null space, thereby guaranteeing that the Bayesian predictive does not underfit. We suggest a matrix-free algorithm for projecting onto this null space, which scales linearly with the number of parameters and quadratically with the number of output dimensions. We further propose an approximation that only scales linearly with parameters to make the method applicable to generative models. An extensive empirical evaluation shows that the approach scales to large models, including vision transformers with 28 million parameters. Code is available at:
https://github.com/h-roy/projected-bayes

1 Underfitting in Bayesian deep learning

Bayesian deep learning tends to underfit. Numerous studies demonstrate that marginalizing approximate weight posteriors yields less accurate predictions than applying a *maximum a posteriori* (MAP) point estimate [Wenzel et al., 2020, Daxberger et al., 2021a, Zhang et al., 2024, Kristiadi et al., 2022]. This significantly reduces the practical value of Bayesian deep learning,
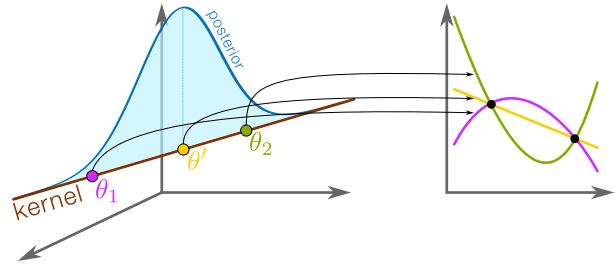

[†] Equal contribution.

Figure 1: *Key idea:* In overparametrized linear models, the *kernel* (null space) contains all models that have identical predictions on the training data. We propose restricting approximate posteriors of deep neural networks to this kernel to avoid underfitting.

despite having attractive theoretical properties [Germain et al., 2016]. To counteract this trend, we first explicate *why* underfitting happens with Gaussian approximate posteriors, and thereafter propose a solution for low-noise data (e.g. images).

What is underfitting? Deep learning performs very well when the training data is subject to limited observation noise. In contrast, Bayesian deep learning often *underfits* in the sense that the Bayesian prediction deviates significantly from a *point estimate* prediction *on the training data* $\mathcal{D}$, i.e.

$$\mathbb{E}_{\boldsymbol{\theta} \sim q}[f(\boldsymbol{\theta}, \mathbf{x})] \neq f(\boldsymbol{\theta}_{\text{MAP}}, \mathbf{x}) \qquad \text{for } \mathbf{x} \in \mathcal{D}, \quad (1)$$

where q is an approximate posterior and $\boldsymbol{\theta}_{\text{MAP}}$ is a point estimate of the neural network f. This notion of underfitting only applies to low-noise data, where the posterior should have little uncertainty on the training data, but this is the most prominent scenario in deep learning.

Why do Bayesian neural networks underfit? Deep neural networks are commonly *overparametrized*, i.e. the model has more parameters than observations, as this tends to improve both optimization and generalization [Allen-Zhu et al., 2019]. Fig. 1 sketches the situation which we later formalize: Consider fitting a quadratic function with three parameters to only two observations. This leaves one degree of freedom

undetermined and a linear subspace of the model parameters exists wherein all parameters yield identical predictions on the training data. For low-noise data, the true posterior will concentrate around this subspace. Unfortunately, common Gaussian approximations to the posterior are axis-aligned (i.e. *mean field* approximations [Bishop, 2006]) and not aligned with the mentioned subspace. This mismatch leads to a poor posterior approximation that tends to underfit. *We suggest, quite simply, to project the approximate posterior to the subspace in which underfitting cannot happen.*

Why is this beneficial? Our proposed posterior reflects the degrees of freedom in the model that fundamentally cannot be determined by even noise-free data. Predicting according to this distribution gives reliable out-of-distribution detection and general uncertainty quantification without underfitting. Our approach captures correlations between layers and retains high-accuracy predictions while scaling linearly with model size. Table 1 contrasts our method with existing ones.

Why is this difficult? We will soon see that the relevant subspace on which to project is given by the *kernel* (i.e. *null space*) of a matrix that is quadratic in the number of model parameters. Even for models of modest size, this matrix is too large to be stored in memory and direct projection methods cannot be applied. *We propose a linear-time sampling algorithm that can be implemented entirely using automatic differentiation.*

Paper outline. Sec. 2 gives the background to derive our approach. A wider discussion of related work is postponed to Sec. 5. We develop our approach in two steps. First, Sec. 3 describes our proposed posterior approximation, while Sec. 4 derives an efficient sampling algorithm. Empirical investigations are conducted in Sec. 6.

2 Background and notation

Notation. Let $f : \mathbb{R}^P \times \mathbb{R}^I \to \mathbb{R}^O$ denote a neural network with parameter $\boldsymbol{\theta} \in \mathbb{R}^P$ that maps inputs $\mathbf{x} \in \mathbb{R}^I$ to outputs $\mathbf{y} \in \mathbb{R}^O$. We define the per-datum Jacobian as $\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}) = \partial_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \mathbf{x}) \in \mathbb{R}^{O \times P}$ and its stacked counterpart $\mathbf{J}_{\boldsymbol{\theta}} = [\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_1); \ldots; \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_N)] \in \mathbb{R}^{NO \times P}$ for a fixed training dataset $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$.

Gaussian approximate posteriors, i.e. $p(\boldsymbol{\theta}|\mathcal{D}) \approx q(\boldsymbol{\theta}|\mathcal{D}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, are ever-present in Bayesian deep learning [Blundell et al., 2015, Botev et al., 2017, Sharma et al., 2021, Khan et al., 2018, Maddox et al., 2019, Stephan et al., 2017, Osawa et al., 2019, Antorán et al., 2022]. For modern neural networks, the parameter dimension P can easily be several billions, such that even storing the covariance $\boldsymbol{\Sigma}$ in memory is infeasible. To allow for linear scaling, common Gaussian approximate posteriors are realized by disregarding

most correlations in $\boldsymbol{\Sigma}$, e.g. with diagonal or block-diagonal approximations.

Unfortunately, there is significant theoretical evidence against disregarding correlations. Foong et al. [2019] shows that diagonal approximations underestimate 'in-between' uncertainty, while Roy et al. [2024] shows that uncorrelated models cannot generally be infinitesimally invariant to reparametrizations even if this property is held by the true posterior.

The *linearized Laplace approximation* (*LLA*; Immer et al. [2021b], Khan et al. [2019]) use a first-order Taylor expansion of $f(\boldsymbol{\theta}, \mathbf{x}) \approx f_{\mathrm{lin}}^{\boldsymbol{\theta}_{\mathrm{MAP}}}(\boldsymbol{\theta}, \mathbf{x}) = f(\boldsymbol{\theta}_{\mathrm{MAP}}, \mathbf{x}) + \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathbf{x})(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}})$, where $\boldsymbol{\theta}_{\mathrm{MAP}}$ is the expansion point (usually a MAP estimate of the parameter). Secondly, LLA perform a standard Laplace approximation [MacKay, 1992] of the linearized model, yielding

$$q_{\mathrm{LLA}}(\boldsymbol{\theta}|\mathcal{D}) = \mathcal{N}\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{\mathrm{MAP}}, (\alpha \mathbb{I} + \mathrm{GGN}_{\boldsymbol{\theta}_{\mathrm{MAP}}})^{-1}\right) \quad (2)$$

$$\mathrm{GGN}_{\boldsymbol{\theta}_{\mathrm{MAP}}} = \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}^\top \mathbf{H}_{\boldsymbol{\theta}_{\mathrm{MAP}}} \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}} \in \mathbb{R}^{P \times P}. \quad (3)$$

Here $\mathrm{GGN}_{\boldsymbol{\theta}_{\mathrm{MAP}}}$ is the so-called *generalized Gauss-Newton matrix*, α is the prior precision, $\mathbf{H}_{\boldsymbol{\theta}}(\mathbf{x}) = -\partial_{f(\boldsymbol{\theta}, \mathbf{x})}^2 \log p(\mathbf{y}|f(\boldsymbol{\theta}, \mathbf{x})) \in \mathbb{R}^{O \times O}$ and $\mathbf{H}_{\boldsymbol{\theta}} \in \mathbb{R}^{NO \times NO}$ is its stacked counterpart. This Hessian takes a simple closed-form for common likelihoods, e.g. it is the identity for Gaussian likelihoods [Immer et al., 2021b]. For notational simplicity, we treat this Hessian as an identity for the remainder of this paper.

The LLA has a strong empirical performance [Immer et al., 2021b], but it is computationally taxing as the GGN has size $P \times P$. This is infeasible to instantiate even for models of modest size, and the $\mathcal{O}(P^3)$ cost associated with sampling presents a significant computational challenge. Iterative matrix-free solvers can potentially alleviate these concerns [Roy et al., 2024], but the GGN is highly ill-conditioned [Papyan, 2020, Miani et al., 2024] and this approach requires overcoming issues of numerical stability. Practical LLA implementations therefore resort to sparse, e.g. diagonal or block-diagonal, approximations of the GGN.

3 The proposed approximate posterior

We next propose a fully correlated Gaussian posterior that is guaranteed to not underfit. Unless otherwise stated, the presented results are novel contributions and proofs of all theorems can be found in the appendix.

As alluded to, we propose restricting the posterior covariance to a particular subspace of the parameter space. Letting $\mathbf{U} \in \mathbb{R}^{P \times R}$ denote an orthogonal basis of this subspace, we consider an *isotropic* model therein,

$$q_{\mathrm{PROJ}}(\boldsymbol{\theta}|\boldsymbol{\theta}_{\mathrm{MAP}}, \mathcal{D}) = \mathcal{N}\left(\boldsymbol{\theta}_{\mathrm{MAP}}, \alpha^{-1} \mathbf{U}\mathbf{U}^\top\right). \quad (4)$$

Marco Miani[†]**, Hrittik Roy**[†]**, Søren Hauberg**

Table 1: Comparisons between Laplace approximations. N: number of data points, O: output dimensions, P: number of parameters, $P_{l,in}$, $P_{l,out}$: input and output dimensions of layer l, P_{ll}: last layer parameters.

METHOD	CORRELATION STRUCTURE	SPACE COMPLEXITY	PER-SAMPLE TIME COMPLEXITY	PREPROCESSING TIME COMPLEXITY	ERROR BOUND	MODEL AGNOSTIC	TRACTABLE optimal LML
Diagonal		P	P	PNO	✗	✓	✗
Kronecker-Factored		$\sum_{l=1}^{L} P_{l,in}^2 + P_{l,out}^2$	$\sum_{l=1}^{L} P_{l,in}^2 + P_{l,out}^2$	$\sum_{l=1}^{L} NP_{l,in} + NP_{l,out} + P_{l,in}^3 + P_{l,out}^3$	✗	✗	✗
Last-Layer		P_{ll}^2	P_{ll}^2	$O^2 NP_{ll}^2 + P_{ll}^3$	✗	✓	✗
This paper		$P + NO^2$ or $P + N$	$t_{\max} PN + NO^3$ or $t_{\max} PN$	0	✓ (Lemmas 3.5 & 4.3)	✓	✓ (Sec. 3)

Specifically, we choose the above subspace as the *kernel* (i.e. *null space*) of the GGN matrix and call the result the *projected posterior*. Formally,

$$\mathbf{U}\mathbf{U}^\top = \mathbb{I}_P - \mathcal{P}(\text{GGN}) = \mathbb{I}_P - \mathbf{V}\,[\mathbf{D}\!>\!0]\,\mathbf{V}^\top, \quad (5)$$

where $\mathbf{V}\mathbf{D}\mathbf{V}^\top$ is an eigen decomposition of the GGN matrix and $[\mathbf{D}\!>\!0] \in \mathbb{R}^{P\times P}$ is diagonal with elements 1 where $\mathbf{D}$ holds positive values. We use the shorthand notation $\mathcal{P}(\text{GGN})$ to denote the projection onto the image of the GGN. *We next justify the projected posterior through the lens of underfitting.*

The projected posterior never underfits. When using the linearized neural network, $f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}) = f(\boldsymbol{\theta}_{\text{MAP}}, \mathbf{x}) + \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x})(\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}})$, we avoid underfitting when $\mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x})(\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}) = \mathbf{0}$ on the training data. The linearized model then avoids underfitting when $\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}$ is restricted to the kernel (null space) of $\mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}} = [\mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_1); \ldots; \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_N)]$. For the proposed projected posterior (4), we, thus, choose $\mathbf{U}$ as an orthonormal basis of the Jacobian kernel and note that this kernel coincides with the zero eigenvectors of the GGN.

These considerations can be formalized as follows.

Lemma 3.1. *The projected posterior (4) is supported on equal functions on the training data, i.e. $\forall \mathbf{x} \in \mathcal{D}$*

$$f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}) = f(\boldsymbol{\theta}_{\text{MAP}}, \mathbf{x}) \qquad \forall \boldsymbol{\theta} \sim q_{\text{PROJ}}, \quad (6)$$

which implies that $\text{VAR}_{\boldsymbol{\theta} \sim q_{\text{PROJ}}} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}) = 0$.

We emphasize that the statement *only holds on the training data*. Elsewhere, the approximate posterior yields strictly positive variances with high probability under strong but reasonable assumptions.

Lemma 3.2. *Let* $\mathbf{J}_{\boldsymbol{\theta}} = [\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_1)^\top \ldots \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_N)^\top]^\top$ *and* $\mathbf{x}_{\text{test}} \in \mathbb{R}^I$, *then*

$$\text{VAR}_{\boldsymbol{\theta} \sim q_{\text{PROJ}}} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_{\text{test}}) > 0 \quad (7)$$

if $\text{RANK}\left(\begin{pmatrix} \mathbf{J}_{\boldsymbol{\theta}_{MAP}} \\ \mathbf{J}_{\boldsymbol{\theta}_{MAP}}(\mathbf{x}_{\text{test}}) \end{pmatrix} \begin{pmatrix} \mathbf{J}_{\boldsymbol{\theta}_{MAP}} \\ \mathbf{J}_{\boldsymbol{\theta}_{MAP}}(\mathbf{x}_{\text{test}}) \end{pmatrix}^\top \right) > \text{RANK}\left(\mathbf{J}_{\boldsymbol{\theta}_{MAP}} \mathbf{J}_{\boldsymbol{\theta}_{MAP}}^\top \right)$.

Jointly the two lemmas show that out-of-distribution data is guaranteed to have higher predictive variance than the training data. The (technical) rank assumption in Lemma 3.2 is known to be satisfied with high probability under reasonable assumptions [Bombari et al., 2022, Nguyen et al., 2021, Karhadkar et al., 2024].

These results can be contrasted with LLA, where we can show the following result.

Theorem 3.3. *For $\alpha > 0$, the predictive variance of LLA on any training datapoint is positive and bounded,*

$$\frac{O\gamma^2}{\gamma^2 + \alpha} \leq \text{VAR}_{\theta \sim q_{\text{LLA}}} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}) \leq \frac{O\lambda^2}{\lambda^2 + \alpha} \quad for\ \mathbf{x} \in \mathcal{D}.$$

Here, λ and γ are the largest and smallest singular values of the dataset Jacobian $\mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}$, respectively.

Here, γ is strictly positive with high probability under reasonable assumption [Bombari et al., 2022, Nguyen et al., 2021, Karhadkar et al., 2024]. *LLA then underfits with high probability, which contrasts our approach (c.f. Lemma 3.1).* Note that this is only true for the (intractable) GGN-based covariance. We expect that sparse approximations to this covariance, e.g. diagonal and KFAC, has even higher training set variance.

Underfitting in existing models. The above analysis justifies the projected posterior and also sheds light on why current Bayesian approximations often underfit. For efficiency, mean field approximations of the posterior covariance are quite common, e.g. *diagonal* or *Kronecker factored* covariances [Ritter et al., 2018, Martens and Grosse, 2015]. These approximations will generally sample outside the GGN kernel, implying strictly positive predictive variances on the training data even for noise-free data. This reasoning also motivates recent works that have attempted to improve these approximations by making their spectrum more aligned with the spectrum of the GGN [George et al., 2018, Dhahri et al., 2024].

Computational benefits. Beyond the above theoretical motivations, our projected covariance also brings computational benefits. Since $\mathbf{U}$ is an orthonormal basis, the covariance $\mathbf{U}\mathbf{U}^\top$ is a projection matrix, implying that its eigenvalues are all 0 or 1. This, in turn, implies that $\mathbf{U}\mathbf{U}^\top = (\mathbf{U}\mathbf{U}^\top)^2 = (\mathbf{U}\mathbf{U}^\top)^\top = (\mathbf{U}\mathbf{U}^\top)^\dagger$, where $\dagger$ denotes pseudo-inversion. Drawing samples from q, thus, only requires matrix-vector products with $\mathbf{U}\mathbf{U}^\top$ and the matrix need not be instantiated. The only open question is then how to efficiently perform such matrix-vector products. We answer this in Sec. 4.

Since all eigenvalues of $\mathbf{U}\mathbf{U}^\top$ are 0 or 1, we can a priori expect the matrix to yield numerically stable compu-

tations even at moderate numerical precision. This is in contrast to the GGN, which easily has condition numbers that exceed 10^6 [Miani et al., 2024].

Tractable model selection. A benefit of Bayesian neural networks is that we can choose α by maximizing the marginal likelihood, $p(\mathcal{D}|\alpha)$, on the training data, i.e. without relying on validation data [MacKay, 1995]. However, since the marginal likelihood of the true posterior is intractable, it is common to consider that of the Laplace approximation [Immer et al., 2021a]

$$\log q_{\text{LLA}}(\mathcal{D}|\alpha) = \log p(\mathcal{D}|\boldsymbol{\theta}_{\text{MAP}}) + \log p(\boldsymbol{\theta}_{\text{MAP}}|\alpha) \\ - \frac{1}{2}\log\det\left(\frac{1}{2\pi}\left(\text{GGN} + \alpha\mathbb{I}\right)\right). \quad (8)$$

This can then be optimized numerically to estimate α. In contrast, the projective posterior (4) does not require optimization as α is available in closed form.

Lemma 3.4. *The marginal likelihood for the projected posterior (4) has a globally optimal α given by*

$$\alpha^* = \frac{\|\boldsymbol{\theta}_{\text{MAP}}\|^2}{P - \text{Tr}(\mathbb{I}_P - \mathcal{P}\left(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}\right))}. \quad (9)$$

In practice, the trace can be approximated using Hutchinson's estimator [1989].

Links to LLA. The projected posterior can be viewed as a fully correlated tractable approximation to the LLA. The following statement shows that the difference between the two approximate posteriors is bounded.

Lemma 3.5. *Let τ denote the smallest non-zero eigenvalue of the GGN [Nguyen et al., 2021], then*

$$\left\|\underbrace{(\alpha\mathbb{I}_P + \text{GGN})^{-1}}_{\text{LLA cov}} - \underbrace{\alpha^{-1}(\mathbb{I}_P - \mathcal{P}\left(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}\right))}_{\text{Proj cov (scaled)}}\right\| \leq \frac{1}{\tau + \alpha}$$

Additionally, let k denote the GGN rank and d the W_2 Wasserstein distance, then $d^2(q_{\text{LLA}}, q_{\text{PROJ}}) \leq (\tau + \alpha)^{-1}k$.
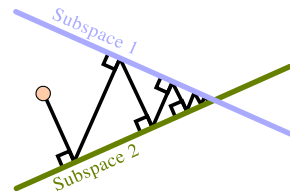
4 Sampling via alternating projections

We next develop a novel scalable algorithm to sample the projected posterior. Recall that we can simulate this posterior as $\mathbf{U}\mathbf{U}^\top\boldsymbol{\epsilon} + \boldsymbol{\theta}_{\text{MAP}}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1}\mathbb{I})$ and $\mathbf{U}\mathbf{U}^\top$ projects to the GGN kernel. In general, the orthogonal projection onto a subspace spanned by the columns of any matrix $\mathbf{M} \in \mathbb{R}^{R \times P}$, where $R < P$, is

$$\mathcal{P}\left(\mathbf{M}^\top\mathbf{M}\right) = \mathbf{M}^\top\left(\mathbf{M}\mathbf{M}^\top\right)^{-1}\mathbf{M} \in \mathbb{R}^{P \times P}. \quad (10)$$

The projection into the kernel of $\mathbf{M}^\top\mathbf{M}$ is then given by the matrix $\mathbb{I} - \mathcal{P}\left(\mathbf{M}^\top\mathbf{M}\right)$. In our case $\mathbf{M} = \mathbf{J}_{\boldsymbol{\theta}}\mathbf{H}_{\boldsymbol{\theta}}^{1/2} \in \mathbb{R}^{NO \times P}$ and Eq. 10 requires inverting a $NO \times NO$ matrix, which is infeasible even for moderate datasets.

Figure 2: To project a point onto the intersection of two subspaces, we alternate between projecting onto the individual subspaces.

Intersections of kernels. To arrive at a feasible algorithm, we note that the kernel of $\mathbf{M}^\top\mathbf{M}$ is the kernel of a sum of positive semi-definite matrices,

$$\ker\left(\sum_b \mathbf{M}_b^\top\mathbf{M}_b\right) = \bigcap_b \ker(\mathbf{M}_b^\top\mathbf{M}_b), \quad (11)$$

where we have decomposed $\mathbf{M}$ into batches,

$$\mathbf{M} = \begin{pmatrix}\mathbf{M}_1 \\ \vdots \\ \mathbf{M}_B\end{pmatrix} \qquad \mathbf{M}^\top\mathbf{M} = \sum_{b=1}^B \mathbf{M}_b^\top\mathbf{M}_b. \quad (12)$$

Hence, we can project onto the GGN kernel by projecting onto the *intersection* of *per batch*-GGN kernels.

von Neumann's [1949] alternating projections algorithm projects onto intersections of subspaces. Fig. 2 illustrates this idea, formalized in Lemma 4.1.

Lemma 4.1. *For any row-partition of a matrix $\mathbf{M} = \begin{pmatrix}\mathbf{M}_1^\top & \cdots & \mathbf{M}_B^\top\end{pmatrix}^\top$ it holds that*

$$\mathbb{I} - \mathcal{P}\left(\mathbf{M}^\top\mathbf{M}\right) = \lim_{t\to\infty}\left(\prod_b\left(\mathbb{I} - \mathcal{P}(\mathbf{M}_b^\top\mathbf{M}_b)\right)\right)^t. \quad (13)$$

Moreover, the limit converges linearly with a rate $c = \prod_{b=1}^B\cos^2(\theta_b)$ where $\theta_b = \min_{b'\neq b}\angle(M_b, M_{b'})$ and $\angle(M_b, M_{b'})$ is the minimum angle between linear combinations of rows of M_b and linear combinations of rows of $M_{b'}$.

In practice, we stop the algorithm after a number of iterations, so that the limit (13) stops at $t = t_{\max}$. The induced error is then upper bound by $c^{t_{\max}}$.

Projection for the approximate posterior. The algorithm projects to the GGN kernel, by iteratively projecting to the kernels of per-batch GGNs. Using Eq. 10, a single step of the algorithm requires inverting an $SO \times SO$ matrix, which is computationally cheap for small batch-sizes S. Fig. 3 illustrates this pipeline.

Implementation details. For sufficiently small batches, we can precompute and store the inverses of $\mathbf{M}_b\mathbf{M}_b^\top \in \mathbb{R}^{SO \times SO}$. Thereafter, matrix-vector products with $\mathbf{M}$ and $\mathbf{M}^\top$ can be efficiently computed using Jacobian-vector products and vector-Jacobian products, respectively. We, thus, do not need to instantiate the per-batch GGN matrices, which gives rise to an efficient low-memory sampling algorithm.

Complexity. The space complexity of the algorithm is $\mathcal{O}(P + NSO^2)$ and time complexity is $\mathcal{O}(t_{\max}PN +$

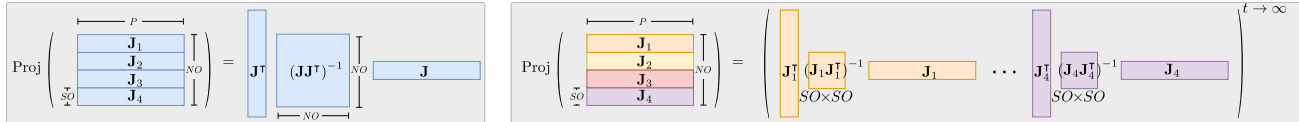**Marco Miani**[†], **Hrittik Roy**[†], **Søren Hauberg**

Figure 3: **Visualization of Jacobian projections** Direct calculation of the projection(left) involves inverting a large $NO \times NO$ matrix. This is replaced by an infinite (in practice, truncated) series of cheap projections (right) which only require precomputing and storing inverses of several small $SO \times SO$ matrices.

NS^2O^3). For overparametrized models where $P \gg NO$ these complexities boil down to $\mathcal{O}(P)$ and $\mathcal{O}(t_{\max}PN)$, respectively. This matches the cost of training for $t_{\max}$ epochs. Consequently, the projection is feasible on hardware that can train a model.

4.1 Scaling to high-dimensional outputs

The computational complexity of the projection algorithm scales superlinearly with the model's output size. For many tasks this is unproblematic, but it renders the method inapplicable to e.g. image-generative models.

To handle high-dimensional outputs, we propose removing the requirement of preserving the predictions of $\boldsymbol{\theta}_{\text{MAP}}$ at the training data. Instead, we propose to (locally) preserve the *loss* at each training point. This is achieved by considering the loss-Jacobian, i.e. the stacking of the per-datum loss-gradients.

$$\mathbf{J}_{\boldsymbol{\theta}}^L = \begin{pmatrix} \nabla_{\boldsymbol{\theta}} l(f(\boldsymbol{\theta}, \mathbf{x}_1), \mathbf{y}_1) \\ \vdots \\ \nabla_{\boldsymbol{\theta}} l(f(\boldsymbol{\theta}, \mathbf{x}_N), \mathbf{y}_N) \end{pmatrix} \in \mathbb{R}^{N \times P} \qquad (14)$$

Lemma 4.2. *The kernel of the stacked loss gradients $\mathbf{J}_{\boldsymbol{\theta}}^L \in \mathbb{R}^{N \times P}$ contains the kernel of the full Jacobian $\mathbf{J}_{\boldsymbol{\theta}} \in \mathbb{R}^{NO \times P}$, i.e. $\ker(\mathbf{J}_{\boldsymbol{\theta}}) \subseteq \ker(\mathbf{J}_{\boldsymbol{\theta}}^L)$. Further, note that these subspaces are identical for $O = 1$.*

Intuitively, for each datapoint, the gradient is a linear combination of its Jacobian rows, namely $\nabla_{\boldsymbol{\theta}} l(f(\boldsymbol{\theta}, \mathbf{x}), \mathbf{y}) = \nabla_{f(\boldsymbol{\theta}, \mathbf{x})} l(f(\boldsymbol{\theta}, \mathbf{x}), \mathbf{y}) \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x})$. This approach can, thus, be seen as aggregating each per-datum Jacobian into a single meaningful row, lowering the row count by a factor O.

We propose the *loss-projected approximate posterior*

$$q_{\text{LOSS}}(\boldsymbol{\theta}|\boldsymbol{\theta}_{\text{MAP}}, \mathcal{D}) = \mathcal{N}\left(\boldsymbol{\theta}_{\text{MAP}}, \alpha^{-1}\mathbf{U}_L\mathbf{U}_L^\top\right), \qquad (15)$$

where $\mathbf{U}_L \in \mathbb{R}^{P \times R}$ denotes an orthogonal basis of the kernel of $\mathbf{J}_{\boldsymbol{\theta}}^L$. Samples from this approximate posterior are guaranteed to have the same per-datum loss as the mode parameter, up to first order,

Lemma 4.3. *For any $\theta \sim q_{\text{LOSS}}$ and $\mathbf{x}_n \in \mathcal{D}$ it holds*

$$|l(f(\boldsymbol{\theta}, \mathbf{x}_n), \mathbf{y}_n) - l(f(\boldsymbol{\theta}_{\text{MAP}}, \mathbf{x}_n), \mathbf{y}_n)| = \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}\|^2)$$

which implies that

$$\text{VAR}_{\boldsymbol{\theta} \sim q_{\text{LOSS}}} l(f(\boldsymbol{\theta}, \mathbf{x}_n), \mathbf{y}_n) \leq \mathcal{O}(\alpha^2). \qquad (16)$$

In particular, these results hold regardless of whether we use the linearized model $f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}$ or the standard one f.

Complexity. The space complexity of the algorithm is $\mathcal{O}(P + NS)$ and the time complexity is $\mathcal{O}(t_{\max}PN + NS^2)$.

5 Related work

Deep neural networks are known to suffer from poor calibration, where predictive uncertainties are often indistinguishable between in-distribution and out-of-distribution data [Guo et al., 2017]. *Deep ensembles* [Lakshminarayanan et al., 2017] remains one of the most widely used solutions as it has strong empirical performance. Unfortunately, this approach requires retraining multiple models from scratch, making it computationally prohibitive for large models.

Gaussian approximate posteriors are commonly used to avoid retraining from scratch and instead only explore the mode of a single model. Established approaches in this area include *Bayes by backprop* [Blundell et al., 2015]), *stochastic weight averaging (SWAG)* [Maddox et al., 2019], and *Laplace approximations* [Daxberger et al., 2021a]. Our work can be viewed as a variant of the Laplace approximation that avoids the underfitting commonly seen in Gaussian posteriors.

The computational costs of working with high-dimensional Gaussians can be daunting. Common workarounds include using low-rank or sparse covariances [Maddox et al., 2019, Ritter et al., 2018, Deng et al., 2022], subspace inference approaches [Izmailov et al., 2020], and subnetwork inference [Daxberger et al., 2021b]. These methods reduce computational costs while preserving strong predictive performance. In contrast, our method additionally provides a rigorous theoretical justification for the choice of subspace. We can infer properties of distributions within this subspace and derive formal error bounds, a level of theoretical grounding often lacking in other approaches.

Antoran et al. [2023] use a sample-then-optimize procedure to simulate the LLA posterior, which has some
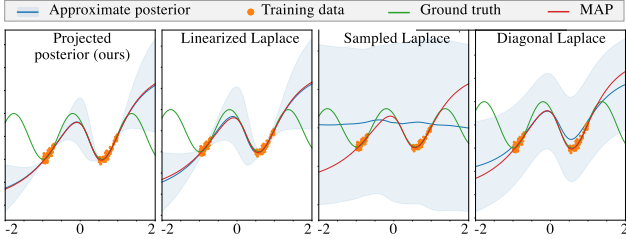
Figure 4: The sparse approximations of the posterior fail to capture in-between uncertainty whereas fully correlated posteriors can capture them.

algorithmic similarities to our work. We both simulate a simple distribution followed by a gradient-based iterative refinement. While Antoran et al. [2023] focus on LLA, we aim to avoid underfitting, which are potentially complementary objectives. Here we emphasize that our approach is general and can be applied to any Bayesian approximation in a post-hoc manner.

A frequentist perspective was given by Madras et al. [2020]. Their *local ensembles* perturb a MAP estimate in the kernel of the network's Hessian. Algorithmically, they approximate the largest Hessian eigenvectors using Lanczos's decomposition [Meurant, 2006] and assume that the orthogonal complement is the Hessian kernel. We improve this in several ways. First, we show that the GGN kernel is more appropriate than the Hessian kernel. Secondly, we propose a significantly more efficient algorithm that projects onto the GGN kernel by leveraging its partitioned structure. Our method guarantees exact convergence to the kernel, and it is not memory-limited as it avoids storing Lanczos vectors.

Alternating projections first appeared in von Neumann's notes on operator theory [1949], while Kayalar and Weinert [1988] proved the convergence rate. To our knowledge, the only recent machine learning use-case is inverting Gram matrices in Gaussian process regression [Wu et al., 2024]. This is rather different from our work.

6 Experiments

We benchmark our *projected* and *loss-projected* posteriors against popular post-hoc Bayesian methods, including SWAG, and diagonal and last-layer Laplace. We also include the prediction MAP. Baseline prior precisions are tuned via grid search, while projection posteriors use the optimal prior precision (9). Details on models and hyperparameters are in Appendix C.

6.1 Toy regression

> **Hypothesis:** The projected posterior captures in-between uncertainties better than sparsely correlated Laplace.

We first assess different Laplace approximations in a toy regression task. Foong et al. [2019] shows that Bayesian neural networks with mean-field variational inference fail to capture in-between uncertainties. We observe a similar issue with sparsely correlated Laplace approximations (Fig. 4). In contrast, the projected posterior correlates all parameters, improving predictive uncertainty estimates for unseen regions. *This illustrates that maintaining full parameter correlations is essential for accurate in-between uncertainties.*

6.2 Predictive uncertainties on standard image classification

> **Hypothesis:** The projected posterior provide competitive or superior predictive uncertainties across tasks, including in-distribution calibration, robustness to distribution shifts, and out-of-distribution detection.

In-distribution performance (Table 2). We next consider standard image classification tasks. We train a LeNet [LeCun et al., 1989] on MNIST and Fashion MNIST, and a ResNet [He et al., 2016] on CIFAR-10 [Krizhevsky et al., 2009] and SVHN [Netzer et al., 2011]. Keeping the MAP fixed we assess the predictive posterior of all methods using metrics such as confidence, accuracy, negative log-likelihood (NLL), Brier score [Brier, 1950], expected calibration error (ECE), and maximum calibration error (MCE) [Naeini et al., 2015].

As shown in Table 2, the projected posterior matches the MAP in-distribution, as expected from Lemma 3.1, with zero variance in-distribution and higher variance out-of-distribution. The loss-projected posterior improves calibration, which is discussed in Appendix B.

Distribution shift (Fig. 6). To assess robustness under distribution shifts, we evaluate the robustness of predictive uncertainties using ROTATED-MNIST, ROTATED-FMNIST, and CORRUPTED CIFAR (with fog and Gaussian blur). The loss-projected posterior maintains low, stable calibration error with increasing shift intensity, while the projected posterior retains high accuracy (Fig. 6).

Out-of-distribution detection (Table 3). For out-of-distribution (OOD) detection, we use the maximum variance of logits across output dimensions as an OOD score for all Bayesian posteriors, while the MAP baseline employs maximum softmax, which is a strong baseline. Lemma 3.1 dictates that the in-distribution variance of the projected posterior is zero, providing a strong OOD detection signal. Table 3 supports the theory, showing that the projected posterior achieves superior AUROC scores across various OOD tasks over baselines.

6.3 ImageNet and CelebA vision transformer

Marco Miani[†], Hrittik Roy[†], Søren Hauberg

Table 2: In-distribution performance across methods trained on MNIST, FMNIST SVHN and CIFAR-10.

		Conf. (↑)	NLL (↓)	Acc. (↑)	Brier (↓)	ECE (↓)	MCE (↓)
MNIST	MAP	0.981±0.002	0.080±0.005	**0.977±0.002**	1.775±0.004	0.787±0.009	0.895±0.014
	Projected posterior (ours)	0.981±0.002	0.080±0.005	**0.977±0.002**	1.774±0.004	0.787±0.009	0.895±0.014
	Loss-projected posterior (ours)	0.813±0.018	1.225±0.099	0.949±0.000	**1.532±0.028**	**0.666±0.007**	0.894±0.011
	SWAG	**0.982±0.001**	**0.064±0.006**	0.982±0.000	1.776±0.002	0.788±0.005	0.906±0.013
	Last-Layer	0.977±0.002	0.090±0.005	0.975±0.002	1.768±0.003	0.784±0.007	**0.887±0.008**
	Diagonal	0.951±0.008	0.129±0.016	0.975±0.001	1.727±0.012	0.784±0.008	0.894±0.015
FMNIST	MAP	**0.938±0.005**	0.325±0.011	0.897±0.002	1.713±0.008	0.732±0.006	0.901±0.006
	Projected posterior (ours)	0.937±0.005	**0.325±0.011**	0.897±0.002	1.713±0.008	0.732±0.006	0.902±0.006
	Loss-projected posterior (ours)	0.744±0.031	1.529±0.371	0.871±0.006	**1.441±0.041**	**0.617±0.025**	0.901±0.013
	SWAG	0.931±0.006	0.327±0.001	**0.898±0.001**	1.703±0.008	0.725±0.003	0.907±0.003
	Last Layer	0.931±0.005	0.339±0.011	0.896±0.002	1.703±0.008	0.727±0.004	0.902±0.004
	Diagonal	0.735±0.024	0.922±0.051	0.855±0.000	1.431±0.033	0.621±0.021	**0.895±0.021**
SVHN	MAP	**0.949±0.004**	0.188±0.004	**0.949±0.002**	1.691±0.004	0.740±0.012	0.889±0.004
	Projected posterior (ours)	**0.949±0.004**	0.188±0.004	**0.949±0.002**	1.691±0.004	0.740±0.012	0.889±0.007
	Loss-projected posterior (ours)	0.948±0.005	0.191±0.007	**0.949±0.003**	1.685±0.013	**0.734±0.017**	0.880±0.012
	SWAG	0.897±0.007	0.217±0.014	0.947±0.004	**1.606±0.010**	0.745±0.007	**0.874±0.003**
	Last-Layer	0.943±0.005	0.197±0.009	0.946±0.001	1.681±0.006	0.740±0.007	0.899±0.009
	Diagonal	0.948±0.003	**0.187±0.005**	**0.949±0.002**	1.689±0.003	0.742±0.011	0.899±0.000
CIFAR-10	MAP	**0.952±0.002**	0.422±0.011	**0.894±0.002**	1.698±0.038	0.703±0.013	0.878±0.009
	Projected posterior (ours)	**0.952±0.001**	0.422±0.011	**0.894±0.002**	1.698±0.038	0.703±0.013	0.878±0.002
	Loss-projected posterior (ours)	0.701±0.013	2.643±0.205	0.855±0.002	**1.387±0.018**	**0.559±0.006**	**0.802±0.005**
	SWAG	0.914±0.035	0.445±0.063	0.865±0.029	1.670±0.049	0.694±0.018	0.881±0.005
	Last-Layer	0.944±0.001	**0.406±0.005**	**0.894±0.001**	1.712±0.001	0.704±0.000	0.880±0.007
	Diagonal	0.934±0.028	0.465±0.069	0.872±0.032	1.698±0.038	0.703±0.013	0.873±0.005

Table 3: Out-of-distribution AUROC (↑) performance for MNIST, FMNIST, SVHN and CIFAR-10.

Trained on		MNIST			FMNIST			CIFAR-10		SVHN	
Tested on	FMNIST	EMNIST	KMNIST	MNIST	EMNIST	KMNIST	SVHN	CIFAR-100	CIFAR-10	CIFAR-100
MAP	0.924±0.015	0.888±0.010	0.846±0.019	0.714±0.001	0.788±0.016	0.664±0.012	0.874±0.002	0.816±0.026	0.960±0.007	0.954±0.008
Proj. (ours)	**0.926±0.013**	0.904±0.004	**0.862±0.011**	0.861±0.012	0.844±0.032	0.867±0.012	**0.881±0.002**	**0.836±0.004**	0.960±0.002	0.957±0.001
Loss-proj. (ours)	0.899±0.011	0.893±0.006	0.856±0.002	**0.914±0.035**	**0.928±0.007**	**0.907±0.026**	0.863±0.008	0.800±0.013	**0.966±0.009**	**0.960±0.006**
SWAG	0.917±0.024	**0.916±0.010**	0.861±0.032	0.685±0.014	0.751±0.032	0.655±0.015	0.798±0.040	0.782±0.042	0.777±0.029	0.787±0.027
Last-Layer	0.793±0.215	0.782±0.182	0.759±0.166	0.768±0.001	0.824±0.016	0.699±0.020	0.811±0.024	0.801±0.026	0.914±0.005	0.908±0.005
Diagonal	0.772±0.218	0.768±0.191	0.745±0.174	0.726±0.028	0.725±0.034	0.683±0.013	0.836±0.022	0.806±0.027	0.917±0.005	0.916±0.004

	Conf. (↑)	NLL (↓)	Acc. (↑)	Brier (↓)	ECE (↓)	MCE (↓)	AUROC (↑)
MAP	**0.626**	**1.358**	**0.726**	0.407	0.136	0.583	0.751
Loss-projected posterior	0.618	1.427	0.716	**0.394**	**0.107**	**0.259**	**0.756**

Table 4: In- and out-of-distribution metrics for SWIN transformer trained on ImageNet and tested on PLACES365.

Figure 5: **Variational autoencoder reconstructions and corresponding uncertainty estimates.** *Left:* mean reconstructions of MNIST and Fashion MNIST images sampled from the latent space. *Right:* pixel-wise uncertainty estimates generated by sampling decoder parameters from the Loss Kernel Posterior, highlighting key semantic features such as edges and contours. This demonstrates Projected Laplace's ability to capture uncertainty in high-dimensional generative models.

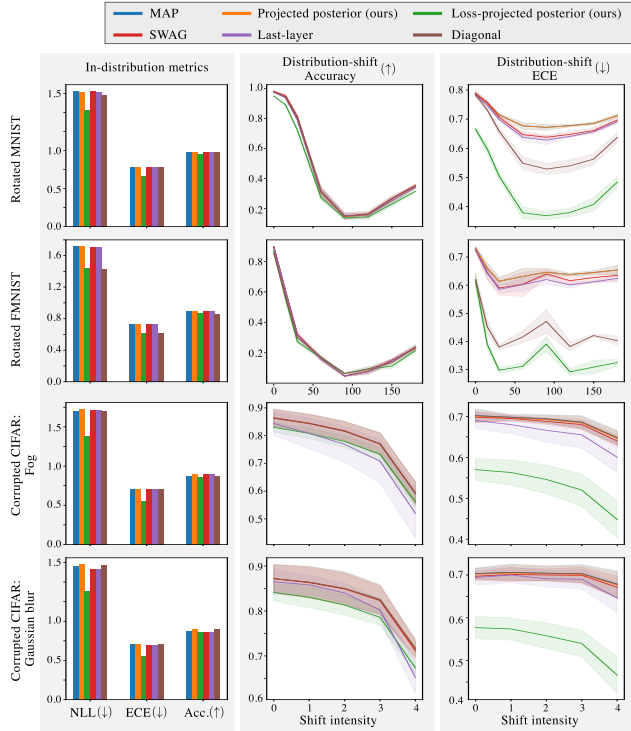Figure 6: Model calibration and fit on in-distribution test data (left) and under distribution shift (middle, right) where we plot shift intensities against accuracy and expected calibration error (ECE), respectively.

> **Hypothesis:** The projected posterior scales effectively to large models and datasets, such as SWIN transformers trained on ImageNet and CelebA.

We assess the scalability of our approach by sampling from the loss-projected posterior of a SWIN transformer [Liu et al., 2021] pre-trained on the ImageNet-1K dataset [Deng et al., 2009, Russakovsky et al., 2015], which includes about 1 million images of size $224 \times 224 \times 3$ with 1000 output classes. Table 4 shows marginal improvement in-distribution calibration over MAP alongside improved out-of-distribution detection on PLACES365 [Zhou et al., 2017]. *The main point is that our method scales gracefully to large models and datasets due to linear complexity.*

We further consider a 4M parameter Visual Attention network (VAN)[Guo et al., 2023]. We train on CelebA [Liu et al., 2015] holding out three classes. Table 5 reports AUROC scores to measure the ability to detect the three held-out classes and FOOD101 [Bossard et al., 2014] as out-of-distribution. Both our method and baselines have a limited memory budget of $300P$ numbers. LLA and local ensemble are implemented using Lanczos and we also include our projected score computed with Lanczos as a reference. The projected posterior outperforms the baselines on the three most challeng-

Table 5: AUROC (3 seeds) for a VAN trained on CelebA. Details are in the appendix.

	Food101	Bald only	Eyeglasses only	Mustache only
Max logit	0.959±0.008	0.32±0.00	0.57±0.01	0.40±0.03
Deep ensemble	0.957	0.71	0.74	0.61
Proj. (ours)	0.958±0.004	**0.81±0.03**	**0.75±0.01**	**0.66±0.01**
Loss-proj. (ours)	0.947±0.007	0.76±0.03	0.72±0.01	0.62±0.01
Proj-Lanczos	0.954±0.004	0.80±0.03	0.74±0.01	0.64±0.01
Local ensemble	0.951±0.003	0.79±0.04	0.73±0.01	0.63±0.01
LLA	0.954±0.00	0.80±0.03	0.74±0.01	0.64±0.01
LLA-diag	0.797±0.021	0.54±0.03	0.57±0.02	0.45±0.03
SCOD	**0.964±0.000**	0.79±0.03	0.73±0.01	0.64±0.01
SWAG	0.740±0.021	0.70±0.08	0.52±0.04	0.49±0.04
SLU	0.953±0.003	0.80±0.03	0.74±0.01	0.64±0.01

ing settings and is only second to SCOD in the last. The loss-projected posterior shows a minor drop in performance.

6.4   Generative models

> **Hypothesis:** The loss-projected posterior scales to generative models with large output dimensions.

We evaluate the loss-projected posterior on variational autoencoders [Kingma, 2013] trained on MNIST and FASHION MNIST, showcasing its flexibility with high-dimensional outputs and diverse tasks. After training, we sample decoder parameters to generate images from latent space samples. Fig. 5 shows that the sampled decoders produce diverse reconstructions with pixel-wise uncertainty estimates that capture meaningful features.

This experiment highlights that our approach scales to generative models while preserving essential predictive uncertainties. Its adaptability beyond classification makes it a robust tool for diverse tasks.

7   Summary and limitations

Theoretically we show that our proposed *projected posterior* is optimal w.r.t. underfitting for noiseless data. We further show that existing Laplace approximations do not meet this fundamental requirement. However, the applicability to high-noise data remains unresolved.

Algorithmically we introduce a memory-efficient algorithm for computing projection-vector-products for a Jacobian kernel. This approach is based on the novel idea of representing the kernel as an intersection of kernels, which enables the use of alternating projections.

Empirically our methods outperform existing baselines in out-of-distribution detection and in-distribution calibration in diverse settings, ranging from toy regression problems to vision transformers on ImageNet.

Extending the method can be done in several ways. We merely project an isotropic Gaussian to isolate the strength of the projection itself, but we stress

that the algorithm lets us project samples from any distribution. For example, it is straightforward to project samples from last-layer Laplace into the kernel. We emphasize that our method is very general as it works for any differentiable model, unlike e.g. KFAC [Dangel et al., 2020] which requires network-specific implementations. Our open-source implementation is applicable whenever we can access Jacobian-vector and vector-Jacobian products.

Acknowledgements

References

Z. Allen-Zhu, Y. Li, and Y. Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32, 2019.

J. Antorán, S. Padhy, R. Barbano, E. Nalisnick, D. Janz, and J. M. Hernández-Lobato. Sampling-based inference for large linear models, with application to linearised laplace. *arXiv preprint arXiv:2210.04994*, 2022.

J. Antoran, S. Padhy, R. Barbano, E. Nalisnick, D. Janz, and J. M. Hernández-Lobato. Sampling-based inference for large linear models, with application to linearised laplace. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=aoDyX6vSqsd.

C. M. Bishop. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

S. Bombari, M. H. Amani, and M. Mondelli. Memorization and optimization in deep neural networks with minimum over-parameterization. *Advances in Neural Information Processing Systems*, 35:7628–7640, 2022.

L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pages 446–461. Springer, 2014.

A. Botev, H. Ritter, and D. Barber. Practical gauss-newton optimisation for deep learning. In *International Conference on Machine Learning*, pages 557–565. PMLR, 2017.

L. M. Bregman. Finding the common point of convex sets by the method of successive projection. In *Doklady Akademii Nauk*, volume 162, pages 487–490. Russian Academy of Sciences, 1965.

G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1 – 3, 1950. doi: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2. URL https://journals.ametsoc.org/view/journals/mwre/78/1/1520-0493_1950_078_0001_vofeit_2_0_co_2.xml.

F. Dangel, F. Kunstner, and P. Hennig. BackPACK: Packing more into backprop. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJlrF24twB.

DarshanDeshpande. Jax models. https://github.com/DarshanDeshpande/jax-models, 2021.

E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103, 2021a.

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

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Z. Deng, F. Zhou, and J. Zhu. Accelerated linearized laplace approximation for bayesian deep learning. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=jftNpltMgz.

R. Dhahri, A. Immer, B. Charpentier, S. Günnemann, and V. Fortuin. Shaving weights with occam's razor: Bayesian sparsification for neural networks using the marginal likelihood. *arXiv preprint arXiv:2402.15978*, 2024.

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

T. George, C. Laurent, X. Bouthillier, N. Ballas, and P. Vincent. Fast approximate natural gradient descent in a kronecker factored eigenbasis. *Advances in Neural Information Processing Systems*, 31, 2018.

P. Germain, F. Bach, A. Lacoste, and S. Lacoste-Julien. Pac-bayesian theory meets bayesian inference. *Advances in Neural Information Processing Systems*, 29, 2016.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu. Visual attention network. *Computational Visual Media*, 9(4):733–752, 2023.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

A. Immer, M. Bauer, V. Fortuin, G. Rätsch, and K. M. Emtiyaz. Scalable marginal likelihood estimation for model selection in deep learning. In *International Conference on Machine Learning*, pages 4563–4573. PMLR, 2021a.

A. Immer, M. Korzepa, and M. Bauer. Improving predictions of Bayesian neural nets via local linearization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 703–711, 2021b.

P. Izmailov, W. J. Maddox, P. Kirichenko, T. Garipov, D. Vetrov, and A. G. Wilson. Subspace inference for bayesian deep learning. In *Uncertainty in Artificial Intelligence*, pages 1169–1179. PMLR, 2020.

K. Karhadkar, M. Murray, and G. Montúfar. Bounds for the smallest eigenvalue of the ntk for arbitrary spherical data of arbitrary dimension. *arXiv preprint arXiv:2405.14630*, 2024.

S. Kayalar and H. L. Weinert. Error bounds for the method of alternating projections. *Mathematics of Control, Signals and Systems*, 1(1):43–59, 1988.

M. E. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *International conference on machine learning*, pages 2611–2620. PMLR, 2018.

M. E. Khan, A. Immer, E. Abedi, and M. Korzepa. Approximate inference turns deep networks into Gaus-

sian processes. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

D. P. Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

A. Kristiadi, M. Hein, and P. Hennig. Being a bit frequentist improves bayesian neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 529–545. PMLR, 2022.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

D. J. MacKay. Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks. *Network: computation in neural systems*, 6(3):469, 1995.

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

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

D. Madras, J. Atwood, and A. D'Amour. Detecting extrapolation with local ensembles. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJl6bANtwH.

J. Martens and R. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

G. Meurant. *The Lanczos and conjugate gradient algorithms: from theory to finite precision computations*. SIAM, 2006.

M. Miani, L. Beretta, and S. Hauberg. Sketched lanczos uncertainty score: a low-memory summary of the

fisher information. In *Neural Information Processing Systems (NeurIPS)*, 2024.

M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, 2011.

Q. Nguyen, M. Mondelli, and G. F. Montufar. Tight bounds on the smallest eigenvalue of the neural tangent kernel for deep relu networks. In *International Conference on Machine Learning*, pages 8119–8129. PMLR, 2021.

K. Osawa, S. Swaroop, M. E. Khan, A. Jain, R. Eschenhagen, R. E. Turner, and R. Yokota. Practical deep learning with bayesian principles. *Advances in neural information processing systems*, 32, 2019.

V. Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 21(252):1–64, 2020.

H. Ritter, A. Botev, and D. Barber. A scalable laplace approximation for neural networks. In *6th international conference on learning representations, ICLR 2018-conference track proceedings*, volume 6. International Conference on Representation Learning, 2018.

H. Roy, M. Miani, C. H. Ek, P. Hennig, M. Pförtner, L. Tatzel, and S. Hauberg. Reparameterization invariance in approximate bayesian inference, 2024. URL https://arxiv.org/abs/2406.03334.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

A. Sharma, N. Azizan, and M. Pavone. Sketching curvature for efficient out-of-distribution detection for deep neural networks. In *Uncertainty in artificial intelligence*, pages 1958–1967. PMLR, 2021.

K. T. Smith, D. C. Solmon, and S. L. Wagner. Practical and mathematical aspects of the problem of reconstructing objects from radiographs. 1977.

M. Stephan, M. D. Hoffman, D. M. Blei, et al. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18 (134):1–35, 2017.

J. von Neumann. On rings of operators. reduction theory. *Annals of Mathematics*, 50:401, 1949. URL https://api.semanticscholar.org/CorpusID:124439084.

F. Wenzel, K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

K. Wu, J. Wenger, H. T. Jones, G. Pleiss, and J. Gardner. Large-scale gaussian processes via alternating projection. In *International Conference on Artificial Intelligence and Statistics*, pages 2620–2628. PMLR, 2024.

Y. Zhang, Y.-S. Wu, L. A. Ortega, and A. R. Masegosa. The cold posterior effect indicates underfitting, and cold posteriors represent a fully bayesian method to mitigate it. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=GZORXGxHHT.

B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

A   Proofs

A.1   Proof of positive variance lemma

In this section, we prove Lemma 3.2. We restate it for convenience.

Let $\mathbf{J}_{\boldsymbol{\theta}} = [\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_1)^\top \ldots \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}_N)^\top]^\top$ and $\mathbf{x}_{\text{test}} \in \mathbb{R}^I$, then

$$\text{VAR}_{\boldsymbol{\theta} \sim q_{\text{PROJ}}} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_{\text{test}}) > 0$$

$$\text{RANK}\left(\begin{pmatrix} \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}} \\ \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_{\text{test}}) \end{pmatrix} \begin{pmatrix} \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}} \\ \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_{\text{test}}) \end{pmatrix}^\top \right) > \text{RANK}\left(\mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}} \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}^\top \right).$$

Proof. It is easier to show the contrapositive i.e. to show that if the variance is 0, then the square matrix has rank at most NO.

Let's first define a more compact notation $\mathbf{J}_D = \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}$ and $\mathbf{J}_T = \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_{test})$ for the two Jacobians. Let $D = \text{RANK}(\mathbf{J}_T)$ and $T = \text{RANK}(\mathbf{J}_T)$. It holds that $D \leq NO$ and $T \leq O$ since the rank is always upper bound by the number of rows. Consider their singular values decompositions

$$J_D = \sum_{i=1}^{D} \sigma_i v_i V_i^\top \qquad J_T = \sum_{j=1}^{T} \gamma_j w_j W_j^\top, \tag{17}$$

where $\sigma_i, \gamma_j > 0$, $v_i \in \mathbb{R}^D$, $V_i \in \mathbb{R}^P$, $w_j \in \mathbb{R}^T$, $W_j \in \mathbb{R}^P$, $v_i^\top v_{i'} = V_i^\top V_{i'} = \delta_{ii'}$ and $w_j^\top W_{j'} = W_j^\top W_{j'} = \delta_{jj'}$ for any $i, i' = 1, \ldots, D$ and $j, j' = 1, \ldots, T$. Moreover let $V_{D+1}, \ldots, V_P \in \mathbb{R}^P$ be a orthonormal completion of $V_1, \ldots, V_D$ as a basis.

$$\text{VAR}_{\boldsymbol{\theta} \sim q} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_{test}) = \text{TR}\left(\sum_{j,j'=1}^{T} \gamma_j w_j W_j^\top \overbrace{\left(\sum_{i=D+1}^{P} V_i V_i^\top \right)}^{\mathbb{I} - \mathcal{P}(\text{GGN})} \gamma_{j'} W_{j'} w_{j'}^\top \right) \tag{18}$$

$$= \sum_{i=D+1}^{P} \sum_{j=1}^{T} \gamma_j^2 W_j^\top V_i V_i^\top W_j \tag{19}$$

$$= \sum_{i=D+1}^{P} \sum_{j,=1}^{T} \gamma_j^2 (W_j^\top V_i)^2 \tag{20}$$

and a sum of positive elements being 0 implies that every element is 0, thus

$$\text{VAR}_{\boldsymbol{\theta} \sim q} f_{\text{lin}}^{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_{test}) = 0 \implies W_j \perp V_i \quad \forall j = 1, \ldots T \quad \forall i = D+1, \ldots, P \tag{21}$$

and since $V_1, \ldots, V_P$ is a basis, this implies that there exists some coefficients $\beta_k^{(j)}$ such that $W_j = \sum_{k=1}^{D} \beta_k^{(j)} V_k$. Consequently

$$W_j W_j^\top = \sum_{k,k'=1}^{D} \beta_k^{(j)} V_k \beta_{k'}^{(j)} V_{k'}^\top = \sum_{k=1}^{D} (\beta_k^{(j)})^2 V_k V_k^\top \quad \forall j = 1, \ldots O \tag{22}$$

Now consider the matrix

$$\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix}^\top\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix} = \mathbf{J}_D^\top\mathbf{J}_D + \mathbf{J}_T^\top\mathbf{J}_T \tag{23}$$

$$= \sum_{i=1}^{D}\sigma_i V_i V_i^\top + \sum_{j=1}^{T}\gamma_j W_j W_j^\top \qquad (Eq.\ 22) \tag{24}$$

$$= \sum_{i=1}^{D}\sigma_i V_i V_i^\top + \sum_{j=1}^{T}\gamma_j\sum_{k=1}^{D}(\beta_k^{(j)})^2 V_k V_k^\top \tag{25}$$

$$= \sum_{i=1}^{D}\sigma_i V_i V_i^\top + \sum_{k=1}^{D}\left(\sum_{j=1}^{T}\gamma_j(\beta_k^{(j)})^2\right) V_k V_k^\top \tag{26}$$

$$= \sum_{i=1}^{D}\left(\sigma_i + \sum_{j=1}^{T}\gamma_j(\beta_i^{(j)})^2\right) V_i V_i^\top \tag{27}$$

which has rank equal to D. Then finally

$$\mathrm{RANK}\left(\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix}\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix}^\top\right) = \mathrm{RANK}\left(\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix}^\top\begin{pmatrix}\mathbf{J}_D\\\mathbf{J}_T\end{pmatrix}\right) = D \le NO \tag{28}$$

which concludes the proof. $\qquad\qquad\square$

A.2   Proof of the upper bound on Linearized Laplace predictive variance

In this section, we prove Theorem 3.3. We restate it below for convenience

Theorem 3.3. *For $\alpha > 0$, the predictive variance of LLA on any training datapoint is positive and bounded,*

$$\frac{O\gamma^2}{\gamma^2 + \alpha} \le \mathrm{VAR}_{\theta\sim q_{\mathrm{LLA}}}f_{\mathrm{lin}}^{\boldsymbol{\theta}_{\mathrm{MAP}}}(\boldsymbol{\theta},\mathbf{x}) \le \frac{O\lambda^2}{\lambda^2 + \alpha} \quad \text{for } \mathbf{x}\in\mathcal{D}.$$

Here, λ and γ are the largest and smallest singular values of the dataset Jacobian $\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}$, respectively.

Proof. Let $\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}} = \sum_{i=1}^{NO}\sigma_i w_i^\top v_i$ be a SVD decomposition of the full dataset Jacobian. Namely, $w_i \in \mathbb{R}^{NO}$, $v_i \in \mathbb{R}^P$ for any i, and $w_i^\top w_j = \delta_{ij}$, $v_i^\top v_j = \delta_{ij}$ for any i, j. Then it holds

$$(\mathrm{GGN} + \alpha\mathbb{I})^{-1} = \alpha^{-1}\mathbb{I} + \sum_{i=1}^{NO}\left(\frac{1}{\sigma_i^2 + \alpha} - \frac{1}{\alpha}\right) v_i^\top v_i \tag{29}$$

and consequently

$$\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathrm{GGN} + \alpha\mathbb{I})^{-1}\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}^\top = \sum_{j,k=1}^{NO}\sigma_j w_j^\top v_j\left(\alpha^{-1}\mathbb{I} + \sum_{i=1}^{NO}\left(\frac{1}{\sigma_i^2 + \alpha} - \frac{1}{\alpha}\right) v_i^\top v_i\right)\sigma_k v_k^\top w_k \tag{30}$$

$$= \sum_{i=1}^{NO}\frac{\sigma_i^2}{\alpha}w_i^\top w_i + \sum_{i=1}^{NO}\sigma_i^2\left(\frac{1}{\sigma_i^2 + \alpha} - \frac{1}{\alpha}\right)w_i^\top w_i \tag{31}$$

$$= \sum_{i=1}^{NO}\frac{\sigma_i^2}{\sigma_i^2 + \alpha}w_i^\top w_i \tag{32}$$

Now note that each datapoint Jacobian can be written in terms of the full dataset Jacobian as $\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathbf{x}_n) = \sum_{i=1}^{O}f_i^\top e_{(n-1)O+i}\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}$ where $\{f_i\}_{i=1\ldots O}$ and $\{e_i\}_{i=1\ldots NO}$ are the canonical bases of $\mathbb{R}^O$ and $\mathbb{R}^{NO}$, respectively.

We can finally express the Linearized Laplace predictive variance, for any train datapoint $\mathbf{x}_n$, as

$$\mathrm{VAR}_{\theta \sim \mathcal{N}(\boldsymbol{\theta}_{\mathrm{MAP}}, (\mathrm{GGN}+\alpha\mathbb{I})^{-1})} f_{\mathrm{lin}}^{\boldsymbol{\theta}_{\mathrm{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_n) = \mathrm{TR}\left(\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathbf{x}_n)(\mathrm{GGN}+\alpha\mathbb{I})^{-1}\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathbf{x}_n)^{\top}\right) \tag{33}$$

$$= \mathrm{TR}\left(\sum_{i,j=1}^{O} f_i^{\top} e_{(n-1)O+i} \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathbf{x}_n)(\mathrm{GGN}+\alpha\mathbb{I})^{-1}\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}^{\top} e_{(n-1)O+j}^{\top} f_j\right)$$

$$= \sum_{i=1}^{O} e_{(n-1)O+i} \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}(\mathrm{GGN}+\alpha\mathbb{I})^{-1}\mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}}^{\top} e_{(n-1)O+i}^{\top}$$

$$= \sum_{i=1}^{O}\sum_{j=1}^{NO} \frac{\sigma_j^2}{\sigma_j^2 + \alpha} e_{(n-1)O+i} w_j^{\top} w_j e_{(n-1)O+i}^{\top}$$

Now the *upper bound* follows by noting that for all j

$$\frac{\sigma_j^2}{\sigma_j^2 + \alpha} \leq \max_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha} \tag{34}$$

and thus from Eq. 33, noting that $\sum_{j=1}^{NO} w_j^{\top} w_j = \mathbb{I}_{NO}$ we have for any train datapoint $\mathbf{x}_n$

$$\mathrm{VAR}_{\theta \sim \mathcal{N}(\boldsymbol{\theta}_{\mathrm{MAP}}, (\mathrm{GGN}+\alpha\mathbb{I})^{-1})} f_{\mathrm{lin}}^{\boldsymbol{\theta}_{\mathrm{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_n) = \sum_{i=1}^{O}\sum_{j=1}^{NO} \frac{\sigma_j^2}{\sigma_j^2 + \alpha} e_{(n-1)O+i} w_j^{\top} w_j e_{(n-1)O+i}^{\top} \tag{35}$$

$$\leq \max_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha} \sum_{i=1}^{O}\sum_{j=1}^{NO} e_{(n-1)O+i} w_j^{\top} w_j e_{(n-1)O+i}^{\top}$$

$$= \max_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha} \sum_{i=1}^{O} e_{(n-1)O+i} e_{(n-1)O+i}^{\top} = O \max_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha}$$

While the *lower bound*, similarly, follows by noting that for all j

$$\frac{\sigma_j^2}{\sigma_j^2 + \alpha} \geq \min_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha} \tag{36}$$

and thus from Eq. 33 we have for any train datapoint $\mathbf{x}_n$

$$\mathrm{VAR}_{\theta \sim \mathcal{N}(\boldsymbol{\theta}_{\mathrm{MAP}}, (\mathrm{GGN}+\alpha\mathbb{I})^{-1})} f_{\mathrm{lin}}^{\boldsymbol{\theta}_{\mathrm{MAP}}}(\boldsymbol{\theta}, \mathbf{x}_n) = \sum_{i=1}^{O}\sum_{j=1}^{NO} \frac{\sigma_j^2}{\sigma_j^2 + \alpha} e_{(n-1)O+i} w_j^{\top} w_j e_{(n-1)O+i}^{\top} \tag{37}$$

$$\geq \min_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha} \sum_{i=1}^{O}\sum_{j=1}^{NO} e_{(n-1)O+i} w_j^{\top} w_j e_{(n-1)O+i}^{\top} = O \min_k \frac{\sigma_k^2}{\sigma_k^2 + \alpha}$$

which concludes the proof by noting that the function $\sigma \mapsto \frac{\sigma}{\sigma+\alpha}$ is monotonic for any $\alpha > 0$, and that the eigenvalues of the GGN are a simple function of the singular values of the Jacobian: $\lambda_{max}(\mathrm{GGN}) = \max_k \sigma_k^2$ and $\lambda_{min}^{\neq 0}(\mathrm{GGN}) = \min_k \sigma_k^2$. We emphasize that the singular values of the Jacobian affect the non-zero eigenvalues of the GGN, thus the minimum singular value corresponds to the minimum-non-zero eigenvalue. $\square$

A.3 Proof of Lemma 3.4

Proof. It follows from equation 8 the approximate log marginal likelihood is given by:

$$\log q_{\mathrm{LLA}}(\mathcal{D}|\alpha) = \log p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}}) + \log p(\boldsymbol{\theta}_{\mathrm{MAP}}|\alpha) + \frac{1}{2}\log\det\left(\frac{1}{2\pi}(\mathrm{GGN}+\alpha\mathbb{I})^{-1}\right) \tag{38}$$

We can approximate $(\text{GGN} + \alpha\mathbb{I})^{-1}$ with $\alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))$(cf. lemma 3.5). Hence log-determinant of $(\text{GGN} + \alpha\mathbb{I})^{-1}$ can also be approximated by sum of log-eigenvalues $\alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))$.

$$\log q_{\text{LLA}}(\mathcal{D}|\alpha) = \log p(\mathcal{D}|\boldsymbol{\theta}_{\text{MAP}}) + \log p(\boldsymbol{\theta}_{\text{MAP}}|\alpha) + \frac{1}{2}\log\det\left(\frac{1}{2\pi}\left(\text{GGN} + \alpha\mathbb{I}\right)^{-1}\right) \tag{39}$$

$$\approx -\frac{1}{2}\alpha\|\boldsymbol{\theta}_{\text{MAP}}\|^2 + \frac{P - Tr(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))}{2}\log(\alpha) + C \tag{40}$$

where C denotes all the terms that don't depend on α. Taking the derivative wrt α of the above equation and setting it to zero gives us the stationary points.

$$\frac{d\log q_{\text{LLA}}(\mathcal{D}|\alpha)}{d\alpha} \approx -\frac{1}{2}\|\boldsymbol{\theta}_{\text{MAP}}\|^2 + \frac{P - Tr(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))}{2}\frac{1}{\alpha} = 0 \tag{41}$$

$$\implies \alpha^* = \frac{\|\boldsymbol{\theta}_{\text{MAP}}\|^2}{P - \text{Tr}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))} \tag{42}$$

To conclude the proof one only needs to notice that the second derivative of $\log q_{\text{LLA}}(\mathcal{D}|\alpha)$ is given by $-\frac{P-Tr(\mathbb{I}_P-\mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))}{2}\frac{1}{\alpha^2}$ which is always negative. Hence the stationary point is the maximum value of the approximate log marginal likelihood. $\qquad\square$

A.4 Proof of Error Bounds in Lemma 3.5

Proof. To prove the bound on the matrix norm of the difference between the covariances of Laplace's approximation and projection posterior, notice that

$$(\alpha\mathbb{I}_P + \text{GGN})^{-1} = \alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}) + V(\Lambda + \alpha)^{-1}V^T$$

Where Λ and V correspond to non-zero eigenvalues and eigenvectors of GGN respectively. Therefore from the properties of the spectral norm, we have that:

$$\left\|(\alpha\mathbb{I}_P + \text{GGN})^{-1} - \alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}})\right\| = \left\|V(\Lambda + \alpha)^{-1}V^T\right\|$$

$$\leq \frac{1}{\tau + \alpha}$$

This proves the first bound. To prove the bound on Wasserstein distance note that the Wasserstein distance between two Gaussian, $\mathcal{N}(\mu_1, \Sigma_1)$ and $\mathcal{N}(\mu_2, \Sigma_2)$, is given by:

$$d^2 = \left\|\mu_1 - \mu_2\right\|^2 + Tr(\Sigma_1 + \Sigma_2 - 2(\Sigma_2^{\frac{1}{2}}\Sigma_1\Sigma_2^{\frac{1}{2}})^{\frac{1}{2}})$$

We plug in $\mu_1 = \mu_1 = \boldsymbol{\theta}_{\text{MAP}}$, $\Sigma_1 = (\alpha\mathbb{I}_P + \text{GGN})^{-1}$ and $\Sigma_2 = \alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))$. Also notice that by the properties of projection matrices, it follows that $\Sigma_2^{\frac{1}{2}}\Sigma_1\Sigma_2^{\frac{1}{2}} = \frac{1}{\alpha^2}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))$. Hence we have that

$$d^2 = Tr\left((\alpha\mathbb{I}_P + \text{GGN})^{-1} - \alpha^{-1}(\mathbb{I}_P - \mathcal{P}(\text{GGN}_{\boldsymbol{\theta}_{\text{MAP}}}))\right)$$
$$= Tr(V(\Lambda + \alpha)^{-1}V^T)$$
$$\leq \frac{k}{\tau + \alpha}$$

$\qquad\square$

A.5 Proof of equation 11

Proof. To prove that $\ker\left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right) = \bigcap_b \ker(\mathbf{M}_b^\top \mathbf{M}_b)$, assume that $v \in \ker\left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right)$ then it follows that:

$$\left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right) v = 0$$

$$\implies v^T \left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right) v = 0$$

$$\implies \sum_b \underbrace{v^T \mathbf{M}_b^\top \mathbf{M}_b v}_{\geq 0} = 0$$

$$\implies v^T \mathbf{M}_b^\top \mathbf{M}_b v = 0 \quad \forall b$$

Hence, we can conclude that $v \in \bigcap_b \ker(\mathbf{M}_b^\top \mathbf{M}_b)$. On the other hand, it is obvious that if $v \in \bigcap_b \ker(\mathbf{M}_b^\top \mathbf{M}_b)$ we have that

$$\mathbf{M}_b^\top \mathbf{M}_b v = 0 \quad \forall b$$

$$\implies \sum_b \mathbf{M}_b^\top \mathbf{M}_b v = 0$$

$$\implies v \in \ker\left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right)$$

This proves that $\ker\left(\sum_b \mathbf{M}_b^\top \mathbf{M}_b\right) = \bigcap_b \ker(\mathbf{M}_b^\top \mathbf{M}_b)$. $\qquad\square$

A.6 Discussion of Lemma 4.1

The Convergence of Alternating Projections for two subspaces first appeared in John Von Neuman's Lecture Notes on Operator Theory[von Neumann, 1949]. It was extended to the case of infinite convex sets in Bregman [1965]. Rate of convergence was analyzed in several works such as Kayalar and Weinert [1988], Smith et al. [1977]. This gives us some understanding of the approximation error incurred by truncating the interactive algorithm after t steps.

A.7 Proof of Lemma 4.2

Proof. Suppose $v \in \ker(\mathbf{J}_{\boldsymbol{\theta}})$. Then we have that $\mathbf{J}_{\boldsymbol{\theta}} v = 0$. Note that $\mathbf{J}_{\boldsymbol{\theta}}^L = \nabla_{f(\boldsymbol{\theta},\mathbf{x})} l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) \mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x})$. Hence, we have that $\mathbf{J}_{\boldsymbol{\theta}}^L v = \nabla_{f(\boldsymbol{\theta},\mathbf{x})} l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) (\mathbf{J}_{\boldsymbol{\theta}}(\mathbf{x}) v) = \nabla_{f(\boldsymbol{\theta},\mathbf{x})} l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) \mathbf{0} = \mathbf{0}$. Thus we have that $v \in \ker(\mathbf{J}_{\boldsymbol{\theta}}^L)$. Therefore, $\ker(\mathbf{J}_{\boldsymbol{\theta}}) \subseteq \ker(\mathbf{J}_{\boldsymbol{\theta}}^L)$. $\qquad\square$

A.8 Proof of Lemma 4.3

In this section, we prove Lemma 4.3. We restate it for convenience.

Lemma 4.3. *For any $\theta \sim q_{\text{LOSS}}$ and $\mathbf{x}_n \in \mathcal{D}$ it holds*

$$|l(f(\boldsymbol{\theta},\mathbf{x}_n),\mathbf{y}_n) - l(f(\boldsymbol{\theta}_{\text{MAP}},\mathbf{x}_n),\mathbf{y}_n)| = \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}\|^2)$$

which implies that

$$\text{VAR}_{\boldsymbol{\theta} \sim q_{\text{LOSS}}} l(f(\boldsymbol{\theta},\mathbf{x}_n),\mathbf{y}_n) \leq \mathcal{O}(\alpha^2). \tag{16}$$

Proof. Consider the first order Taylor expansion of f around $\boldsymbol{\theta}_{\text{MAP}}$

$$f(\boldsymbol{\theta},\mathbf{x}) = f(\boldsymbol{\theta}_{\text{MAP}},\mathbf{x}) + \mathbf{J}_{\boldsymbol{\theta}_{\text{MAP}}}(\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}) + \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}}\|^2) \tag{43}$$

and the first order Taylor expansion of l around $f(\boldsymbol{\theta}_{\text{MAP}},\mathbf{x})$, which we shorten as f_{MAP}

$$l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) = l(f_{\text{MAP}},\mathbf{y}) + \nabla_{f_{\text{MAP}}} l(f_{\text{MAP}},\mathbf{y})(l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) - l(f_{\text{MAP}},\mathbf{y})) + \mathcal{O}(\|l(f(\boldsymbol{\theta},\mathbf{x}),\mathbf{y}) - l(f_{\text{MAP}},\mathbf{y})\|^2) \tag{44}$$

And we can use these to write

$$l(f(\boldsymbol{\theta}, \mathbf{x}), \mathbf{y}) - l(f_{\mathrm{MAP}}, \mathbf{y}) = \nabla_{f_{\mathrm{MAP}}} l(f_{\mathrm{MAP}}, \mathbf{y}) \mathbf{J}_{\boldsymbol{\theta}_{\mathrm{MAP}}} (\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}) + \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}\|^2) \tag{45}$$

$$= \nabla_{\boldsymbol{\theta}_{\mathrm{MAP}}} l(f(\boldsymbol{\theta}_{\mathrm{MAP}}, \mathbf{x}), \mathbf{y})(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}) + \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}\|^2) \tag{46}$$

where we collected all the second order terms in $\boldsymbol{\theta}$ in the $\mathcal{O}$ term.

Then, for any $(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{D}$ and for any $\boldsymbol{\theta} \sim q_{\mathrm{LOSS}}$, by definition of the loss kernel in Eq. 15, it holds that

$$\nabla_{\boldsymbol{\theta}_{\mathrm{MAP}}} l(f(\boldsymbol{\theta}_{\mathrm{MAP}}, \mathbf{x}_n), \mathbf{y}_n)(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}) = 0 \tag{47}$$

which we plug back into Eq. 45 and we get

$$l(f(\boldsymbol{\theta}, \mathbf{x}_n), \mathbf{y}_n) - l(f(\boldsymbol{\theta}_{\mathrm{MAP}}, \mathbf{x}_n), \mathbf{y}_n) = \mathcal{O}(\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}\|^2) \qquad \text{for any } \boldsymbol{\theta} \sim q_{\mathrm{LOSS}} \tag{48}$$

and the first part of the Lemma is proved. The variance bound directly follows since the norm $\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{MAP}}\|$ is controlled by the precision scale α of q_{LOSS}. $\qquad\square$

B   Calibration of loss kernel projection

To demonstrate how the poss-projected posterior improves calibration, consider a simple example of a classification task with C output classes. Suppose the maximum a posteriori (MAP) estimate is overly confident in its predictions, consistently assigning a probability of 1.0 to the predicted class and 0.0 to the remaining $C - 1$ classes. For simplicity, assume this MAP estimate correctly classifies 80% of the examples while misclassifying the remaining 20%.

In this scenario, the MAP estimate's confidence—represented by the maximum output probability is uniformly 1.0 for both correctly classified and misclassified examples. This results in overconfident predictions for the 20% of cases where the model is wrong.
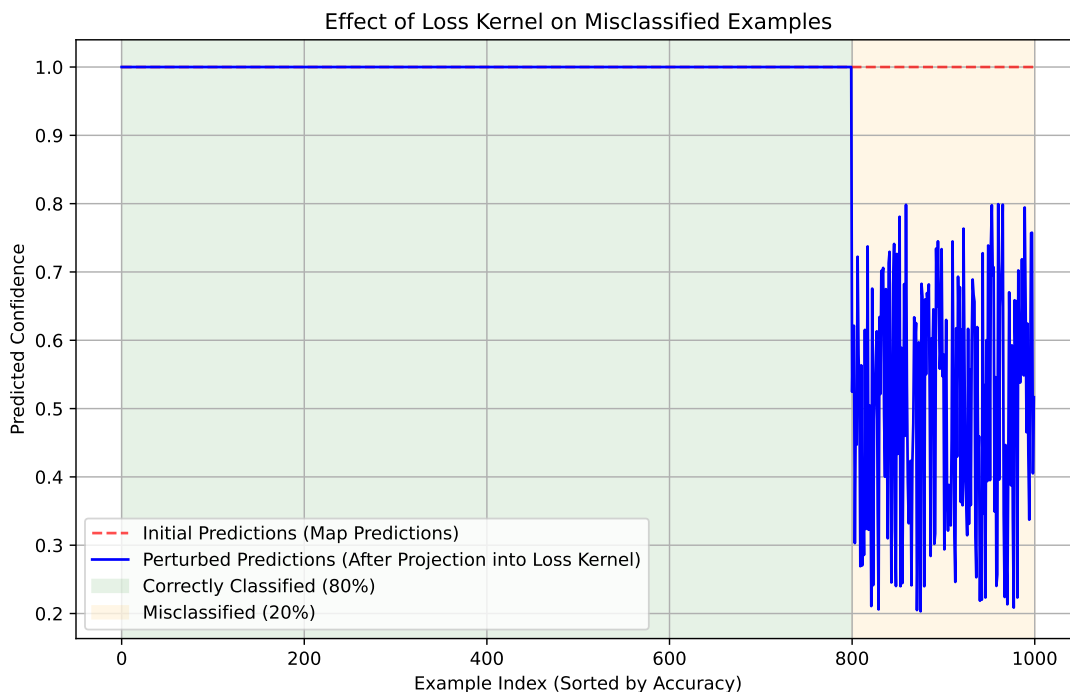

Figure 7: The loss-projected posterior perturb the predicted probabilities of all the classes except the true label. This leads to a lower confidence in misclassified examples, leading to better calibration.

Now, consider using the loss-projected posterior for predictions. This posterior is designed to preserve the model's accuracy by maintaining the probability assigned to the true label, especially for correctly classified examples. Thus, for the 80% of cases where the MAP is correct, both accuracy and confidence remain largely unchanged.

However, for the 20% of misclassified examples, the loss-projected posterior adjusts the predicted probabilities. Specifically, it reduces overconfidence by perturbing the probabilities of all classes except the true label. By redistributing some of the confidence away from the incorrect class, the model achieves better calibration. Misclassified examples no longer exhibit extreme certainty, thus reducing the calibration error and yielding predictions that are more aligned with the model's actual performance.

This behavior is illustrated in Figure 7, which shows how confidence is moderated on misclassified examples, leading to an overall improvement in the calibration of the model.

C   Implementation details and experimental setup

In this section, we outline the specifics of the experimental setup and provide key hyperparameter details. Any additional information about the experimental setup can be found in the submitted source code.

C.1   Motivation for the choice of baselines

We benchmark our method against several baselines, including MAP, Diagonal Laplace, Last-Layer Laplace, and SWAG. The motivation behind these choices is to compare our approach with other post-hoc methods that approximate a Gaussian posterior centered at the same mode.

We chose MAP as a baseline because the maximum softmax probability is a strong baseline for out-of-distribution (OOD) detection.

The last-layer Laplace approximation was selected as a baseline following recommendations from [Daxberger et al., 2021a]. This method is considered a strong representative of Laplace approximations and is expected to provide near-optimal performance across various configurations of Laplace approximations.

We include SWAG as another baseline, which provides a simple yet effective method for uncertainty quantification. SWAG builds a Gaussian approximation close to the MAP but with a low-rank covariance matrix which is different from Laplace approximations.

Finally, diagonal Laplace is included to assess the impact of posterior correlations. This method simplifies the full Laplace by using only diagonal covariance, providing insight into the difference between sparsely correlated and fully correlated posteriors.

We exclude KFAC-Laplace from the benchmarks because it requires specialized layer-specific implementations, which are not readily available for modern architectures like SWIN transformers.

C.2   Toy regression

We train a two-layer MLP with 10 hidden units per layer on a simple regression task. For uncertainty estimation, we sample from approximate posteriors of various methods, including projected posterior, loss-projected posterior, linearized Laplace, sampled Laplace, and diagonal Laplace.

While linearized Laplace and sampled Laplace refer to the same Gaussian distribution in parameter space, they differ in how predictions are generated. Linearized Laplace uses the linearized neural network for predictions and sampled Laplace uses neural network for predictions. The diagonal Laplace method, on the other hand, approximates the covariance of this Gaussian by only considering its diagonal, which leads to sparsely correlated posteriors. For a consistent and fair comparison, all methods utilize a prior precision of 1.0.

C.3   Image classification on MNIST and FMNIST

We train a standard LeNet for the MNIST and FashionMNIST experiments. We train LeNet with Adam optimizer and a 10^{-3} learning rate. We choose the prior precision for each baseline by doing a grid search over $\{0.1, 1.0, 5.0, 10.0, 50.0, 100.0\}$. For Projected Posterior and Loss Projected Posteriors, we use the analytical

expression for the optimal prior precision and we do 1000 iterations of alternating projections for both and a projection batch size, $S = 16$. For all Bayesian methods, we use 30 Monte Carlo samples for predictions. Whereas for SWAG we use a learning rate of 10^{-2} with momentum of 0.9 and weight decay of 3×10^{-4} and the low-rank covariance structure in all experiments. We collect 20 models to sample from the posterior. For Projected Posterior and Loss Projected Posterior, we use the linearized predictive and for Last-Layer and Diagonal baselines we use the neural network predictive.

C.4   Image classification on CIFAR-10 and SVHN

We train a ResNet architecture consisting of three groups of three ResNet blocks. The model is trained using Stochastic Gradient Descent (SGD) with a learning rate of 0.1, momentum, and weight decay. The prior precision is chosen in the same way as the experiment above, and the same predictive functions are applied.

For the projected posteriors, we perform 1000 iterations of alternating projections with a projection batch size of $S = 16$. All Bayesian methods utilize 30 Monte Carlo samples to compute predictions.

For SWAG on CIFAR-10, we use a learning rate of 10^{-2} with momentum of 0.9 and weight decay of 3×10^{-4} and the low-rank covariance structure in all experiments. We collect 20 models to sample from the posterior.

C.5   Image binary multi-classification on CelebA

We first remove from the training dataset all the images belonging to the classes 'Bald', 'Eyeglasses', and 'Mustache'. Then we train a VisualAttentionNetwork [Guo et al., 2023] with blocks of depths $(3, 3, 5, 2)$ and embedded dimensions of $(32, 64, 160, 256)$ with relu activation functions, we trained with Adam for 50 epochs with batch size 128 and a learning rate decreasing from 10^{-3} to 10^{-5}; parameter size is $P = 3858309$. All experiments are run on a single H100 GPU.

We compare the scores of a series of baselines including Max Logit [Hendrycks and Gimpel, 2016] and a Deep Ensemble (DE) [Lakshminarayanan et al., 2017] of 10 independently trained models, the high training cost is the reason why DE is missing standard deviations, since that would require training 30 models. We included several low-rank-approximation methods: Linearized Laplace Approximation (LLA) [Immer et al., 2021b], Local Ensemble (LE) [Madras et al., 2020], Stochastic Weight Averaging Gaussian (SWAG) [Maddox et al., 2019], Sketching Curvature for OoD Detection (SCOD) [Sharma et al., 2021] and Sketched Lanczos Uncertainty (SLU) [Miani et al., 2024]. All of these but SLU use a rank 300 approximation for memory limit, while SLU uses a rank 1000 and a sketch size of 1M. All of these but SCOD and SWAG are based on Lanczos algorithm, thus we also computed our Projected score using Lanczos algorithm to compute the top eigenvectors, this is referred to as 'Proj-Lanczos' in Table 5. Lastly, we also include the diagonal version of Laplace (LLA-d) which is a common method used for large models thanks to its low memory requirement.

C.6   Image classification on ImageNet

We obtain a pre-trained SWIN transformer[Liu et al., 2021], with 28 million parameters, from [DarshanDeshpande, 2021]. We sample from the Loss Projected Posterior. We use 5 Monte Carlo samples for predictions and do 15 iterations of alternating projections. We use a projection batch size of $S = 16$.

C.7   Generative model

We train a VAE with approximately $100,000$ parameters on MNIST and FMNIST. While keeping the encoder fixed we sample various decoders from the loss-projected posterior. We do 5 iterations of alternating projections and use 20 Monte Carlo Samples for predictions. Prior precision is chosen in the usual way.