Incorporating Neural Network Structure in the Bayesian Learning Rule

Eiki Shimizu[1]* **Mohammad Emtiyaz Khan**[2,3] **Thomas Möllenhoff**[3]

[1]The Institute of Statistical Mathematics, Japan
[2]TU Darmstadt & Hessian Center for AI, Darmstadt, Germany
[3]RIKEN Center for Advanced Intelligence Project, Japan

Abstract

The Bayesian Learning Rule (BLR) specifies how loss gradients update an exponential family (EF) posterior, but computing these gradients efficiently through a neural network (NN) requires exploiting its layered structure. We introduce a constrained Lagrangian formulation that connects EF duality with NN backpropagation, coupling forward propagation, backward sensitivities, and BLR posterior updates. Backpropagating through the resulting distributional states yields approximate BLR loss-site natural parameters. For Gaussian weight posteriors, we instantiate this framework with a sampling-free projection of layer-state means and covariances. Under delta-method localization, its forward and backward recursions recover the point-estimate forward pass, backpropagation, and Hessian backpropagation as special cases. More generally, this connects forward moment propagation with the corresponding backward propagation of distributional sensitivities, bringing together two perspectives often studied separately. We use the resulting sampling-free gradient and diagonal-Hessian estimates in IVON, replacing its Monte Carlo-based estimators while retaining its practical update rules.

1 Introduction

Learning in neural networks (NNs) couples three computations: a forward pass propagates representations, backpropagation computes loss sensitivities, and an optimization step updates the parameters. By exploiting the network's compositional structure, these computations can be carried out efficiently through a sequence of local operations. In variational Bayesian learning, parameters are represented by an approximate posterior distribution, and the corresponding computations involve expectations over uncertain networks that are generally intractable for deep nonlinear NNs. What is the variational Bayesian counterpart of these coupled forward, backward, and update computations?

The Bayesian Learning Rule (BLR) [Khan and Rue, 2023] provides the posterior update part of the answer. BLR updates an exponential family (EF) approximation $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ using natural gradients [Amari, 1998]. Write $\boldsymbol{\lambda}$ for its natural parameter, $\boldsymbol{\mu} = \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})]$ for its expectation parameter, and $\ell_n(\boldsymbol{\theta})$ for the loss on data example n. The corresponding expected loss is $\mathcal{L}_n(\boldsymbol{\mu}) := \mathbb{E}_{q_{\boldsymbol{\lambda}(\boldsymbol{\mu})}}[\ell_n(\boldsymbol{\theta})]$. Its *loss-site natural parameter* is

$$\widehat{\boldsymbol{\lambda}}_n := -\widetilde{\nabla}_{\boldsymbol{\lambda}} \mathcal{L}_n = -\nabla_{\boldsymbol{\mu}} \mathcal{L}_n, \tag{1}$$

where $\widetilde{\nabla}_{\boldsymbol{\lambda}}$ is the natural gradient and the equality follows from the classical EF duality. The linear site $\widehat{\ell}_n(\boldsymbol{\theta}) := -\langle \widehat{\boldsymbol{\lambda}}_n, \mathbf{T}(\boldsymbol{\theta}) \rangle$ contributes to the posterior natural parameter through the BLR update. The challenge is to compute these loss sites efficiently while respecting NN structure.

*Part of this work was done while affiliated with RIKEN Center for Advanced Intelligence Project.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

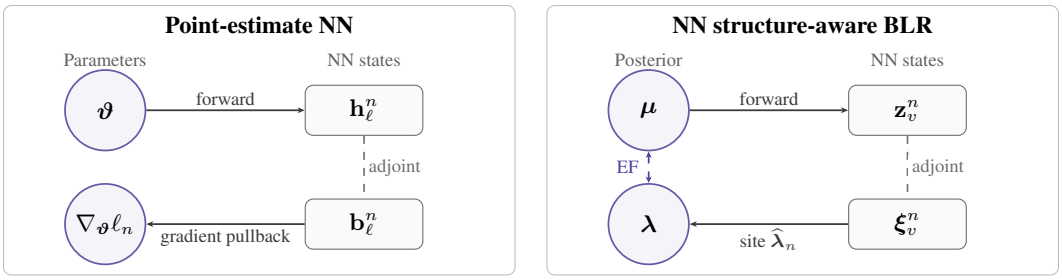


Figure 1: **From ordinary backpropagation to NN structure-aware BLR.** Our formulation extends the coupled forward, backward, and parameter-update computations of ordinary NN learning to variational Bayesian learning. Posterior expectation parameters $\boldsymbol{\mu}$ drive distributional forward propagation, while backward adjoints yield data-wise loss-site contributions $\widehat{\boldsymbol{\lambda}}_n$ to the global natural parameter $\boldsymbol{\lambda}$, connecting NN computational duality with EF duality.

Existing methods offer practical ways to tackle this challenge. Monte Carlo methods estimate loss derivatives by backpropagating through sampled networks, while sampling-free methods such as PBP [Hernández-Lobato and Adams, 2015] and DVI [Wu et al., 2019] propagate approximate activation moments and differentiate through them. Both approaches enable practical variational learning in nonlinear NNs, but they do not explicitly reveal a common structure linking the forward propagation of uncertainty, the backward propagation of gradient and curvature information, and the Bayesian posterior update. We establish this connection through a constrained Lagrangian formulation, in which backward sensitivities of distributional NN states yield approximate BLR loss-site natural parameters for the posterior update. Under a suitable Gaussian projection, the same forward–backward structure recovers ordinary backpropagation (BP) and Hessian backpropagation (HBP), connecting classical NN learning with variational learning.

To realize this connection, we introduce variables representing the distributions of intermediate NN states and impose two sets of constraints: posterior moments must agree with the EF expectation parameters, and intermediate states must satisfy the NN's local forward-propagation rules. The Lagrange multipliers associated with these constraints correspond, respectively, to the posterior natural parameters [Wainwright and Jordan, 2008] and the adjoints that propagate loss sensitivities backward through the NN [LeCun, 1988]. Their coupling allows the backward sensitivities to be propagated to posterior expectation coordinates, yielding the natural parameter contributions required by the BLR update. Figure 1 illustrates this formulation.

For Gaussian weight posteriors, we instantiate this framework with a projection that propagates layer-state means and covariances using Gaussian-averaged activation derivatives. Under delta-method localization, its forward and backward recursions recover the point-estimate forward pass, BP, and HBP without requiring posterior covariances to vanish. At finite variance, the same construction provides sampling-free mean gradient and diagonal Hessian estimates. We use these estimates in a new variant of IVON [Shen et al., 2024], replacing its Monte Carlo-based estimators while retaining its practical optimizer updates.

Beyond its algorithmic implications, our formulation connects two classical forms of duality: the natural–expectation parameter duality of EFs [Amari and Nagaoka, 2000] and the forward–adjoint duality underlying backpropagation, sensitivity analysis, and automatic differentiation [Tran and Leok, 2024, Baydin et al., 2018]. It also makes the Bayesian duality [Khan, 2025b,a] perspective concrete in NN learning: adjoint computations produce loss-site natural parameter contributions data-wise that are aggregated into the posterior update.

2 Preliminaries

2.1 Bayesian Learning Rule (BLR) and Bayesian Duality

Let $q_t(\boldsymbol{\theta}) = q_{\boldsymbol{\lambda}_t}(\boldsymbol{\theta})$ denote an exponential family (EF) variational posterior at iteration t, with $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta})\rangle - A(\boldsymbol{\lambda})\}$, where h is the EF carrier, $\mathbf{T}$ is the sufficient statistic, $\boldsymbol{\lambda}$ is the natural parameter, and $\boldsymbol{\mu} = \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\mathbf{T}(\boldsymbol{\theta})]$ is the expectation parameter. Given a prior $p_0(\boldsymbol{\theta})$ and

individual data losses $\ell_n(\boldsymbol{\theta})$, indexed by $n = 1, \ldots, N$, define the prior cost $\ell_0(\boldsymbol{\theta}) := \log h(\boldsymbol{\theta}) - \log p_0(\boldsymbol{\theta})$. The variational Bayes objective can then be written as

$$J(\boldsymbol{\lambda}) = \sum_{n=1}^{N} \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\ell_n(\boldsymbol{\theta})] + \mathrm{KL}(q_{\boldsymbol{\lambda}} \| p_0) = \sum_{n=0}^{N} \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\ell_n(\boldsymbol{\theta})] + \mathcal{R}_h(q_{\boldsymbol{\lambda}}).$$

Here $\mathcal{R}_h(q) := \mathbb{E}_q[\log(q/h)]$ denotes the entropy relative to the EF carrier h, which reduces to the negative entropy $-\mathcal{H}(q)$ up to an additive constant when h is constant. On the EF family, it satisfies $\nabla_{\boldsymbol{\mu}} \mathcal{R}_h(q_{\boldsymbol{\lambda}}) = \boldsymbol{\lambda}$. The prior term $\ell_0 = \log h - \log p_0$ ensures that $\mathcal{R}_h(q) + \mathbb{E}_q[\ell_0] = \mathrm{KL}(q \| p_0)$. Further details are given in Appendix B.1.

Write $\mathcal{L}_n(\boldsymbol{\mu}) := \mathbb{E}_{q_{\boldsymbol{\lambda}(\boldsymbol{\mu})}}[\ell_n(\boldsymbol{\theta})]$ for the exact expected loss, and $\mathcal{L}_0(\boldsymbol{\mu})$ for the expected prior cost. BLR performs natural gradient steps on this objective in the natural parameter $\boldsymbol{\lambda}$. For a regular EF, the Fisher information is $\mathcal{I}(\boldsymbol{\lambda}) = \nabla^2_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda})$, and for any differentiable scalar objective J, EF duality implies $\widetilde{\nabla}_{\boldsymbol{\lambda}} J = \mathcal{I}(\boldsymbol{\lambda})^{-1} \nabla_{\boldsymbol{\lambda}} J = \nabla_{\boldsymbol{\mu}} J$. Thus an ordinary gradient in expectation coordinates gives the natural gradient in the dual natural coordinates. We define the site natural parameter contribution of each prior or data cost as $\widehat{\boldsymbol{\lambda}}_{n,t}$, and for a step size $\rho_t > 0$, the BLR update becomes

$$\widehat{\boldsymbol{\lambda}}_{n,t} := -\widetilde{\nabla}_{\boldsymbol{\lambda}_t} \mathbb{E}_{q_t}[\ell_n(\boldsymbol{\theta})] = -\nabla_{\boldsymbol{\mu}_t} \mathbb{E}_{q_t}[\ell_n(\boldsymbol{\theta})], \qquad \boldsymbol{\lambda}_{t+1} = (1-\rho_t)\boldsymbol{\lambda}_t + \rho_t \sum_{n=0}^{N} \widehat{\boldsymbol{\lambda}}_{n,t}. \tag{2}$$

The loss-site representation makes the aggregation in (2) explicit. Define the iteration-local linear site $\widehat{\ell}_{n|t}(\boldsymbol{\theta}) := -\langle \widehat{\boldsymbol{\lambda}}_{n,t}, \mathbf{T}(\boldsymbol{\theta}) \rangle$. The BLR update then has the equivalent distributional form

$$q_{t+1}(\boldsymbol{\theta}) \propto h(\boldsymbol{\theta})^{\rho_t} q_t(\boldsymbol{\theta})^{1-\rho_t} \prod_{n=0}^{N} \exp\{-\rho_t \widehat{\ell}_{n|t}(\boldsymbol{\theta})\}.$$

Each site factor contributes $\rho_t \widehat{\boldsymbol{\lambda}}_{n,t}$ to the posterior natural parameter. This is the *Bayesian duality* view [Möllenhoff et al., 2026, Khan, 2025b,a]: information from individual data losses is transformed, via natural gradients, into site natural parameters that are aggregated in the posterior.

To illustrate how different choices and approximations within BLR recover familiar learning algorithms, consider a Gaussian posterior $q_t(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}_t, \mathbf{S}_t^{-1})$. For the Gaussian EF, h is constant in $\boldsymbol{\theta}$, so $\ell_0(\boldsymbol{\theta}) = -\log p_0(\boldsymbol{\theta})$ up to a constant. Let $\bar{\ell}(\boldsymbol{\theta}) := \sum_{n=0}^{N} \ell_n(\boldsymbol{\theta})$. Using Bonnet's and Price's identities [Price, 1958, Rezende et al., 2014], the BLR update becomes

$$\mathbf{S}_{t+1} = (1-\rho_t)\mathbf{S}_t + \rho_t \mathbb{E}_{q_t}[\nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})], \qquad \mathbf{m}_{t+1} = \mathbf{m}_t - \rho_t \mathbf{S}_{t+1}^{-1} \mathbb{E}_{q_t}[\nabla_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})] \tag{3}$$

Thus, Gaussian BLR expresses posterior updates in terms of expected gradients and Hessians. With suitable delta-method approximations, it recovers classical gradient and Newton-type methods. It also motivates scalable variational optimizers such as IVON [Shen et al., 2024], which maintains a diagonal Gaussian posterior and uses Monte Carlo samples to estimate gradient and curvature information, making variational learning practical for large NNs.

Further derivations and discussion are given in Appendix B.

2.2 Lagrangian View of Statistical and Computational Duality

Section 2.1 shows that BLR aggregates loss-site natural parameters $\widehat{\boldsymbol{\lambda}}_n = -\nabla_{\boldsymbol{\mu}} \mathbb{E}_q[\ell_n(\boldsymbol{\theta})]$ to update the posterior natural parameter $\boldsymbol{\lambda}$. The latter also admits a Lagrangian interpretation. For a fixed expectation parameter $\boldsymbol{\mu}$, consider

$$\mathfrak{L}_{\mathrm{EF}}(q, \boldsymbol{\lambda}, \alpha; \boldsymbol{\mu}) = \mathcal{R}_h(q) + \langle \boldsymbol{\lambda}, \boldsymbol{\mu} - \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] \rangle + \alpha \left(1 - \int q(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}\right).$$

Stationarity with respect to q gives the EF form, while stationarity with respect to $\boldsymbol{\lambda}$ enforces moment consistency:

$$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda})\}, \qquad \boldsymbol{\mu} = \nabla_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda}).$$

Thus, the posterior natural parameter $\boldsymbol{\lambda}$ is identified with the Lagrange multiplier enforcing *statistical moment consistency* [Wainwright and Jordan, 2008].

A complementary Lagrangian structure underlies backpropagation in a point estimate NN. Consider a network $\mathbf{h}_\ell = F_\ell(\mathbf{h}_{\ell-1}; \boldsymbol{\vartheta}_\ell), \quad \ell = 1, \dots, L$, with output loss $\ell^{\text{out}}(\mathbf{h}_L)$. Introducing multipliers $\mathbf{b}_\ell$ for the forward constraints gives

$$\mathfrak{L}_{\text{NN}}(\boldsymbol{\vartheta}, \mathbf{h}, \mathbf{b}) = \ell^{\text{out}}(\mathbf{h}_L) + \sum_{\ell=1}^{L} \langle \mathbf{b}_\ell, F_\ell(\mathbf{h}_{\ell-1}; \boldsymbol{\vartheta}_\ell) - \mathbf{h}_\ell \rangle.$$

Stationarity with respect to the network states yields the backward adjoint recursion, whose pullback to the parameters gives

$$\mathbf{b}_L = \nabla_{\mathbf{h}_L} \ell^{\text{out}}, \qquad \mathbf{b}_{\ell-1} = (D_{\mathbf{h}_{\ell-1}} F_\ell)^* \mathbf{b}_\ell, \qquad \nabla_{\boldsymbol{\vartheta}_\ell} \ell^{\text{out}} = (D_{\boldsymbol{\vartheta}_\ell} F_\ell)^* \mathbf{b}_\ell. \tag{4}$$

Here $(DF)^*$ denotes the adjoint of a derivative map DF under the chosen inner products, defined by $\langle \mathbf{b}, DF\, \mathbf{u} \rangle = \langle (DF)^* \mathbf{b}, \mathbf{u} \rangle$. The multipliers $\mathbf{b}_\ell$ enforce *computational consistency* while propagating loss sensitivities backward through the network [LeCun, 1988]. This is the Lagrangian interpretation of backpropagation, in which parameter gradients are assembled through local adjoint operations. Appendix C provides the corresponding derivations.

These constructions reveal two complementary dual structures: $\boldsymbol{\lambda}$ is associated with posterior moment consistency, while $\mathbf{b}_\ell$ are adjoints of the NN forward constraints. The key is to couple them so that neural sensitivities contribute directly to the BLR posterior update. The next section establishes this connection by lifting the variational objective onto distributional NN states whose propagation depends on $\boldsymbol{\mu}$. The resulting adjoint pullback to $\boldsymbol{\mu}$ yields the loss-site natural parameters $\widehat{\boldsymbol{\lambda}}_n$.

3 Incorporating NN Structure into BLR

We now incorporate NN structure into BLR for general computation graphs. We first introduce distributional states and local projection maps that approximate the NN forward computation. We then construct a constrained Lagrangian coupling these computations with posterior moment constraints and derive its stationarity conditions, which yield the distributional forward pass, adjoint backward pass, and loss-site natural parameters used in the BLR posterior update.

3.1 Setup: Distributional States

Let $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ be an EF posterior over the NN parameters. For each example n, let v index nodes of the neural computation graph, and let $\mathbf{z}_v^n$ denote a distributional state at node v. We write $\mathbf{z}_{\text{pa}(v)}^n := \{\mathbf{z}_u^n : u \in \text{pa}(v)\}$ for the collection of states at its parent nodes. These states are propagated through local maps

$$\mathbf{z}_v^n = \Psi_v^n\big(\mathbf{z}_{\text{pa}(v)}^n; \boldsymbol{\mu}\big), \tag{5}$$

where $\boldsymbol{\mu}$ denotes the weight posterior expectation parameters, which enter the maps for operations involving NN weights.

Each map Ψ_v^n specifies how a local NN operation propagates the chosen moment state. Rather than tracking the generally intractable full distribution at each node, $\mathbf{z}_v^n$ stores a finite-dimensional summary, such as distributional moments. For example, if a node is represented by an EF approximation q_v^n with sufficient statistic $\mathbf{T}_v$, one may take $\mathbf{z}_v^n = \mathbb{E}_{q_v^n}[\mathbf{T}_v]$. Different moment approximations and localization rules can therefore be represented through the choice of $\mathbf{z}_v^n$ and Ψ_v^n.

For example, a sequence of affine operations and nonlinear activations has preactivation and postactivation states, $\mathbf{z}_\ell^{n,a} = \Psi_\ell^a(\mathbf{z}_{\ell-1}^{n,h}; \boldsymbol{\mu}), \quad \mathbf{z}_\ell^{n,h} = \Psi_\ell^h(\mathbf{z}_\ell^{n,a})$. The affine map depends on the weight posterior, while the activation map propagates moments through a parameter-free nonlinearity. For Gaussian approximations, these moment states can be represented by means and covariances, as developed in Section 4.

Let $\mathbf{z}_{n,\text{out}}$ denote the output state and $\mathcal{L}_n^{\text{proj}}(\mathbf{z}_{n,\text{out}})$ the corresponding approximate expected loss. The local maps $\{\Psi_v^n\}$ and the output loss together define a function of the posterior expectation parameter $\boldsymbol{\mu}$. We distinguish this projected objective from the exact expected loss $\mathcal{L}_n(\boldsymbol{\mu}) = \mathbb{E}_q[\ell_n(\boldsymbol{\theta})]$. They agree only when the selected propagation and output rules evaluate the posterior expectation exactly.

3.2 Lagrangian Formulation for BLR

We now incorporate the distributional state propagation of Section 3.1 into BLR. Starting from the variational Bayes objective in Section 2.1, define the prior contribution in expectation coordinates as $\mathcal{L}_0(\boldsymbol{\mu}) := \mathbb{E}_{q_{\boldsymbol{\lambda}(\boldsymbol{\mu})}}[\ell_0(\boldsymbol{\theta})]$. We treat q and $\boldsymbol{\mu}$ as separate variables linked by the posterior moment constraint $\boldsymbol{\mu} = \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})]$. We also impose the NN forward constraints (5), with $\mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}})$ approximating each expected data loss. Introducing multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\xi}_v^n$ gives

$$\begin{aligned}\mathfrak{L}_{\mathrm{BLR}}(q, \boldsymbol{\mu}, \boldsymbol{\lambda}, \mathbf{z}, \boldsymbol{\xi}) = &\underbrace{\sum_n \mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}}) + \mathcal{L}_0(\boldsymbol{\mu}) + \mathcal{R}_h(q)}_{\text{projected variational Bayes objective}} \\ &+ \underbrace{\langle \boldsymbol{\lambda}, \boldsymbol{\mu} - \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] \rangle}_{\text{posterior moment constraint}} + \underbrace{\sum_{n,v} \langle \boldsymbol{\xi}_v^n, \Psi_v^n(\mathbf{z}_{\mathrm{pa}(v)}^n; \boldsymbol{\mu}) - \mathbf{z}_v^n \rangle}_{\text{NN forward constraints}},\end{aligned} \tag{6}$$

where the normalization multiplier for q is omitted. We now derive the stationarity conditions, which yield statistical moment consistency, distributional forward propagation, and adjoint backward propagation. Crucially, their coupling through the shared expectation parameter $\boldsymbol{\mu}$ turns the adjoint pullback into the loss-site natural parameters required by BLR.

Statistical moment consistency. Stationarity with respect to $\boldsymbol{\lambda}$ enforces $\boldsymbol{\mu} = \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})]$, while stationarity with respect to q gives $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda})\}$. Thus, as in Section 2.2, the multiplier enforcing statistical moment consistency is the posterior natural parameter $\boldsymbol{\lambda}$. The statistical lifting therefore preserves the EF variational family, while approximations enter through the projected NN propagation and loss.

Distributional forward pass. Stationarity with respect to $\boldsymbol{\xi}_v^n$ enforces the local propagation constraints, recovering the distributional forward computation in (5). Unlike the point-valued activations of an ordinary NN forward pass, the states $\mathbf{z}_v^n$ propagate distributional information, such as activation moments, through the network.

Adjoint backward pass. Stationarity with respect to $\mathbf{z}_u^n$ gives the reverse recursion

$$\boldsymbol{\xi}_u^n = \mathbf{c}_u^n + \sum_{v \in \mathrm{ch}(u)} \left(D_{\mathbf{z}_u^n} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n, \tag{7}$$

where $\mathrm{ch}(u)$ denotes the children of node u, and $\mathbf{c}_{n,\mathrm{out}} = \nabla_{\mathbf{z}_{n,\mathrm{out}}} \mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}})$, $\quad \mathbf{c}_u^n = 0$ otherwise. The multipliers $\boldsymbol{\xi}_v^n$ therefore act as computational adjoints, propagating loss sensitivities backward through the distributional state computation. This parallels ordinary backpropagation, but the sensitivities are now taken with respect to distributional states rather than point-valued activations.

Loss-site natural parameters and BLR update. At parameterized nodes, the local propagation maps depend on the posterior expectation parameter $\boldsymbol{\mu}$. Pulling the adjoints back through these maps gives the loss-site natural parameter

$$\widehat{\boldsymbol{\lambda}}_n = -\sum_v \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n. \tag{8}$$

The adjoint pullback thus gives the negative gradient of the projected expected loss with respect to $\boldsymbol{\mu}$. It approximates the exact BLR loss site $-\nabla_{\boldsymbol{\mu}} \mathcal{L}_n(\boldsymbol{\mu})$ in (1), and agrees with it when the projected loss and its derivative are exact. Thus, backpropagation through the NN's moment states provides approximate loss-site natural parameters for the BLR update.

For the prior, define $\widehat{\boldsymbol{\lambda}}_{0,t} := -\nabla_{\boldsymbol{\mu}_t} \mathcal{L}_0(\boldsymbol{\mu}_t)$. At each iteration, the prior site is computed from the prior cost, while the data loss sites are obtained through the distributional forward and backward computations. These contributions are then combined using the original BLR update (2). Thus, our framework changes how the loss sites are computed, not the posterior update itself.

Coupling the two dual structures. The coupling between the posterior natural parameter and computational adjoints becomes explicit from stationarity with respect to their shared $\boldsymbol{\mu}$. Using (8) and $\widehat{\boldsymbol{\lambda}}_0 := -\nabla_{\boldsymbol{\mu}} \mathcal{L}_0(\boldsymbol{\mu})$, we obtain $\mathbf{0} = \boldsymbol{\lambda} - \widehat{\boldsymbol{\lambda}}_0 - \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n$. Hence, at a stationary point,

$$\boldsymbol{\lambda}^\star = \widehat{\boldsymbol{\lambda}}_0^\star + \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n^\star = \widehat{\boldsymbol{\lambda}}_0^\star - \sum_{n=1}^{N} \sum_{v} \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^{n,\star}. \tag{9}$$

Equation (9) expresses the posterior natural parameter as an aggregation of prior information and contributions from individual data examples, giving a data-wise decomposition of Bayesian duality [Khan, 2025b,a]. Our formulation makes this decomposition architecture-aware: each loss-site natural parameter is constructed through local NN computations, by propagating distributional states forward and loss sensitivities backward through adjoint operations. This provides a layerwise realization of the data-to-parameter aggregation underlying Bayesian duality. Also note that for a same-EF prior, $\mathcal{L}_0(\boldsymbol{\mu}) = -\langle \boldsymbol{\lambda}_0, \boldsymbol{\mu} \rangle + \text{const}$ and thus $\widehat{\boldsymbol{\lambda}}_0 = \boldsymbol{\lambda}_0$, recovering $\boldsymbol{\lambda}^\star - \boldsymbol{\lambda}_0 = \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n^\star$.

4 Gaussian Instantiation and Delta-Method Recovery

We instantiate the framework of Section 3 with a Gaussian parameter posterior and Gaussian approximations to the layer states. The construction below concerns local activation computations and does not assume a particular NN architecture. Our projection is designed to satisfy two properties: it propagates finite-variance information using tractable local Gaussian approximations, and, under delta-method localization of its coefficients, it recovers the forward pass, backpropagation (BP), and Hessian backpropagation (HBP) of a point estimate NN.

4.1 Gaussian Layer States and Averaged Sensitivities

We represent each Gaussian layer state by its mean and covariance, $\mathbf{z}_v = (\mathbf{m}_v, \boldsymbol{\Sigma}_v)$, suppressing the data index where unambiguous. For a Gaussian distribution, these are equivalent to the canonical EF expectation coordinates, since $\mathbb{E}[\mathbf{x}\mathbf{x}^\mathsf{T}] = \boldsymbol{\Sigma} + \mathbf{m}\mathbf{m}^\mathsf{T}$. We use the mean–covariance representation to make the propagation of gradients and Hessian explicit, in line with the Gaussian BLR update (3). For affine operations, the required moments can often be computed analytically under suitable assumptions, as discussed in Section 6.1; here we focus on nonlinear activations.

Consider an activation $\mathbf{h} = \sigma(\mathbf{a})$ with Gaussian preactivation $\mathbf{a} \sim \mathcal{N}(\mathbf{m}_a, \boldsymbol{\Sigma}_a)$. Define the Gaussian-averaged activation, Jacobian, and Hessians by

$$\boldsymbol{\eta} := \mathbb{E}[\sigma(\mathbf{a})], \qquad \bar{\mathbf{J}} := \mathbb{E}[D\sigma(\mathbf{a})], \qquad \bar{\mathbf{H}}_i := \mathbb{E}[\nabla^2 \sigma_i(\mathbf{a})]. \tag{10}$$

Bonnet's and Price's identities [Price, 1958, Rezende et al., 2014] give $D_{\mathbf{m}_a} \boldsymbol{\eta} = \bar{\mathbf{J}}, \quad 2D_{\boldsymbol{\Sigma}_a} \boldsymbol{\eta}_i = \bar{\mathbf{H}}_i$. Thus, the Gaussian-averaged Jacobian and Hessians describe how the expected activation changes with its input mean and covariance, respectively. These identities allow us to propagate both gradient and curvature information backward.

In the mean–covariance representation, we express the computational adjoint $\boldsymbol{\xi}_v$ by a gradient channel $\mathbf{g}_v$ and a Hessian channel $\mathbf{H}_v$, with pairing $\langle \boldsymbol{\xi}_v, \delta \mathbf{z}_v \rangle = \langle \mathbf{g}_v, \delta \mathbf{m}_v \rangle + \frac{1}{2} \langle \mathbf{H}_v, \delta \boldsymbol{\Sigma}_v \rangle_F$. Here $\mathbf{g}_v$ measures sensitivity to the mean, while $\mathbf{H}_v$ is twice the sensitivity to the covariance, matching the gradient and curvature quantities in Gaussian BLR. The corresponding change from canonical Gaussian expectation coordinates is detailed in Appendix E.

4.2 A Backpropagation-Compatible Gaussian Projection

To propagate a Gaussian state through a nonlinear activation, we need to determine the output mean and covariance. We use the Gaussian-averaged activation $\boldsymbol{\eta}$ for the mean and approximate the covariance using the averaged Jacobian. At iteration t, this gives the projection

$$\Psi_{\mathrm{FO},t}^{\mathrm{loc}}(\mathbf{m}_a, \boldsymbol{\Sigma}_a) := \left(\boldsymbol{\eta}(\mathbf{m}_a, \boldsymbol{\Sigma}_a), \bar{\mathbf{J}}_t \boldsymbol{\Sigma}_a \bar{\mathbf{J}}_t^\mathsf{T}\right), \tag{11}$$

where $\bar{\mathbf{J}}_t$ is evaluated at the current Gaussian state. The covariance approximation follows the usual first-order (FO) linearization rule, but uses a Jacobian averaged over the uncertain preactivation.

We design the corresponding backward computation to retain the structure of ordinary BP and HBP. Specifically, when differentiating the covariance approximation, we hold $\bar{\mathbf{J}}_t$ fixed, while keeping the mean map $\boldsymbol{\eta}$ differentiable. Using the Gaussian identities above, the resulting adjoint is

$$\mathbf{g}_a = \bar{\mathbf{J}}_t^\mathsf{T} \mathbf{g}_h, \qquad \mathbf{H}_a = \bar{\mathbf{J}}_t^\mathsf{T} \mathbf{H}_h \bar{\mathbf{J}}_t + \sum_i (\mathbf{g}_h)_i \bar{\mathbf{H}}_{i,t}. \tag{12}$$

The gradient is propagated through the averaged Jacobian. For Hessian, the first term propagates curvature from the next layer, while the second accounts for the nonlinearity of the activation. This has the same structure as HBP, with pointwise derivatives replaced by Gaussian averages.

Holding $\bar{\mathbf{J}}_t$ fixed in the covariance approximation is a deliberate design choice, implemented by stopping gradients through this coefficient; we still differentiate the mean map $\boldsymbol{\eta}$. Without this convention, derivatives of $\bar{\mathbf{J}}$ would add terms to the curvature recursion, so the backward pass would no longer take the HBP form. We therefore interpret (11) as an iteration-local projection with the derivative convention specified here. A second-order covariance projection is given in Appendix F.2.

4.3 Delta-Method Recovery of Point Estimate BP and HBP

We now show how the Gaussian projection recovers ordinary point estimate NN computation. Following the delta-method approximation used in Gaussian BLR, we replace the Gaussian averages by evaluations at the preactivation mean $\boldsymbol{\eta} \approx \sigma(\mathbf{m}_a)$, $\quad \bar{\mathbf{J}} \approx D\sigma(\mathbf{m}_a)$, $\quad \bar{\mathbf{H}}_i \approx \nabla^2 \sigma_i(\mathbf{m}_a)$. These approximations are applied to the activation mean and its sensitivity coefficients separately. In particular, the Hessian coefficient is retained even when the mean value is evaluated at a point, and no zero-covariance limit is required.

The forward mean then becomes $\mathbf{m}_h = \sigma(\mathbf{m}_a)$, recovering the point-estimate activation. Writing $\mathbf{J} := D\sigma(\mathbf{m}_a)$ and $\mathbf{H}_i := \nabla^2 \sigma_i(\mathbf{m}_a)$, the backward equations (12) reduce to

$$\mathbf{g}_a = \mathbf{J}^\mathsf{T} \mathbf{g}_h, \qquad \mathbf{H}_a = \mathbf{J}^\mathsf{T} \mathbf{H}_h \mathbf{J} + \sum_i (\mathbf{g}_h)_i \mathbf{H}_i. \tag{13}$$

These are precisely the standard BP and HBP recursions [Becker and LeCun, 1989, Elsayed et al., 2024]. Together with corresponding pointwise rules for the network's other operations, including affine modules and the output loss, this recovers the point estimate forward pass, BP, and HBP along the posterior-mean trajectory.

In variational Bayesian learning, the posterior represents parameter uncertainty, making it natural to retain the resulting variance information in the layer states and use Gaussian-averaged coefficients for forward and backward propagation. Our projection thereby provides sampling-free approximations to gradients and Hessian that account for finite-variance uncertainty.

5 Connections to Existing Work

Moment propagation and localization. Sampling-free uncertainty propagation in NNs has been studied using various local approximations. For example, Li et al. [2025] use local linearization for Gaussian prediction, corresponding to the forward part of our first-order projection under delta-method localization. Probabilistic backpropagation (PBP) and deterministic variational inference (DVI) [Hernández-Lobato and Adams, 2015, Wu et al., 2019] instead propagate approximate activation moments and differentiate through them for Bayesian learning. These methods primarily focus on constructing tractable moment approximations, with backward derivatives used in their respective inference procedures. Their forward computations can be represented by alternative choices of moment states and propagation maps in our framework, although their complete inference algorithms differ from the BLR update. For nonlinear activations, moment matching generally gives $\mathbb{E}[\sigma(\mathbf{a})] \neq \sigma(\mathbf{m}_a)$ at finite covariance, so recovering the point-estimate forward pass requires taking the zero-covariance limit. In contrast, our delta-localized projection recovers point-estimate forward propagation, BP, and HBP without requiring this limit. Table 1 summarizes these recoveries and connections to existing methods, including IVON. More broadly, our Lagrangian formulation explicitly couples distributional forward propagation, backward sensitivities, and BLR posterior updates, with adjoint pullbacks yielding approximate loss-site natural parameters.

Table 1: **BP/HBP recovery and connections to related methods.** Point-estimate BP and HBP are recovered under delta-method localization of the Gaussian activation coefficients. Other rows describe connections at the level of propagation rules or loss-site estimators.

Method	Choice in our framework	Recovery or connection
BP	Delta-localized backward gradients	Recovers point-estimate BP
HBP	Delta-localized gradients and Hessians	Recovers point-estimate HBP
Local linearization	Delta-localized forward moments	Recovers forward rule
PBP / DVI	Moment-matching maps	Connects forward propagation
IVON	Sampled NN computation maps	MC estimates of BLR sites

Dual variational inference. A related Lagrangian formulation appears in dual variational Gaussian inference [Khan et al., 2013], which uses auxiliary Gaussian moments and convex duality to construct an inference algorithm. Its data-site moment constraints resemble our computational-consistency constraints for a single linear layer. Although it also works with Gaussian mean and covariance parameters, these are simply an invertible reparameterization of EF expectation coordinates. Our focus is different: we couple adjoints of nonlinear NN computation graphs to the expectation-coordinate gradients required by BLR, without relying on global convexity of the NN objective.

6 Affine-Operation Instantiation and NN Structure-Aware IVON

We now specify the preactivation map Ψ_ℓ^a for affine operations, including dense and convolutional layers, and use the resulting distributional forward and backward computations in IVON. We consider factorized Gaussian parameter posteriors and diagonal Gaussian layer states, for which preactivation means and marginal variances can be computed efficiently [Wu et al., 2019, Li et al., 2025]. The nonlinear activation projection follows Section 4.2.

6.1 Affine Operations with Factorized Gaussian Parameters and Diagonal Layer States

Consider a layer with an affine preactivation operation followed by an elementwise GeLU activation. For a dense layer, this is $\mathbf{a}_\ell^n = \mathbf{W}_\ell \mathbf{h}_{\ell-1}^n + \boldsymbol{d}_\ell, \quad \mathbf{h}_\ell^n = \mathrm{GeLU}(\mathbf{a}_\ell^n)$. For a convolutional layer, the matrix product is replaced by convolution with shared filters. We assume a factorized Gaussian posterior over the distinct trainable weights and biases, writing for the dense case $(\mathbf{W}_\ell)_{ij} \sim \mathcal{N}((\mathbf{M}_\ell)_{ij}, (\mathbf{V}_\ell)_{ij})$, and $(\boldsymbol{d}_\ell)_i \sim \mathcal{N}((\mathbf{c}_\ell)_i, (\mathbf{s}_\ell)_i)$. The same notation $\mathbf{M}_\ell, \mathbf{V}_\ell$ denotes the means and variances of the shared filter parameters for convolutional layers. For each example n, we approximate the preactivation and activation states by diagonal Gaussians, writing $\mathbf{z}_\ell^{n,a} = (\mathbf{m}_\ell^{n,a}, \mathrm{Diag}(\mathbf{v}_\ell^{n,a}))$ and $\mathbf{z}_\ell^{n,h} = (\mathbf{m}_\ell^{n,h}, \mathrm{Diag}(\mathbf{v}_\ell^{n,h}))$.

Preactivation projection. Using the diagonal approximation to the incoming layer state and independence of the current-layer parameters, the preactivation mean and diagonal variance for a dense layer are

$$\mathbf{m}_\ell^{n,a} = \mathbf{M}_\ell \mathbf{m}_{\ell-1}^{n,h} + \mathbf{c}_\ell, \qquad \mathbf{v}_\ell^{n,a} = (\mathbf{M}_\ell^{\circ 2} + \mathbf{V}_\ell)\mathbf{v}_{\ell-1}^{n,h} + \mathbf{V}_\ell (\mathbf{m}_{\ell-1}^{n,h})^{\circ 2} + \mathbf{s}_\ell. \tag{14}$$

These analytic moment equations define Ψ_ℓ^a. The variance includes uncertainty from both the incoming activations and the weights.

For a convolutional layer, the corresponding equations have the same form, with matrix multiplication replaced by convolution:

$$\mathbf{m}_\ell^{n,a} = \mathrm{Conv}_{\mathbf{M}_\ell}(\mathbf{m}_{\ell-1}^{n,h}) + \mathbf{c}_\ell, \quad \mathbf{v}_\ell^{n,a} = \mathrm{Conv}_{\mathbf{M}_\ell^{\circ 2} + \mathbf{V}_\ell}(\mathbf{v}_{\ell-1}^{n,h}) + \mathrm{Conv}_{\mathbf{V}_\ell}((\mathbf{m}_{\ell-1}^{n,h})^{\circ 2}) + \mathbf{s}_\ell. \tag{15}$$

Here $\mathrm{Conv}_{\boldsymbol{K}}$ denotes convolution with filter $\boldsymbol{K}$, and the bias moments are broadcast across spatial locations. These formulas compute the marginal output moments under the stated independence and diagonal-input approximations. Parameter sharing is preserved in the convolution, although the diagonal layer-state approximation discards correlations between output locations induced by shared weights and overlapping receptive fields.

Activation projection. We instantiate the activation map Ψ_ℓ^h using GeLU and apply the first-order Gaussian projection of Section 4.2 elementwise. For $A \sim \mathcal{N}(\mu, v)$, define the Gaussian-averaged coefficients $\eta(\mu, v) = \mathbb{E}[\mathrm{GeLU}(A)], \quad j(\mu, v) = \partial_\mu \eta(\mu, v), \quad \kappa(\mu, v) = 2\partial_v \eta(\mu, v)$. These correspond to the averaged activation, first derivative, and second derivative in (10). Our main variant, *Diag-FO*, uses η for the activation mean and j for the first-order variance approximation, while j and κ determine the backward gradient and curvature propagation. Closed-form expressions for these coefficients are given in Appendix E.6, including ReLU as a limiting case of the generalized GeLU family. The corresponding forward and backward equations are provided in Appendix F.2.

6.2 NN Structure-Aware IVON

We use these projected forward and backward computations to supply the gradient and diagonal-Hessian information required by IVON [Shen et al., 2024].

Standard IVON. Write $\ell_{\mathrm{data}}(\boldsymbol{\theta}) := \sum_{n=1}^N \ell_n(\boldsymbol{\theta})$ for the data loss, excluding the prior term ℓ_0. IVON maintains a diagonal Gaussian posterior $q_t(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}_t, \mathrm{Diag}(\boldsymbol{\sigma}_t^{\circ 2}))$. At each iteration, it samples $\boldsymbol{\theta}_t = \mathbf{m}_t + \boldsymbol{\sigma}_t \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and differentiates the sampled NN. The gradient estimators are $\widehat{\mathbf{g}}_{\mathrm{IVON}} = \widehat{\nabla} \ell_{\mathrm{data}}(\boldsymbol{\theta}_t), \quad \widehat{\mathbf{h}}_{\mathrm{IVON}} = \widehat{\mathbf{g}}_{\mathrm{IVON}} \odot \frac{\boldsymbol{\theta}_t - \mathbf{m}_t}{\boldsymbol{\sigma}_t^{\circ 2}}$. Gaussian integration by parts gives $\mathbb{E}[\widehat{\mathbf{h}}_{\mathrm{IVON}}] = \mathbb{E}_{q_t}[\mathrm{diag}(\nabla^2_{\boldsymbol{\theta}} \ell_{\mathrm{data}}(\boldsymbol{\theta}))]$, where the expectation also includes any unbiased minibatch sampling. Thus IVON estimates the expected gradient and diagonal Hessian through sampled networks, without explicitly backpropagating Hessians.

NN structure-aware IVON. Our variant replaces these Monte Carlo estimators by gradients and Hessian obtained from the projected NN computation. Let $\mathcal{L}^{\mathrm{proj}}_{\mathrm{data},t}(\mathbf{m}, \boldsymbol{\sigma}^{\circ 2})$ denote the sum $\sum_{n=1}^N \mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}})$, viewed as a function of the Gaussian parameter mean and variance through the iteration-local propagation rules of Section 4.2. Evaluating derivatives at the current posterior gives $\widehat{\mathbf{g}}_{\mathbf{m}} = \nabla_{\mathbf{m}} \mathcal{L}^{\mathrm{proj}}_{\mathrm{data},t}, \quad \widehat{\mathbf{h}} = 2\nabla_{\boldsymbol{\sigma}^{\circ 2}} \mathcal{L}^{\mathrm{proj}}_{\mathrm{data},t}$. The derivatives are computed by distributional forward propagation and adjoint backpropagation, with the coefficient convention specified in Section 4.2. We substitute $(\widehat{\mathbf{g}}_{\mathbf{m}}, \widehat{\mathbf{h}})$ for $(\widehat{\mathbf{g}}_{\mathrm{IVON}}, \widehat{\mathbf{h}}_{\mathrm{IVON}})$. The practical IVON optimizer also contains momentum and a quadratic correction to its curvature update; we retain those terms and its treatment of prior regularization. Thus our change concerns how the gradient and curvature are estimated, not the entire optimizer. Algorithm 2 in Appendix H summarizes the procedure.

Comparison. The posterior expected gradients and Hessians required by Gaussian BLR are generally intractable for nonlinear NNs. Standard IVON uses Monte Carlo samples and obtains unbiased raw estimators, but these can have substantial sampling variance. Our NN structure-aware variant instead uses sampling-free Gaussian layer-state approximations and their adjoints, avoiding parameter sampling variance at the cost of projection bias. For dense and convolutional affine operations, marginal Gaussian moments are inexpensive to compute, but for architectures with more complicated operations, accurate and efficient moment projections may be difficult to obtain, making standard IVON more broadly applicable. Appendix H.2 shows how to extend the distributional forward–adjoint formulation itself to stochastic propagation maps: ordinary backpropagation through sampled networks then yields Monte Carlo estimates of the same BLR loss-site natural parameters. Both approaches target gradient and Hessian quantities motivated by Gaussian BLR, but evaluate them through sampled computations or projected moment computations, respectively; the practical IVON update additionally includes momentum and curvature correction. This parallels the familiar trade-off between analytic approximations and Monte Carlo estimation in variational inference.

7 Numerical Results

We investigate whether NN structure-aware forward and backward computations improve upon standard IVON, a BLR-based optimizer. We evaluate MLPs on MNIST, Fashion-MNIST, and UCI regression benchmarks, and a CNN on CIFAR-10. All methods retain the same IVON posterior update, differing only in their gradient and diagonal-Hessian estimators. We focus on Diag-FO and IVON, as Diag-SO gives similar results. Experimental details and further results are given in Appendix I, with additional comparisons and ablations in Appendix J.

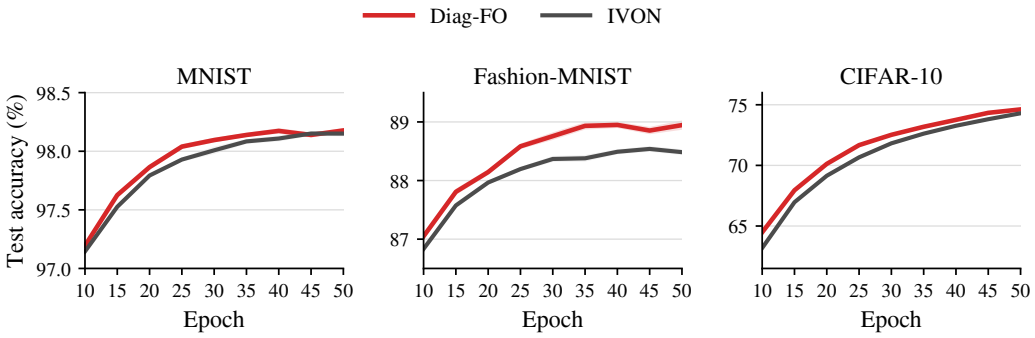


Figure 2: Test accuracy for IVON and Diag-FO on MNIST, Fashion-MNIST, and CIFAR-10. Curves show the mean ± standard error over 20 seeds (epochs 10–50).

Table 2: Final classification test performance at epoch 50 (mean ± sample standard deviation over 20 seeds). Higher accuracy and lower NLPD/ECE are better. Best displayed means are bolded.

Dataset	Method	Accuracy (%) ↑	NLPD ↓	ECE ↓
MNIST	IVON	98.15 ± 0.10	0.0859 ± 0.0052	0.0348 ± 0.0051
	Diag-FO	**98.18 ± 0.09**	**0.0824 ± 0.0057**	**0.0311 ± 0.0056**
Fashion-MNIST	IVON	88.48 ± 0.22	0.3299 ± 0.0058	0.0258 ± 0.0062
	Diag-FO	**88.95 ± 0.36**	**0.3238 ± 0.0069**	**0.0105 ± 0.0028**
CIFAR-10	IVON	74.30 ± 0.59	0.7629 ± 0.0138	0.0365 ± 0.0055
	Diag-FO	**74.62 ± 0.49**	**0.7480 ± 0.0116**	**0.0196 ± 0.0055**

Classification. Figure 2 compares test accuracy throughout training. Both methods improve rapidly during the earlier epochs, but Diag-FO reaches higher accuracy earlier, particularly on Fashion-MNIST and CIFAR-10, while performing similarly to IVON on MNIST. Table 2 shows that Diag-FO also achieves lower test NLPD and ECE on all three datasets. The final accuracy gains are modest, but the improvements in ECE are more pronounced on Fashion-MNIST and CIFAR-10, indicating better predictive calibration. Diag-SO yields similar final performance and is included in the additional results in Appendix I.

UCI regression. Diag-FO achieves lower mean test RMSE and NLPD than standard IVON on all eight UCI datasets (Table 6 in Appendix I), with particularly pronounced improvements on Naval and Yacht.

8 Conclusion and Limitations

We introduced an NN structure-aware extension of the Bayesian Learning Rule (BLR) that couples distributional forward propagation, backward adjoints, and posterior updates through a constrained Lagrangian. Adjoint pullbacks through projected layer moments yield approximate BLR loss-site natural parameters, connecting exponential family duality with NN computation. For Gaussian states, delta-method localization recovers point-estimate forward propagation, ordinary backpropagation (BP), and Hessian backpropagation (HBP) without requiring a zero-covariance limit. We demonstrated this connection by replacing IVON's Monte Carlo gradient and diagonal-Hessian estimators with sampling-free computations while retaining its posterior update. Experiments on MLPs and a CNN show improvements over standard IVON, particularly in predictive calibration. Our current implementation relies on diagonal Gaussian approximations, which neglect correlations and may accumulate projection errors with depth. Extending the framework to more complex architectures requires efficient and reliable moment projections and their adjoints. Further discussion of limitations and future directions is provided in Appendix K.

Acknowledgments and Disclosure of Funding

This work is supported by the Bayes duality project, JST CREST Grant Number JPMJCR2112. TM acknowledges the support of JSPS KAKENHI Grant Number 26H02541. The authors declare no competing interests.

References

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998. doi: 10.1162/089976698300017746.

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society, 2000. doi: 10.1090/mmono/191.

Atılım Güneş Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018. URL https://www.jmlr.org/papers/v18/17-468.html.

Sue Becker and Yann LeCun. Improving the convergence of back-propagation learning with second order methods. In David S. Touretzky, Geoffrey E. Hinton, and Terrence J. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 29–37, San Mateo, CA, 1989. Morgan Kaufmann. URL https://yann.lecun.com/exdb/publis/pdf/becker-lecun-89.pdf.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

Mohamed Elsayed, Homayoon Farrahi, Felix Dangel, and A. Rupam Mahmood. Revisiting scalable Hessian diagonal approximations for applications in reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2024.

José Miguel Hernández-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning (ICML)*, 2015.

Mohammad Emtiyaz Khan. Information geometry of variational Bayes. *Information Geometry*, 8: 275–289, 2025a. doi: 10.1007/s41884-025-00174-3.

Mohammad Emtiyaz Khan. Knowledge adaptation as posterior correction, 2025b. URL https://arxiv.org/abs/2506.14262.

Mohammad Emtiyaz Khan and Håvard Rue. The Bayesian learning rule. *Journal of Machine Learning Research*, 24(281):1–46, 2023. URL http://jmlr.org/papers/v24/22-0291.html.

Mohammad Emtiyaz Khan, Aleksandr Aravkin, Michael Friedlander, and Matthias Seeger. Fast dual variational inference for non-conjugate latent gaussian models. In *International Conference on Machine Learning (ICML)*, 2013.

Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. In *NeurIPS 2021 Workshop on Differentiable Programming*, 2021.

Yann LeCun. A theoretical framework for back-propagation. In David S. Touretzky, Geoffrey E. Hinton, and Terrence J. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28, San Mateo, CA, 1988. Morgan Kaufmann. URL https://yann.lecun.com/exdb/publis/pdf/lecun-88.pdf.

Rui Li, Marcus Klasson, Arno Solin, and Martin Trapp. Streamlining prediction in Bayesian deep learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Adrian Robert Minut, Nico Daheim, Marco Miani, Mohammad Emtiyaz Khan, Wu Lin, and Thomas Möllenhoff. SOAP-Bubbles: Effective and scalable variational learning with structured covariances. In *Advances in Neural Information Processing Systems*, 2026.

Thomas Möllenhoff, Siddharth Swaroop, Finale Doshi-Velez, and Mohammad Emtiyaz Khan. Federated ADMM from Bayesian duality. In *International Conference on Learning Representations (ICLR)*, 2026.

Robert Price. A useful theorem for nonlinear devices having gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72, 1958. doi: 10.1109/TIT.1958.1057444.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, 2014.

Yuesong Shen, Nico Daheim, Bai Cong, Peter Nickl, Gian Maria Marconi, Bazan Clement Emile Marcel Raoul, Rio Yokota, Iryna Gurevych, Daniel Cremers, Mohammad Emtiyaz Khan, and Thomas Möllenhoff. Variational learning is effective for large deep networks. In *International Conference on Machine Learning (ICML)*, 2024.

Brian Kha Tran and Melvin Leok. Geometric methods for adjoint systems. *Journal of Nonlinear Science*, 34:25, 2024. doi: 10.1007/s00332-023-09999-7.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008. doi: 10.1561/2200000001.

Tobias Jan Wieczorek, Leon de Andrade, Thomas Möllenhoff, and Marcus Rohrbach. Calibrated sampling-free uncertainty estimation in Bayesian deep learning, 2026. URL https://arxiv.org/abs/2606.16214.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, José Miguel Hernández-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust Bayesian neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

A Notation

Table 3: Main notation used throughout the paper.

Symbol	Meaning
$n = 0$	Prior-cost site, $\ell_0(\boldsymbol{\theta}) = \log h(\boldsymbol{\theta}) - \log p_0(\boldsymbol{\theta})$.
$n = 1, \dots, N$	Data-example index; $\ell_n(\boldsymbol{\theta})$ is its loss.
$\ell = 1, \dots, L$	Layer index (distinct from the loss symbol ℓ_n); v indexes an NN node.
$\bar{\ell}(\boldsymbol{\theta})$	Total cost $\sum_{n=0}^{N} \ell_n(\boldsymbol{\theta})$.
$\ell_n^{\mathrm{out}}(\mathbf{h}_L^n)$	Point-estimate output loss as a function of the terminal NN state.
$\mathcal{L}_n(\boldsymbol{\mu})$, $\mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}})$	Exact expected data loss and its projected approximation.
$\mathcal{L}_{\mathrm{data}}(\boldsymbol{\theta})$	Data objective $\sum_{n=1}^{N} \ell_n(\boldsymbol{\theta})$, excluding the prior cost.
$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$	EF variational posterior over NN parameters $\boldsymbol{\theta}$.
$h, \mathbf{T}, A$	EF carrier, sufficient statistic, and log partition function.
$\boldsymbol{\lambda}, \boldsymbol{\mu}$	Posterior natural and expectation parameters; $\boldsymbol{\mu} = \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})]$.
$\widehat{\boldsymbol{\lambda}}_{n,t}$	Loss-site natural parameter contribution, $-\nabla_{\boldsymbol{\mu}_t} \mathbb{E}_{q_t}[\ell_n]$.
$\mathcal{L}_0(\boldsymbol{\mu})$	Expected prior cost, $\mathbb{E}_q[\ell_0(\boldsymbol{\theta})]$.
$\mathbf{a}_\ell^n, \mathbf{h}_\ell^n$	Preactivation and postactivation variables for example n.
$\mathbf{z}_v^n, \mathbf{z}_{n,\mathrm{out}}$	Distributional states at NN nodes and at the output.
Ψ_v^n	Local distributional-state propagation map; Ψ_ℓ^a, Ψ_ℓ^h are affine and activation examples.
$\boldsymbol{\xi}_v^n$	Computational adjoint paired with $\mathbf{z}_v^n$, not a layer natural parameter.
$(\mathbf{m}, \boldsymbol{\Sigma})$	Gaussian mean–covariance coordinates; canonical expectation coordinates are $(\mathbf{m}, \boldsymbol{\Sigma} + \mathbf{m}\mathbf{m}^\mathsf{T})$.
$(\mathbf{g}, \mathbf{H})$	Gaussian adjoint gradient and Hessian channels, with pairing $\langle \mathbf{g}, d\mathbf{m} \rangle + \frac{1}{2}\langle \mathbf{H}, d\boldsymbol{\Sigma} \rangle_F$.
$\mathbf{G} = \mathbf{H}/2$	Hessian-gradient covector, used in intermediate appendix derivations.
$\mathbf{M}_\ell, \mathbf{V}_\ell$	Means and elementwise variances of factorized Gaussian weights or shared filters.
$\boldsymbol{d}_\ell, \mathbf{c}_\ell, \mathbf{s}_\ell$	Bias, its mean, and its elementwise variance.
$\boldsymbol{\eta}, \bar{\mathbf{J}}, \bar{\mathbf{H}}_i$	Gaussian-averaged activation mean, Jacobian, and Hessian coefficients.
$\mathbf{j}, \boldsymbol{\kappa}$	Elementwise averaged first- and second-derivative activation coefficients.

B Bayesian Learning Rule Details

This appendix provides derivations for Section 2.1. We first derive the BLR formulation employed in the main text and give its loss-site interpretation. We then show how a same-family prior is recovered as a special case, specialize the update to Gaussian posteriors using Bonnet's and Price's identities, and review the delta-method approximations that recover deterministic gradient and Newton-type updates.

B.1 BLR Update in Natural and Expectation Coordinates

Consider a regular minimal exponential family

$$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\left\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda})\right\}, \qquad \boldsymbol{\mu} = \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\mathbf{T}(\boldsymbol{\theta})] = \nabla_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda}).$$

The carrier h need not be normalized. We define $\mathcal{R}_h(q) := \mathbb{E}_q[\log(q/h)]$ whenever this expectation is well defined. Unlike a KL divergence between probability distributions, this quantity need not be nonnegative. For constant h, it reduces to the negative entropy up to an additive constant.

On the EF family,

$$\mathcal{R}_h(q_{\boldsymbol{\lambda}}) = \langle \boldsymbol{\lambda}, \boldsymbol{\mu} \rangle - A(\boldsymbol{\lambda}) = A^*(\boldsymbol{\mu}), \qquad \nabla_{\boldsymbol{\mu}} \mathcal{R}_h(q_{\boldsymbol{\lambda}}) = \boldsymbol{\lambda},$$

where A^* is the convex conjugate of A.

Given a prior $p_0(\boldsymbol{\theta})$, define $\ell_0(\boldsymbol{\theta}) = \log h(\boldsymbol{\theta}) - \log p_0(\boldsymbol{\theta})$. Then

$$\mathrm{KL}(q_{\boldsymbol{\lambda}} \| p_0) = \mathcal{R}_h(q_{\boldsymbol{\lambda}}) + \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\ell_0(\boldsymbol{\theta})].$$

For data losses ℓ_n, $n = 1, \ldots, N$, the variational objective becomes

$$J(\boldsymbol{\lambda}) = \mathcal{R}_h(q_{\boldsymbol{\lambda}}) + \sum_{n=0}^{N} \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\ell_n(\boldsymbol{\theta})].$$

For a regular exponential family, the Fisher information in natural coordinates is

$$\mathcal{I}(\boldsymbol{\lambda}) = \nabla^2_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda}) = \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}}.$$

Hence, for any differentiable scalar function $J(\boldsymbol{\lambda}) = J(\boldsymbol{\mu}(\boldsymbol{\lambda}))$,

$$\widetilde{\nabla}_{\boldsymbol{\lambda}} J := \mathcal{I}(\boldsymbol{\lambda})^{-1} \nabla_{\boldsymbol{\lambda}} J = \nabla_{\boldsymbol{\mu}} J.$$

This is the natural–expectation coordinate duality used by BLR [Amari, 1998, Khan and Rue, 2023].

It follows that

$$\nabla_{\boldsymbol{\mu}} J = \boldsymbol{\lambda} + \sum_{n=0}^{N} \nabla_{\boldsymbol{\mu}} \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\ell_n(\boldsymbol{\theta})].$$

Defining, as in Section 2.1,

$$\widehat{\boldsymbol{\lambda}}_{n,t} := -\nabla_{\boldsymbol{\mu}_t} \mathbb{E}_{q_t}[\ell_n(\boldsymbol{\theta})], \qquad n = 0, \ldots, N,$$

a natural gradient step of size $\rho_t > 0$ gives

$$\boldsymbol{\lambda}_{t+1} = (1 - \rho_t)\boldsymbol{\lambda}_t + \rho_t \sum_{n=0}^{N} \widehat{\boldsymbol{\lambda}}_{n,t},$$

which is Eq. (2). The index $n = 0$ denotes the prior site, while $n = 1, \ldots, N$ denote the loss-sites.

B.2 Loss-Site and Distributional Forms

Define the iteration-local linear sites

$$\widehat{\ell}_{n|t}(\boldsymbol{\theta}) := -\langle \widehat{\boldsymbol{\lambda}}_{n,t}, \mathbf{T}(\boldsymbol{\theta}) \rangle, \qquad n = 0, \ldots, N.$$

Their exponential factors satisfy

$$\exp\{-\rho_t \widehat{\ell}_{n|t}(\boldsymbol{\theta})\} = \exp\left\{ \langle \rho_t \widehat{\boldsymbol{\lambda}}_{n,t}, \mathbf{T}(\boldsymbol{\theta}) \rangle \right\},$$

so each site contributes $\rho_t \widehat{\boldsymbol{\lambda}}_{n,t}$ to the natural parameter. The BLR update can therefore be written in distributional form as

$$q_{t+1}(\boldsymbol{\theta}) \propto h(\boldsymbol{\theta})^{\rho_t} q_t(\boldsymbol{\theta})^{1-\rho_t} \prod_{n=0}^{N} \exp\{-\rho_t \widehat{\ell}_{n|t}(\boldsymbol{\theta})\}.$$

Indeed, $q_t^{1-\rho_t}$ contributes carrier $h^{1-\rho_t}$, while the factor h^{ρ_t} restores the carrier h of the updated exponential family distribution. The resulting natural parameter is

$$(1 - \rho_t)\boldsymbol{\lambda}_t + \rho_t \sum_{n=0}^{N} \widehat{\boldsymbol{\lambda}}_{n,t},$$

which reproduces Eq. (2).

B.3 Same-Family Prior as a Special Case

Suppose the prior belongs to the same exponential family,

$$p_0(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\left\{ \langle \boldsymbol{\lambda}_0, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda}_0) \right\}.$$

Then

$$\ell_0(\boldsymbol{\theta}) = -\langle \boldsymbol{\lambda}_0, \mathbf{T}(\boldsymbol{\theta}) \rangle + A(\boldsymbol{\lambda}_0),$$

and consequently

$$\mathbb{E}_q[\ell_0(\boldsymbol{\theta})] = -\langle \boldsymbol{\lambda}_0, \boldsymbol{\mu} \rangle + A(\boldsymbol{\lambda}_0), \qquad \widehat{\boldsymbol{\lambda}}_0 = \boldsymbol{\lambda}_0.$$

The general BLR update therefore reduces to

$$\boldsymbol{\lambda}_{t+1} = (1-\rho_t)\boldsymbol{\lambda}_t + \rho_t \boldsymbol{\lambda}_0 + \rho_t \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_{n,t}.$$

Likewise, the general distributional update reduces to

$$q_{t+1}(\boldsymbol{\theta}) \propto q_t(\boldsymbol{\theta})^{1-\rho_t} p_0(\boldsymbol{\theta})^{\rho_t} \prod_{n=1}^{N} \exp\{-\rho_t \widehat{\ell}_{n|t}(\boldsymbol{\theta})\}.$$

Thus the familiar same-family BLR equations are recovered directly from the more general prior-as-cost formulation.

B.4 Gaussian BLR via Bonnet's and Price's Identities

For the Gaussian exponential family, the carrier is constant in $\boldsymbol{\theta}$, so the prior cost reduces, up to an additive constant, to

$$\ell_0(\boldsymbol{\theta}) = -\log p_0(\boldsymbol{\theta}).$$

Define

$$\bar{\ell}(\boldsymbol{\theta}) := \ell_0(\boldsymbol{\theta}) + \sum_{n=1}^{N} \ell_n(\boldsymbol{\theta}).$$

Consider the Gaussian posterior used in Section 2.1,

$$q_t(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}_t, \mathbf{S}_t^{-1}), \qquad \mathbf{C}_t := \mathbf{S}_t^{-1}.$$

Its natural parameters are

$$\boldsymbol{\lambda}_t^{(1)} = \mathbf{S}_t \mathbf{m}_t, \qquad \boldsymbol{\lambda}_t^{(2)} = -\frac{1}{2}\mathbf{S}_t,$$

and its expectation parameters are

$$\boldsymbol{\mu}_t^{(1)} = \mathbf{m}_t, \qquad \boldsymbol{\mu}_t^{(2)} = \mathbf{C}_t + \mathbf{m}_t \mathbf{m}_t^{\mathsf{T}}.$$

For a smooth scalar function f, define

$$J(\mathbf{m}, \mathbf{C}) := \mathbb{E}_{\mathcal{N}(\mathbf{m},\mathbf{C})}[f(\boldsymbol{\theta})].$$

Because $\boldsymbol{\mu}^{(1)} = \mathbf{m}$ and $\boldsymbol{\mu}^{(2)} = \mathbf{C} + \mathbf{m}\mathbf{m}^{\mathsf{T}}$, the chain rule gives

$$\nabla_{\boldsymbol{\mu}^{(2)}} J = \nabla_{\mathbf{C}} J, \qquad \nabla_{\boldsymbol{\mu}^{(1)}} J = \nabla_{\mathbf{m}} J - 2(\nabla_{\mathbf{C}} J)\mathbf{m}.$$

Bonnet's and Price's identities [Price, 1958, Rezende et al., 2014] give

$$\nabla_{\mathbf{m}} J = \mathbb{E}_{\mathcal{N}(\mathbf{m},\mathbf{C})}[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})], \qquad \nabla_{\mathbf{C}} J = \frac{1}{2}\mathbb{E}_{\mathcal{N}(\mathbf{m},\mathbf{C})}[\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta})].$$

Combining these relations,

$$\nabla_{\boldsymbol{\mu}^{(1)}} J = \mathbb{E}[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})] - \mathbb{E}[\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta})]\mathbf{m}, \qquad \nabla_{\boldsymbol{\mu}^{(2)}} J = \frac{1}{2}\mathbb{E}[\nabla^2_{\boldsymbol{\theta}} f(\boldsymbol{\theta})].$$

Applying these identities to the BLR update with $f = \bar{\ell}$ gives

$$\mathbf{S}_{t+1} = (1-\rho_t)\mathbf{S}_t + \rho_t \mathbb{E}_{q_t}[\nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})].$$

For the first natural parameter,

$$\mathbf{S}_{t+1}\mathbf{m}_{t+1} = (1-\rho_t)\mathbf{S}_t\mathbf{m}_t - \rho_t \left(\mathbb{E}_{q_t}[\nabla_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})] - \mathbb{E}_{q_t}[\nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})]\mathbf{m}_t\right).$$

Using the precision update above, the terms multiplying $\mathbf{m}_t$ combine to $\mathbf{S}_{t+1}\mathbf{m}_t$, yielding

$$\mathbf{m}_{t+1} = \mathbf{m}_t - \rho_t \mathbf{S}_{t+1}^{-1} \mathbb{E}_{q_t}[\nabla_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})].$$

Together these are exactly Eq. (3).

B.5 Deterministic Method Recovery by Delta-Method Approximation

The deterministic optimization special cases of BLR arise by replacing posterior expectations with delta-method approximations at the posterior mean [Khan and Rue, 2023]. This is an approximation to the expectations; it does not require taking a singular zero-variance limit of the variational family.

For the expected gradient,

$$\mathbb{E}_{q_t}[\nabla_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})] \approx \nabla_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t).$$

If BLR is restricted to a fixed-covariance Gaussian location family, for example $q_t = \mathcal{N}(\mathbf{m}_t, \mathbf{I})$, the mean update reduces to

$$\mathbf{m}_{t+1} = \mathbf{m}_t - \rho_t \nabla_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t),$$

which is gradient descent with $\mathbf{m}_t$ as the deterministic iterate.

For the full Gaussian family, additionally applying

$$\mathbb{E}_{q_t}[\nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\boldsymbol{\theta})] \approx \nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t)$$

gives

$$\mathbf{S}_{t+1} \approx (1 - \rho_t)\mathbf{S}_t + \rho_t \nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t),$$

and

$$\mathbf{m}_{t+1} \approx \mathbf{m}_t - \rho_t \mathbf{S}_{t+1}^{-1} \nabla_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t).$$

With $\rho_t = 1$, this becomes Newton's method,

$$\mathbf{m}_{t+1} = \mathbf{m}_t - \left[\nabla^2_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t)\right]^{-1} \nabla_{\boldsymbol{\theta}} \bar{\ell}(\mathbf{m}_t).$$

For $0 < \rho_t < 1$, the precision is an exponentially smoothed curvature estimate, leading to an online damped Newton-type update.

C Lagrangian Derivations of Statistical and Computational Duality

Section 2.2 compares two constrained Lagrangian formulations. We derive both results in detail here.

C.1 Exponential Families from Statistical Moment Consistency

Let $h(\boldsymbol{\theta})$ be a nonnegative EF carrier with respect to the reference measure. We restrict q to the support of h and assume the following integrals are well defined. For a moment vector $\boldsymbol{\mu}$, consider

$$\min_q \mathcal{R}_h(q) \qquad \text{subject to} \qquad \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] = \boldsymbol{\mu}, \qquad \int q(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} = 1,$$

where

$$\mathcal{R}_h(q) := \int q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{h(\boldsymbol{\theta})}\, \mathrm{d}\boldsymbol{\theta}.$$

When $h = 1$, this objective reduces to the negative entropy $-\mathcal{H}(q)$. When h is a normalized density, $\mathcal{R}_h(q)$ is the usual KL divergence. An unnormalized carrier is also allowed; then $\mathcal{R}_h(q)$ need not be nonnegative.

Using the same sign convention as in Section 2.2, the Lagrangian is

$$\mathfrak{L}_{\mathrm{EF}}(q, \boldsymbol{\lambda}, \alpha; \boldsymbol{\mu}) = \mathcal{R}_h(q) + \langle \boldsymbol{\lambda}, \boldsymbol{\mu} - \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] \rangle + \alpha \left(1 - \int q(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}\right).$$

Here $\boldsymbol{\lambda}$ is the multiplier associated with statistical moment consistency, while α enforces normalization.

Taking the first variation with respect to q gives

$$\log \frac{q(\boldsymbol{\theta})}{h(\boldsymbol{\theta})} + 1 - \langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - \alpha = 0.$$

Therefore,

$$q(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\left\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle + \alpha - 1\right\}.$$

Normalization introduces the log-partition function

$$A(\boldsymbol{\lambda}) := \log \int h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta})\rangle\} \, \mathrm{d}\boldsymbol{\theta}, \qquad \alpha = 1 - A(\boldsymbol{\lambda}),$$

so

$$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\left\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta})\rangle - A(\boldsymbol{\lambda})\right\}.$$

Stationarity with respect to $\boldsymbol{\lambda}$ recovers the prescribed moment constraint,

$$\boldsymbol{\mu} = \mathbb{E}_{q_{\boldsymbol{\lambda}}}[\mathbf{T}(\boldsymbol{\theta})] = \nabla_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda}).$$

Under standard regularity conditions,

$$\nabla^2_{\boldsymbol{\lambda}} A(\boldsymbol{\lambda}) = \mathrm{Cov}_{q_{\boldsymbol{\lambda}}}[\mathbf{T}(\boldsymbol{\theta})] = \mathcal{I}(\boldsymbol{\lambda}).$$

Thus the natural parameter $\boldsymbol{\lambda}$ has two equivalent interpretations relevant to this paper. It is the coordinate dual to the expectation parameter $\boldsymbol{\mu}$, and it is the Lagrange multiplier enforcing statistical moment consistency.

C.2 Deterministic NN Computational Duality

We now derive the Lagrangian view of deterministic backpropagation used in Section 2.2. Consider a dataset indexed by $n = 1, \ldots, N$, a network with parameters $\boldsymbol{\vartheta} = (\boldsymbol{\vartheta}_1, \ldots, \boldsymbol{\vartheta}_L)$, layer maps F_ℓ, and inputs $\mathbf{h}_0^n = \mathbf{x}_n$. The forward constraints are

$$\mathbf{h}_\ell^n = F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell), \qquad \ell = 1, \ldots, L.$$

Introducing multipliers $\mathbf{b}_\ell^n$ for these constraints gives

$$\mathfrak{L}_{\mathrm{NN}}(\boldsymbol{\vartheta}, \mathbf{h}, \mathbf{b}) = \sum_{n=1}^{N} \left[\ell_n^{\mathrm{out}}(\mathbf{h}_L^n) + \sum_{\ell=1}^{L} \langle \mathbf{b}_\ell^n, F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell) - \mathbf{h}_\ell^n \rangle \right].$$

The stationarity conditions recover the forward and backward computations [LeCun, 1988].

Forward pass. Stationarity with respect to $\mathbf{b}_\ell^n$ gives

$$F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell) - \mathbf{h}_\ell^n = 0, \qquad \text{hence} \qquad \mathbf{h}_\ell^n = F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell).$$

Backward pass. For the terminal state $\mathbf{h}_L^n$,

$$0 = \nabla_{\mathbf{h}_L^n} \mathfrak{L}_{\mathrm{NN}} = \nabla_{\mathbf{h}_L^n} \ell_n^{\mathrm{out}}(\mathbf{h}_L^n) - \mathbf{b}_L^n,$$

so

$$\mathbf{b}_L^n = \nabla_{\mathbf{h}_L^n} \ell_n^{\mathrm{out}}(\mathbf{h}_L^n).$$

For an intermediate state $\mathbf{h}_{\ell-1}^n$, with $\ell = L, \ldots, 2$, the relevant terms are

$$-\langle \mathbf{b}_{\ell-1}^n, \mathbf{h}_{\ell-1}^n \rangle + \langle \mathbf{b}_\ell^n, F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell) \rangle.$$

Stationarity gives

$$0 = -\mathbf{b}_{\ell-1}^n + \left(D_{\mathbf{h}_{\ell-1}} F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell)\right)^* \mathbf{b}_\ell^n,$$

and hence

$$\mathbf{b}_{\ell-1}^n = \left(D_{\mathbf{h}_{\ell-1}} F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell)\right)^* \mathbf{b}_\ell^n.$$

For a feed-forward chain this is the usual layerwise backpropagation recursion. More generally, for a computational node with multiple children, its adjoint is the sum of the pullbacks from all child nodes.

Parameter gradient. Differentiating with respect to $\boldsymbol{\vartheta}_\ell$ gives

$$\nabla_{\boldsymbol{\vartheta}_\ell} \mathfrak{L}_{\mathrm{NN}} = \sum_{n=1}^{N} \left(D_{\boldsymbol{\vartheta}_\ell} F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell)\right)^* \mathbf{b}_\ell^n.$$

After imposing the forward and adjoint equations, this is the parameter gradient of the constrained network loss,

$$\nabla_{\boldsymbol{\vartheta}_\ell} \sum_{n=1}^{N} \ell_n^{\mathrm{out}}(\mathbf{h}_L^n) = \sum_{n=1}^{N} \left(D_{\boldsymbol{\vartheta}_\ell} F_\ell(\mathbf{h}_{\ell-1}^n; \boldsymbol{\vartheta}_\ell)\right)^* \mathbf{b}_\ell^n.$$

For the single-example notation used in Eq. (4), the superscript n and the sum over examples are omitted.

Here $(DF)^*$ denotes the adjoint of the derivative map DF with respect to the chosen inner products:

$$\langle \mathbf{b}, DF\, \mathbf{u} \rangle = \langle (DF)^* \mathbf{b}, \mathbf{u} \rangle.$$

D Derivations for the BLR Lagrangian Formulation

This appendix gives the stationarity calculations underlying Section 3.2.

Recall the lifted Lagrangian from Eq. (6),

$$\mathfrak{L}_{\mathrm{BLR}}(q, \boldsymbol{\mu}, \boldsymbol{\lambda}, \mathbf{z}, \boldsymbol{\xi}) = \sum_{n=1}^{N} \mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}}) + \mathcal{L}_0(\boldsymbol{\mu}) + \mathcal{R}_h(q) \\ + \langle \boldsymbol{\lambda}, \boldsymbol{\mu} - \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] \rangle + \sum_{n,v} \langle \boldsymbol{\xi}_v^n, \Psi_v^n(\mathbf{z}_{\mathrm{pa}(v)}^n; \boldsymbol{\mu}) - \mathbf{z}_v^n \rangle.$$

For the derivation below, a scalar multiplier for the normalization constraint $\int q(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta} = 1$ is included when taking the variation with respect to q.

D.1 Stationarity of the Lifted Problem

Statistical consistency and posterior form. Stationarity with respect to $\boldsymbol{\lambda}$ gives

$$\boldsymbol{\mu} = \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})].$$

Including a normalization multiplier α, the first variation with respect to q is

$$\log \frac{q(\boldsymbol{\theta})}{h(\boldsymbol{\theta})} + 1 - \langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - \alpha = 0.$$

Hence

$$q_{\boldsymbol{\lambda}}(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda})\},$$

where the normalization constant has been absorbed into the log-partition function $A(\boldsymbol{\lambda})$. Thus the Lagrange multiplier of statistical moment consistency is exactly the posterior natural parameter.

Distributional forward pass. Stationarity with respect to each computational multiplier $\boldsymbol{\xi}_v^n$ gives

$$\mathbf{z}_v^n = \Psi_v^n(\mathbf{z}_{\mathrm{pa}(v)}^n; \boldsymbol{\mu}),$$

which is the distributional-state forward computation in Eq. (5).

Distributional backward pass. Consider a distributional state $\mathbf{z}_u^n$. It appears with coefficient $-\boldsymbol{\xi}_u^n$ in its own residual and through every child map Ψ_v^n with $v \in \mathrm{ch}(u)$. If a local objective is attached to node u, denote its derivative by $\mathbf{c}_u^n$. Stationarity with respect to $\mathbf{z}_u^n$ therefore gives

$$\boldsymbol{\xi}_u^n = \mathbf{c}_u^n + \sum_{v \in \mathrm{ch}(u)} \left(D_{\mathbf{z}_u^n} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n,$$

which is Eq. (7). With only a terminal loss,

$$\mathbf{c}_{n,\mathrm{out}} = \nabla_{\mathbf{z}_{n,\mathrm{out}}} \mathcal{L}_n^{\mathrm{proj}}(\mathbf{z}_{n,\mathrm{out}}), \qquad \mathbf{c}_u^n = 0 \quad \text{for non-output nodes.}$$

Thus the computational multipliers are precisely the adjoints of the distributional forward computation.

The dependence of the local projection maps on the posterior expectation coordinate $\boldsymbol{\mu}$ contributes

$$\sum_v \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n$$

for example n. Defining

$$\widehat{\boldsymbol{\lambda}}_n := -\sum_v \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n,$$

recovers Eq. (8). This is the negative expectation-coordinate gradient of the projected expected loss. By EF duality it is the loss-site natural parameter contribution for that projected objective, approximating the exact BLR site unless the projection and its derivative are exact. This is the quantity defined in (8).

Stationarity in the shared posterior moments. The terms in the Lagrangian that depend explicitly on $\boldsymbol{\mu}$ are the prior contribution $\mathcal{L}_0(\boldsymbol{\mu})$, the statistical-consistency term, and the parameterized distributional-state maps. Therefore

$$\mathbf{0} = \nabla_{\boldsymbol{\mu}} \mathcal{L}_0(\boldsymbol{\mu}) + \boldsymbol{\lambda} + \sum_{n=1}^{N} \sum_{v} \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^n.$$

With

$$\widehat{\boldsymbol{\lambda}}_0 := -\nabla_{\boldsymbol{\mu}} \mathcal{L}_0(\boldsymbol{\mu}),$$

and the definition of $\widehat{\boldsymbol{\lambda}}_n$ above, this becomes

$$\mathbf{0} = \boldsymbol{\lambda} - \widehat{\boldsymbol{\lambda}}_0 - \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n.$$

Hence, at a stationary point,

$$\boldsymbol{\lambda}^\star = \widehat{\boldsymbol{\lambda}}_0^\star + \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n^\star = \widehat{\boldsymbol{\lambda}}_0^\star - \sum_{n=1}^{N} \sum_{v} \left(D_{\boldsymbol{\mu}} \Psi_v^n\right)^* \boldsymbol{\xi}_v^{n,\star},$$

which is the coupling relation in Eq. (9).

For a prior in the same exponential family,

$$p_0(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\{\langle \boldsymbol{\lambda}_0, \mathbf{T}(\boldsymbol{\theta}) \rangle - A(\boldsymbol{\lambda}_0)\}.$$

Since

$$\ell_0(\boldsymbol{\theta}) = \log h(\boldsymbol{\theta}) - \log p_0(\boldsymbol{\theta}) = -\langle \boldsymbol{\lambda}_0, \mathbf{T}(\boldsymbol{\theta}) \rangle + A(\boldsymbol{\lambda}_0),$$

we have

$$\mathcal{L}_0(\boldsymbol{\mu}) = -\langle \boldsymbol{\lambda}_0, \boldsymbol{\mu} \rangle + A(\boldsymbol{\lambda}_0), \qquad \widehat{\boldsymbol{\lambda}}_0 = \boldsymbol{\lambda}_0.$$

The general stationary relation therefore reduces to

$$\boldsymbol{\lambda}^\star - \boldsymbol{\lambda}_0 = \sum_{n=1}^{N} \widehat{\boldsymbol{\lambda}}_n^\star.$$

E Gaussian Case Derivations

E.1 Coordinate Conversion

For a Gaussian distribution, the canonical EF expectation coordinates are $(\mathbf{m}, \boldsymbol{\Omega})$, with

$$\boldsymbol{\Omega} = \boldsymbol{\Sigma} + \mathbf{m}\mathbf{m}^\mathsf{T}.$$

The main text instead uses the equivalent mean–covariance coordinates $(\mathbf{m}, \boldsymbol{\Sigma})$, because they make the Gaussian BLR gradient and curvature quantities explicit. Let J be a differentiable scalar functional. Write its differential in canonical expectation coordinates as

$$dJ = \langle \mathbf{u}, d\mathbf{m} \rangle + \langle \mathbf{U}, d\boldsymbol{\Omega} \rangle_F,$$

and in mean–covariance coordinates as

$$dJ = \langle \mathbf{g}, d\mathbf{m} \rangle + \frac{1}{2} \langle \mathbf{H}, d\boldsymbol{\Sigma} \rangle_F.$$

The factor $1/2$ defines the curvature channel $\mathbf{H} = 2\nabla_{\boldsymbol{\Sigma}} J$; equivalently, the covariance-gradient covector is $\mathbf{G} := \mathbf{H}/2$. Since

$$d\boldsymbol{\Omega} = d\boldsymbol{\Sigma} + d\mathbf{m}\, \mathbf{m}^\mathsf{T} + \mathbf{m}\, d\mathbf{m}^\mathsf{T},$$

we obtain, for symmetric $\mathbf{U}$,

$$\mathbf{g} = \mathbf{u} + 2\mathbf{U}\mathbf{m}, \qquad \mathbf{H} = 2\mathbf{U}.$$

Thus

$$\mathbf{u} = \mathbf{g} - \mathbf{H}\mathbf{m}, \qquad \mathbf{U} = \frac{1}{2}\mathbf{H}.$$

Consequently, the corresponding BLR loss-site natural parameter is

$$-\nabla_{\boldsymbol{\mu}} J = \left(-\mathbf{g} + \mathbf{H}\mathbf{m}, -\frac{1}{2}\mathbf{H}\right).$$

For diagonal Gaussians, replace the covariance and matrix Hessian by the variance and diagonal-curvature vectors, so that the product $\mathbf{H}\mathbf{m}$ becomes elementwise multiplication. This conversion applies to the Gaussian parameter posterior as well as to Gaussian layer-state adjoints, with the latter remaining computational adjoints rather than separate posterior natural parameters.

E.2 Moments of the Centered Quadratic Surrogate

We suppress the example and layer indices (n, ℓ). Let $\boldsymbol{\delta} = \mathbf{a} - \mathbf{m}_a \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_a)$, and define

$$Q_i := \boldsymbol{\delta}^\mathsf{T} \bar{\mathbf{H}}_i \boldsymbol{\delta} - \operatorname{tr}(\bar{\mathbf{H}}_i \boldsymbol{\Sigma}_a).$$

The centered quadratic activation surrogate is

$$\widehat{\sigma}_i(\mathbf{a}) := \boldsymbol{\eta}_i + (\bar{\mathbf{J}}\boldsymbol{\delta})_i + \frac{1}{2} Q_i. \tag{16}$$

Its mean satisfies

$$\mathbb{E}[\widehat{\sigma}_i(\mathbf{a})] = \boldsymbol{\eta}_i,$$

because $\mathbb{E}[Q_i] = 0$. The covariance of the linear terms is

$$\operatorname{Cov}(\bar{\mathbf{J}}\boldsymbol{\delta}) = \bar{\mathbf{J}} \boldsymbol{\Sigma}_a \bar{\mathbf{J}}^\mathsf{T}.$$

The cross-covariance between the centered linear Gaussian term and the centered quadratic term is zero. Moreover, we have

$$\operatorname{Cov}(Q_i, Q_j) = 2 \operatorname{tr}(\bar{\mathbf{H}}_i \boldsymbol{\Sigma}_a \bar{\mathbf{H}}_j \boldsymbol{\Sigma}_a).$$

Thus the covariance contribution of the quadratic terms $Q_i/2$ is

$$\frac{1}{2} \bar{K}_\sigma[\boldsymbol{\Sigma}_a, \boldsymbol{\Sigma}_a], \qquad [\bar{K}_\sigma[\boldsymbol{\Sigma}, \boldsymbol{\Sigma}]]_{ij} = \operatorname{tr}(\bar{\mathbf{H}}_i \boldsymbol{\Sigma} \bar{\mathbf{H}}_j \boldsymbol{\Sigma}).$$

Thus the second-order (SO) Gaussian projection is

$$\Psi^{\mathrm{loc}}_{\mathrm{SO,t}}(\mathbf{m}_a, \boldsymbol{\Sigma}_a) := \left(\boldsymbol{\eta}(\mathbf{m}_a, \boldsymbol{\Sigma}_a), \bar{\mathbf{J}}_t \boldsymbol{\Sigma}_a \bar{\mathbf{J}}_t^\mathsf{T} + \frac{1}{2} \bar{K}_{\sigma,t}[\boldsymbol{\Sigma}_a, \boldsymbol{\Sigma}_a]\right). \tag{17}$$

Dropping the quadratic covariance correction recovers the first-order projection (11). Here the averaged coefficients are evaluated at the current Gaussian state and held fixed only in the covariance approximation, as in the main text.

E.3 Moment Propagation and Backpropagation Derivations

We again suppress (n, ℓ). As in Section 4.2, the averaged coefficients are evaluated at the current Gaussian state and held fixed when they appear in the covariance approximation. The mean map $\boldsymbol{\eta}(\mathbf{m}_a, \boldsymbol{\Sigma}_a)$ remains differentiable; this distinction is necessary for the averaged first- and second-derivative terms in the adjoint.

For the first-order activation map (11),

$$\mathbf{m}_h = \boldsymbol{\eta}, \qquad \boldsymbol{\Sigma}_h = \bar{\mathbf{J}} \boldsymbol{\Sigma}_a \bar{\mathbf{J}}^\mathsf{T}.$$

Using $D_{\mathbf{m}_a} \boldsymbol{\eta} = \bar{\mathbf{J}}$ and $2 D_{\boldsymbol{\Sigma}_a} \boldsymbol{\eta}_i = \bar{\mathbf{H}}_i$, its differential is

$$d\mathbf{m}_h = \bar{\mathbf{J}}\, d\mathbf{m}_a + \frac{1}{2}\left[\operatorname{tr}(\bar{\mathbf{H}}_i\, d\boldsymbol{\Sigma}_a)\right]_i, \qquad d\boldsymbol{\Sigma}_h = \bar{\mathbf{J}}\, d\boldsymbol{\Sigma}_a\, \bar{\mathbf{J}}^\mathsf{T}.$$

Pairing with the postactivation gradient pair $(\mathbf{g}_h, \mathbf{G}_h)$ gives

$$\mathbf{g}_a = \bar{\mathbf{J}}^\mathsf{T} \mathbf{g}_h, \qquad \mathbf{G}_a = \bar{\mathbf{J}}^\mathsf{T} \mathbf{G}_h \bar{\mathbf{J}} + \frac{1}{2} \sum_i (\mathbf{g}_h)_i \bar{\mathbf{H}}_i,$$

which is (12).

For the second-order activation map (17), the additional covariance term is

$$\frac{1}{2}\bar{K}_\sigma[\mathbf{\Sigma}_a, \mathbf{\Sigma}_a].$$

Define its adjoint contribution $\mathcal{K}_\sigma^*$ by

$$\langle \mathcal{K}_\sigma^*[\mathbf{G}_h], d\mathbf{\Sigma}_a \rangle_F = \frac{1}{2}\sum_{i,j}(\mathbf{G}_h)_{ij}\, d\,\mathrm{tr}(\bar{\mathbf{H}}_i \mathbf{\Sigma}_a \bar{\mathbf{H}}_j \mathbf{\Sigma}_a)[d\mathbf{\Sigma}_a].$$

Equivalently, for symmetric $\mathbf{G}_h$ and symmetric $\bar{\mathbf{H}}_i$,

$$\mathcal{K}_\sigma^*[\mathbf{G}_h] = \frac{1}{2}\sum_{i,j}(\mathbf{G}_h)_{ij}\left(\bar{\mathbf{H}}_j \mathbf{\Sigma}_a \bar{\mathbf{H}}_i + \bar{\mathbf{H}}_i \mathbf{\Sigma}_a \bar{\mathbf{H}}_j\right).$$

Adding this contribution gives

$$\mathbf{G}_a = \bar{\mathbf{J}}^\mathsf{T}\mathbf{G}_h\bar{\mathbf{J}} + \frac{1}{2}\sum_i (\mathbf{g}_h)_i \bar{\mathbf{H}}_i + \mathcal{K}_\sigma^*[\mathbf{G}_h],$$

which is (18).

The covariance-gradient channel is related to the Hessian/curvature channel by $\mathbf{H} = 2\mathbf{G}$. Therefore the first-order rule is equivalently

$$\mathbf{H}_a = \bar{\mathbf{J}}^\mathsf{T}\mathbf{H}_h\bar{\mathbf{J}} + \sum_i (\mathbf{g}_h)_i \bar{\mathbf{H}}_i.$$

For the second-order rule,

$$\mathbf{H}_a = \bar{\mathbf{J}}^\mathsf{T}\mathbf{H}_h\bar{\mathbf{J}} + \sum_i (\mathbf{g}_h)_i \bar{\mathbf{H}}_i + 2\mathcal{K}_\sigma^*[\mathbf{G}_h]. \tag{18}$$

Since $\mathcal{K}_\sigma^*$ is linear in its argument, the last term may also be written as $\mathcal{K}_\sigma^*[\mathbf{H}_h]$.

E.4 Diagonal Covariance Equations

We suppress the example and layer indices (n, ℓ). Let $\mathbf{v} = \mathrm{diag}(\mathbf{\Sigma})$, and let $\boldsymbol{\gamma}$ denote the diagonal covariance-gradient channel paired with perturbations $d\mathbf{v}$. The corresponding diagonal Hessian/curvature channel is

$$\mathbf{H}^{\mathrm{diag}} := 2\boldsymbol{\gamma}.$$

For an elementwise activation,

$$\bar{\mathbf{J}} = \mathrm{Diag}(\mathbf{j}), \qquad \bar{\mathbf{H}}_i = \boldsymbol{\kappa}_i \mathbf{e}_i \mathbf{e}_i^\mathsf{T},$$

where

$$\mathbf{j} = \partial_{\mathbf{m}_a}\boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a), \qquad \boldsymbol{\kappa} = 2\partial_{\mathbf{v}_a}\boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a)$$

elementwise. The factor 2 in $\boldsymbol{\kappa}$ follows from Price's identity.

First-order case. The constant-coefficient first-order diagonal closure is

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a), \qquad \mathbf{v}_h = \mathbf{j}^{\circ 2} \odot \mathbf{v}_a.$$

The corresponding backward gradient equations are

$$\mathbf{g}_a = \mathbf{j} \odot \mathbf{g}_h, \qquad \boldsymbol{\gamma}_a = \mathbf{j}^{\circ 2} \odot \boldsymbol{\gamma}_h + \frac{1}{2}\boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Equivalently, in the Hessian/curvature channel $\mathbf{H}^{\mathrm{diag}} = 2\boldsymbol{\gamma}$,

$$\mathbf{H}_a^{\mathrm{diag}} = \mathbf{j}^{\circ 2} \odot \mathbf{H}_h^{\mathrm{diag}} + \boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Second-order case. The constant-coefficient second-order diagonal closure keeps the quadratic covariance contribution:

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a), \qquad \mathbf{v}_h = \mathbf{j}^{\circ 2} \odot \mathbf{v}_a + \frac{1}{2}\boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a^{\circ 2}.$$

The corresponding backward gradient equations are

$$\mathbf{g}_a = \mathbf{j} \odot \mathbf{g}_h, \qquad \boldsymbol{\gamma}_a = \left(\mathbf{j}^{\circ 2} + \boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a\right) \odot \boldsymbol{\gamma}_h + \frac{1}{2}\boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Equivalently, in the Hessian/curvature channel $\mathbf{H}^{\mathrm{diag}} = 2\boldsymbol{\gamma}$,

$$\mathbf{H}_a^{\mathrm{diag}} = \left(\mathbf{j}^{\circ 2} + \boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a\right) \odot \mathbf{H}_h^{\mathrm{diag}} + \boldsymbol{\kappa} \odot \mathbf{g}_h.$$

The additional factor $\boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a$ comes from differentiating the second-order covariance term $\frac{1}{2}\boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a^{\circ 2}$ with constant $\boldsymbol{\kappa}$.

E.5 Delta-Method Recovery of Point Estimate BP and HBP

Proposition 1 (Point estimate forward, BP, and HBP recovery)**.** *Let $q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ be a Gaussian parameter posterior and represent the NN layer states by Gaussian means and covariances $\mathbf{z}_v = (\mathbf{m}_v, \boldsymbol{\Sigma}_v)$. Assume that the relevant activation expectations and derivatives exist, and use the first-order projection* (11) *with the coefficient convention of Section 4.2. Apply the delta-method approximation to the Gaussian-averaged activation values, Jacobians, and Hessians, together with corresponding pointwise rules for the other NN operations and the output loss. Then the mean forward computation recovers the point estimate NN evaluated at the posterior mean, and the gradient and curvature adjoints recover BP and HBP along that same trajectory. This recovery does not require setting the Gaussian covariances to zero.*

Proof. For a preactivation $\mathbf{a} \sim \mathcal{N}(\mathbf{m}_a, \boldsymbol{\Sigma}_a)$, the Gaussian mean map obeys

$$D_{\mathbf{m}_a}\boldsymbol{\eta} = \bar{\mathbf{J}}, \qquad 2D_{\boldsymbol{\Sigma}_a}\boldsymbol{\eta}_i = \bar{\mathbf{H}}_i.$$

Delta-method localization replaces the three corresponding quantities by their values at the preactivation mean,

$$\boldsymbol{\eta} \approx \sigma(\mathbf{m}_a), \qquad \bar{\mathbf{J}} \approx D\sigma(\mathbf{m}_a), \qquad \bar{\mathbf{H}}_i \approx \nabla^2\sigma_i(\mathbf{m}_a).$$

The mean channel therefore reproduces the point estimate activation. The backward formulas in (12), using the pairing with curvature channel $\mathbf{H} = 2\mathbf{G}$, become

$$\mathbf{g}_a = \mathbf{J}^{\mathsf{T}}\mathbf{g}_h, \qquad \mathbf{H}_a = \mathbf{J}^{\mathsf{T}}\mathbf{H}_h\mathbf{J} + \sum_i (\mathbf{g}_h)_i \mathbf{H}_i,$$

where $\mathbf{J} = D\sigma(\mathbf{m}_a)$ and $\mathbf{H}_i = \nabla^2\sigma_i(\mathbf{m}_a)$. These are precisely the BP and HBP recursions for the activation. The standard chain rule and its second-order counterpart give the analogous recursions for affine operations and other differentiable NN modules.

The localization treats the activation mean and its local derivative coefficients together, rather than differentiating the simplified expression $\sigma(\mathbf{m}_a)$ as a globally covariance-independent function. To make this explicit, at the current state $(\mathbf{m}_{a,t}, \boldsymbol{\Sigma}_{a,t})$ one may consider the local surrogate

$$\boldsymbol{\eta}_{i,t}^{\delta}(\mathbf{m}_a, \boldsymbol{\Sigma}_a) := \sigma_i(\mathbf{m}_a) + \frac{1}{2}\langle \nabla^2\sigma_i(\mathbf{m}_{a,t}), \boldsymbol{\Sigma}_a - \boldsymbol{\Sigma}_{a,t}\rangle_F.$$

At the current state, its value, mean derivative, and twice its covariance derivative are exactly the three delta-localized quantities above. Thus the covariance sensitivity that generates HBP remains available without collapsing the Gaussian state.

Finally, for the Gaussian parameter posterior, Bonnet's and Price's identities give

$$\nabla_{\mathbf{m}}\mathbb{E}_q[\bar{\ell}(\boldsymbol{\theta})] = \mathbb{E}_q[\nabla_{\boldsymbol{\theta}}\bar{\ell}(\boldsymbol{\theta})], \qquad 2\nabla_{\boldsymbol{\Sigma}}\mathbb{E}_q[\bar{\ell}(\boldsymbol{\theta})] = \mathbb{E}_q[\nabla^2_{\boldsymbol{\theta}}\bar{\ell}(\boldsymbol{\theta})].$$

Delta localization of these expectations yields the point estimate gradient and Hessian of the total cost. The distributional forward and adjoint computations recover the corresponding NN algorithms for evaluating them, under the stated pointwise rules for each module. □

E.6 Scalar Averaged Activation Coefficients

For a scalar preactivation $A \sim \mathcal{N}(\mu, v)$, $v > 0$, define

$$\eta_\sigma(\mu, v) := \mathbb{E}[\sigma(A)], \qquad j_\sigma(\mu, v) := \partial_\mu \eta_\sigma(\mu, v), \qquad \kappa_\sigma(\mu, v) := 2\partial_v \eta_\sigma(\mu, v).$$

For smooth activations, Bonnet's and Price's identities give

$$j_\sigma(\mu, v) = \mathbb{E}[\sigma'(A)], \qquad \kappa_\sigma(\mu, v) = \mathbb{E}[\sigma''(A)].$$

Let

$$\alpha = \frac{\mu}{\sqrt{v}}, \qquad z = \frac{\mu}{\sqrt{1+v}},$$

and let φ, Φ denote the standard normal pdf and cdf, respectively. Useful coefficients are listed in Table 4.

Table 4: Gaussian-averaged scalar activation coefficients. For ReLU, κ_σ is interpreted as the Gaussian average of the distributional second derivative.

Activation	$\eta_\sigma(\mu, v)$	$j_\sigma(\mu, v)$	$\kappa_\sigma(\mu, v)$
ReLU	$\mu\Phi(\alpha) + \sqrt{v}\,\varphi(\alpha)$	$\Phi(\alpha)$	$\varphi(\alpha)/\sqrt{v}$
Probit $\Phi(x)$	$\Phi(z)$	$\varphi(z)/\sqrt{1+v}$	$-\mu\varphi(z)/(1+v)^{3/2}$
GeLU $x\Phi(x)$	$\mu\Phi(z) + \frac{v}{\sqrt{1+v}}\varphi(z)$	$\Phi(z) + \frac{\mu}{(1+v)^{3/2}}\varphi(z)$	$\frac{(1+v)(2+v)-\mu^2}{(1+v)^{5/2}}\varphi(z)$
Sine	$e^{-v/2}\sin\mu$	$e^{-v/2}\cos\mu$	$-e^{-v/2}\sin\mu$

We briefly derive the entries. For ReLU,

$$\eta_{\mathrm{ReLU}}(\mu, v) = \mathbb{E}[A\mathbf{1}\{A > 0\}] = \int_{-\alpha}^{\infty} (\mu + \sqrt{v}\,t)\varphi(t)\,\mathrm{d}t = \mu\Phi(\alpha) + \sqrt{v}\,\varphi(\alpha).$$

Differentiating gives

$$j_{\mathrm{ReLU}}(\mu, v) = \Phi(\alpha), \qquad \kappa_{\mathrm{ReLU}}(\mu, v) = \frac{\varphi(\alpha)}{\sqrt{v}}.$$

For the probit activation, introduce an independent $Z \sim \mathcal{N}(0, 1)$. Then

$$\mathbb{E}[\Phi(A)] = \Pr(Z \le A) = \Pr(A - Z \ge 0) = \Phi\left(\frac{\mu}{\sqrt{1+v}}\right),$$

and differentiating with respect to μ and v gives the table entries.

For GeLU, use

$$\mathbb{E}[A\Phi(A)] = \mathbb{E}[A\mathbf{1}\{A - Z \ge 0\}], \qquad Z \sim \mathcal{N}(0, 1),$$

with Z independent of A. Since $A - Z \sim \mathcal{N}(\mu, 1 + v)$, conditioning on $A - Z$ gives

$$\eta_{\mathrm{GeLU}}(\mu, v) = \mu\Phi(z) + \frac{v}{\sqrt{1+v}}\varphi(z), \qquad z = \frac{\mu}{\sqrt{1+v}}.$$

Differentiating this expression yields

$$j_{\mathrm{GeLU}}(\mu, v) = \Phi(z) + \frac{\mu}{(1+v)^{3/2}}\varphi(z),$$

and

$$\kappa_{\mathrm{GeLU}}(\mu, v) = \frac{(1+v)(2+v) - \mu^2}{(1+v)^{5/2}}\varphi(z).$$

For sine, the Gaussian characteristic function

$$\mathbb{E}[e^{iA}] = \exp(i\mu - v/2)$$

gives

$$\eta_{\sin}(\mu, v) = e^{-v/2}\sin\mu,$$

and differentiating gives the corresponding $j_{\sin}$ and $\kappa_{\sin}$.

Finally, ReLU can be viewed as a singular limit of a smoothed GeLU family. For $\beta > 0$, define

$$\mathrm{GeLU}_\beta(x) = x\Phi(\beta x).$$

Then $\mathrm{GeLU}_\beta \to \mathrm{ReLU}$ pointwise as $\beta \to \infty$. For finite β, the averaged Hessian coefficient is the ordinary expectation $\mathbb{E}[\mathrm{GeLU}''_\beta(A)]$. In the sharp limit, $\mathrm{GeLU}''_\beta \Rightarrow \delta_0$, and hence

$$\mathbb{E}[\mathrm{GeLU}''_\beta(A)] \to p_A(0) = \frac{\varphi(\mu/\sqrt{v})}{\sqrt{v}},$$

which is exactly the ReLU coefficient $\kappa_{\mathrm{ReLU}}(\mu, v)$. Thus the ReLU coefficient is a distributional kink mass, explaining its possible instability near $v = 0$.

F Affine-Operation Forward and Backward Equations

F.1 Dense and Convolutional Projections and AD Implementation

This appendix gives the projected affine-operation equations underlying the NN structure-aware IVON estimates. The experiments use MLPs and a CNN with diagonal Gaussian layer states, while the marginal-moment rules also apply to convolutional layers with shared Gaussian filters. Full-covariance dense-layer variants are included for completeness.

For a dense layer $\mathbf{a}_\ell^n = \mathbf{W}_\ell \mathbf{h}_{\ell-1}^n + \boldsymbol{d}_\ell$, assume independent factorized Gaussian weights and biases $(\mathbf{W}_\ell)_{ij} \sim \mathcal{N}((\mathbf{M}_\ell)_{ij}, (\mathbf{V}_\ell)_{ij})$ and $(\boldsymbol{d}_\ell)_i \sim \mathcal{N}((\mathbf{c}_\ell)_i, (\mathbf{s}_\ell)_i)$. For a diagonal incoming state, $q_{\ell-1}^{n,h} = \mathcal{N}(\mathbf{m}_{\ell-1}^{n,h}, \mathrm{Diag}(\mathbf{v}_{\ell-1}^{n,h}))$, the analytic preactivation moments are

$$\mathbf{m}_\ell^{n,a} = \mathbf{M}_\ell \mathbf{m}_{\ell-1}^{n,h} + \mathbf{c}_\ell, \qquad \mathbf{v}_\ell^{n,a} = (\mathbf{M}_\ell^{\circ 2} + \mathbf{V}_\ell)\mathbf{v}_{\ell-1}^{n,h} + \mathbf{V}_\ell (\mathbf{m}_{\ell-1}^{n,h})^{\circ 2} + \mathbf{s}_\ell.$$

For a full-covariance incoming state, the corresponding moments are

$$\mathbf{m}_\ell^{n,a} = \mathbf{M}_\ell \mathbf{m}_{\ell-1}^{n,h} + \mathbf{c}_\ell, \qquad \boldsymbol{\Sigma}_\ell^{n,a} = \mathbf{M}_\ell \boldsymbol{\Sigma}_{\ell-1}^{n,h} \mathbf{M}_\ell^\mathsf{T} + \mathrm{Diag}\Big(\mathbf{V}_\ell \Big[\mathrm{diag}(\boldsymbol{\Sigma}_{\ell-1}^{n,h}) + (\mathbf{m}_{\ell-1}^{n,h})^{\circ 2}\Big]\Big) + \mathrm{Diag}(\mathbf{s}_\ell).$$

The diagonal convolutional analogue uses the same formulas with matrix multiplication replaced by convolution with the mean and variance filters:

$$\mathbf{m}_\ell^{n,a} = \mathrm{Conv}_{\mathbf{M}_\ell}(\mathbf{m}_{\ell-1}^{n,h}) + \mathbf{c}_\ell, \qquad \mathbf{v}_\ell^{n,a} = \mathrm{Conv}_{\mathbf{M}_\ell^{\circ 2} + \mathbf{V}_\ell}(\mathbf{v}_{\ell-1}^{n,h}) + \mathrm{Conv}_{\mathbf{V}_\ell}((\mathbf{m}_{\ell-1}^{n,h})^{\circ 2}) + \mathbf{s}_\ell.$$

Bias moments are broadcast across output locations. The same random filter parameter is shared at every location, and the equations give marginal output means and variances under the diagonal-input and independence assumptions. Correlations induced by shared weights and overlapping receptive fields are not retained by the diagonal layer-state projection.

The activation projection Ψ_ℓ^h is specified in Appendix F.2. We obtain the projected mean-gradient and curvature estimates by automatic differentiation of the forward computation. The coefficient convention is essential: $\mathbf{j}$ and $\boldsymbol{\kappa}$ are held fixed only where they occur in the activation covariance approximation; the averaged activation mean $\boldsymbol{\eta}$ remains differentiable. Consequently, AD gives $\partial_m \boldsymbol{\eta} = \mathbf{j}$ and $2\partial_v \boldsymbol{\eta} = \boldsymbol{\kappa}$ while avoiding the additional coefficient-derivative terms outside the chosen FO/SO projections.

F.2 Forward and Backward Equations for the Four Variants

For completeness, we list the projected activation maps and adjoints for Diag-FO, Diag-SO, Full-FO, and Full-SO. The main experiments use diagonal layer states, with Diag-FO as the primary method; higher-order and full-covariance equations are included here for completeness. We suppress the example and layer indices (n, ℓ). Define

$$\mathbf{D}_j = \mathrm{Diag}(\mathbf{j}), \qquad \mathbf{D}_\kappa = \mathrm{Diag}(\boldsymbol{\kappa}),$$

where $\mathbf{j}$ and $\boldsymbol{\kappa}$ are the Gaussian-averaged GeLU coefficients. They are held fixed only in covariance terms, as explained in Appendix E.3. For diagonal states, $\mathbf{v}$ is the variance vector and $\boldsymbol{\gamma}$ the derivative with respect to that variance; the diagonal curvature channel is $\mathbf{H}^{\mathrm{diag}} = 2\boldsymbol{\gamma}$. For full-covariance states, the covariance-gradient channel is $\mathbf{G} = \mathbf{H}/2$, where $\mathbf{H}$ is the curvature channel of the main text.

Diag-FO. The diagonal first-order closure is

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a), \qquad \mathbf{v}_h = \mathbf{j}^{\circ 2} \odot \mathbf{v}_a.$$

The backward gradient equations are

$$\mathbf{g}_a = \mathbf{j} \odot \mathbf{g}_h, \qquad \boldsymbol{\gamma}_a = \mathbf{j}^{\circ 2} \odot \boldsymbol{\gamma}_h + \frac{1}{2}\boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Equivalently, in the diagonal Hessian/curvature channel,

$$\mathbf{H}_a^{\mathrm{diag}} = \mathbf{j}^{\circ 2} \odot \mathbf{H}_h^{\mathrm{diag}} + \boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Diag-SO. The diagonal second-order closure is

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \mathbf{v}_a), \qquad \mathbf{v}_h = \mathbf{j}^{\circ 2} \odot \mathbf{v}_a + \frac{1}{2}\boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a^{\circ 2}.$$

The backward gradient equations are

$$\mathbf{g}_a = \mathbf{j} \odot \mathbf{g}_h, \qquad \boldsymbol{\gamma}_a = \left(\mathbf{j}^{\circ 2} + \boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a\right) \odot \boldsymbol{\gamma}_h + \frac{1}{2}\boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Equivalently, in the diagonal Hessian/curvature channel,

$$\mathbf{H}_a^{\mathrm{diag}} = \left(\mathbf{j}^{\circ 2} + \boldsymbol{\kappa}^{\circ 2} \odot \mathbf{v}_a\right) \odot \mathbf{H}_h^{\mathrm{diag}} + \boldsymbol{\kappa} \odot \mathbf{g}_h.$$

Full-FO. The full-covariance first-order closure is

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \operatorname{diag} \boldsymbol{\Sigma}_a), \qquad \boldsymbol{\Sigma}_h = \mathbf{D}_j \boldsymbol{\Sigma}_a \mathbf{D}_j.$$

The backward gradient equations are

$$\mathbf{g}_a = \mathbf{D}_j \mathbf{g}_h, \qquad \mathbf{G}_a = \mathbf{D}_j \mathbf{G}_h \mathbf{D}_j + \frac{1}{2}\operatorname{Diag}(\boldsymbol{\kappa} \odot \mathbf{g}_h).$$

Equivalently, in the Hessian/curvature channel,

$$\mathbf{H}_a = \mathbf{D}_j \mathbf{H}_h \mathbf{D}_j + \operatorname{Diag}(\boldsymbol{\kappa} \odot \mathbf{g}_h).$$

Full-SO. The full-covariance second-order closure is

$$\mathbf{m}_h = \boldsymbol{\eta}(\mathbf{m}_a, \operatorname{diag} \boldsymbol{\Sigma}_a), \qquad \boldsymbol{\Sigma}_h = \mathbf{D}_j \boldsymbol{\Sigma}_a \mathbf{D}_j + \frac{1}{2}\bar{K}_\sigma[\boldsymbol{\Sigma}_a, \boldsymbol{\Sigma}_a],$$

where, for elementwise GeLU under the stopped-coefficient convention,

$$[\bar{K}_\sigma[\boldsymbol{\Sigma}, \boldsymbol{\Sigma}]]_{ij} = \boldsymbol{\kappa}_i \boldsymbol{\kappa}_j \boldsymbol{\Sigma}_{ij}^2.$$

The backward gradient equations are

$$\mathbf{g}_a = \mathbf{D}_j \mathbf{g}_h, \qquad \mathbf{G}_a = \mathbf{D}_j \mathbf{G}_h \mathbf{D}_j + \frac{1}{2}\operatorname{Diag}(\boldsymbol{\kappa} \odot \mathbf{g}_h) + \mathcal{K}_\sigma^*[\mathbf{G}_h],$$

with the explicit adjoint contribution

$$\mathcal{K}_\sigma^*[\mathbf{G}_h] = \mathbf{D}_\kappa(\mathbf{G}_h \odot \boldsymbol{\Sigma}_a)\mathbf{D}_\kappa$$

for symmetric $\mathbf{G}_h$ and $\boldsymbol{\Sigma}_a$. Equivalently, in the Hessian/curvature channel,

$$\mathbf{H}_a = \mathbf{D}_j \mathbf{H}_h \mathbf{D}_j + \operatorname{Diag}(\boldsymbol{\kappa} \odot \mathbf{g}_h) + 2\mathcal{K}_\sigma^*[\mathbf{G}_h].$$

Since $\mathcal{K}_\sigma^*$ is linear in its argument, the last term can also be written as $\mathcal{K}_\sigma^*[\mathbf{H}_h]$. The Full-SO and Diag-SO rules include the second-order covariance correction by definition.

F.3 Computational Cost and Approximation Error of Gaussian Activation Closures

Let d be the width of the current layer. Table 5 reports the activation-closure cost, excluding the affine matrix multiplications. The discussion applies to elementwise activations under the constant-coefficient Gaussian closures. For such activations, the second-order covariance correction is elementwise in the covariance entries, so it changes constant factors.

Table 5: Activation-closure costs for the four Gaussian activation variants. The main cost difference is whether layer-state covariances are diagonal or full; using the second-order closure changes only constant factors.

Variant	State storage	Forward	Backward
Diag-FO	$O(d)$	$O(d)$	$O(d)$
Diag-SO	$O(d)$	$O(d)$	$O(d)$
Full-FO	$O(d^2)$	$O(d^2)$	$O(d^2)$
Full-SO	$O(d^2)$	$O(d^2)$	$O(d^2)$

For a convolutional affine operation, computing the diagonal marginal moments uses the same convolution primitives and has the same order of cost as a small number of standard convolutions. For a dense affine layer with input width p and output width d, the diagonal Gaussian affine projection and backward gradient equations cost $O(dp)$ per example, matching ordinary dense-layer scaling up to constant factors. Full-covariance affine propagation requires products such as $\mathbf{M}_\ell \boldsymbol{\Sigma}_{\ell-1}^{n,h} \mathbf{M}_\ell^\mathsf{T}$, and the backward pass requires terms such as $\mathbf{M}_\ell^\mathsf{T} \mathbf{G}_\ell^{n,a} \mathbf{M}_\ell$. These have cost $O(dp^2 + d^2 p)$ and require $O(d^2)$ output-state storage. Thus, for MLPs, full covariance is usually the main computational bottleneck; the SO activation correction is relatively cheap once the full covariance is already stored.

The four variants differ in two approximation choices. Diag variants discard cross-covariances between hidden units, while Full variants retain them. FO keeps only the linearized activation covariance term, whereas SO also keeps the centered quadratic covariance contribution. For a smooth activation with $\mathbf{a} = \mathbf{m} + \boldsymbol{\xi}$, $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, and $\epsilon = \|\boldsymbol{\Sigma}\|$, the mean is exact if $\boldsymbol{\eta} = \mathbb{E}[\sigma(\mathbf{a})]$ is computed exactly. Under the constant-coefficient Taylor expansion,

$$\mathrm{Cov}[\sigma(\mathbf{a})] = \bar{\mathbf{J}} \boldsymbol{\Sigma} \bar{\mathbf{J}}^\mathsf{T} + O(\epsilon^2)$$

for the FO closure, while the SO closure captures the leading quadratic covariance term:

$$\mathrm{Cov}[\sigma(\mathbf{a})] = \bar{\mathbf{J}} \boldsymbol{\Sigma} \bar{\mathbf{J}}^\mathsf{T} + \frac{1}{2} \bar{K}_\sigma[\boldsymbol{\Sigma}, \boldsymbol{\Sigma}] + O(\epsilon^3).$$

Equivalently, the centered random-variable approximation errors are $O(\epsilon)$ for FO and $O(\epsilon^{3/2})$ for SO in L^2. For nonsmooth activations such as ReLU, these statements should be interpreted away from kinks or in the Gaussian-averaged weak sense discussed in Appendix E.6.

G Terminal Objectives and Prediction Rules

This appendix summarizes the projected output losses and prediction rules used in our experiments. For MNIST, Fashion-MNIST and CIFAR-10, we use a second-order delta approximation to the multiclass cross-entropy under a diagonal Gaussian logit state. For UCI regression, we use the analytic expected heteroscedastic Gaussian negative log-likelihood. In both cases, the required terminal adjoints are obtained by automatic differentiation of the stated objectives.

G.1 Multiclass Classification

For classification, let the terminal logit state be $\mathbf{Z} \sim \mathcal{N}(\mathbf{m}, \mathrm{Diag}(\mathbf{v}))$, with label $y \in \{1, \dots, K\}$. The exact expected cross-entropy,

$$\mathbb{E}_\mathbf{Z}[-\log \mathrm{softmax}_y(\mathbf{Z})] = \mathbb{E}_\mathbf{Z}[-Z_y + \mathrm{logsumexp}(\mathbf{Z})]\,,$$

has no closed form for general K. We therefore use the second-order delta approximation. Let $\mathbf{p} = \mathrm{softmax}(\mathbf{m})$ and $\mathbf{H}_{\mathrm{sm}} = \mathrm{Diag}(\mathbf{p}) - \mathbf{p}\mathbf{p}^\mathsf{T}$. Then

$$\mathcal{L}^{\mathrm{proj}}_{\mathrm{cls},\Delta}(\mathbf{m}, \mathbf{v}; y) = -m_y + \mathrm{logsumexp}(\mathbf{m}) + \frac{1}{2} \sum_{k=1}^{K} p_k (1 - p_k) v_k.$$

This is the deterministic terminal objective used for MNIST, Fashion-MNIST and CIFAR-10. During evaluation, we sample logits $\mathbf{Z}^{(s)} \sim \mathcal{N}(\mathbf{m}, \mathrm{Diag}(\mathbf{v}))$, average the softmax probabilities, and compute predictions from

$$\widehat{\mathbf{p}} = \frac{1}{S}\sum_{s=1}^{S} \mathrm{softmax}(\mathbf{Z}^{(s)}), \qquad \widehat{y} = \arg\max_k \widehat{p}_k.$$

We report accuracy and negative log-likelihood using these Monte Carlo averaged probabilities.

G.2 Heteroscedastic Gaussian Regression

For UCI regression, the network outputs a predictive mean channel and a log-noise-variance channel. For each output coordinate i,

$$y_i \mid M_i, \ell_i \sim \mathcal{N}(M_i, \exp(\ell_i)).$$

We use a diagonal Gaussian output-state approximation, $M_i \sim \mathcal{N}(m_{M,i}, v_{M,i})$ and $\ell_i \sim \mathcal{N}(m_{\ell,i}, v_{\ell,i})$. With $\mathbf{r} = \mathbf{m}_M - \mathbf{y}$, define

$$\boldsymbol{\alpha}_{\mathrm{het}} = \exp\left(\frac{1}{2}\mathbf{v}_\ell - \mathbf{m}_\ell\right), \qquad \boldsymbol{\beta}_{\mathrm{het}} = \mathbf{v}_M + \mathbf{r}^{\circ 2}.$$

The analytic expected heteroscedastic negative log-likelihood is

$$\mathcal{L}^{\mathrm{proj}}_{\mathrm{het}}(\mathbf{m}_M, \mathbf{v}_M, \mathbf{m}_\ell, \mathbf{v}_\ell; y) = \frac{1}{2}\sum_i \left[\log(2\pi) + m_{\ell,i} + \alpha_{\mathrm{het},i}\beta_{\mathrm{het},i}\right].$$

This objective is used for training the UCI regression models. For evaluation, we sample $(\mathbf{M}^{(s)}, \boldsymbol{\ell}^{(s)})$ from the diagonal Gaussian output state and use the Monte Carlo predictive mean

$$\widehat{\mathbf{y}} = \frac{1}{S}\sum_{s=1}^{S} \mathbf{M}^{(s)}$$

to compute RMSE. The reported NLPD is computed by Monte Carlo averaging the heteroscedastic Gaussian likelihood values from the sampled outputs.

H IVON Algorithms and Stochastic Propagation

H.1 IVON Algorithms

We first restate standard IVON [Shen et al., 2024], and then give our NN structure-aware variant. IVON maintains a diagonal Gaussian posterior $q_t(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}_t, \mathrm{Diag}(\boldsymbol{\sigma}_t^{\circ 2}))$, where $\boldsymbol{\sigma}_t = 1/\sqrt{\lambda_{\mathrm{ivon}}(\mathbf{h}_t + \delta)}$. Standard IVON estimates the mean-gradient and diagonal Hessian from sampled networks, whereas our variant replaces only these estimates by deterministic forward moment computations and their adjoints. Both algorithms retain IVON's momentum, quadratic curvature correction. The red lines mark the estimator changes.

Algorithm 1: Standard IVON [Shen et al., 2024]

Req.	Learning rates $\{\alpha_t\}$, damping/weight decay $\delta > 0$, momenta $\beta_1, \beta_2 \in [0, 1)$, Hessian init $h_0 > 0$, effective sample size λ_{ivon}.
Init.	$\mathbf{m} \leftarrow$ NN weights, $\mathbf{h} \leftarrow h_0$, $\mathbf{g} \leftarrow \mathbf{0}$, $\boldsymbol{\sigma} \leftarrow 1/\sqrt{\lambda_{\mathrm{ivon}}(\mathbf{h} + \delta)}$.
1	for $t = 1, 2, \ldots$ do
2	**Sampling and backward:** Sample $\boldsymbol{\theta} = \mathbf{m} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and compute $\widehat{\mathbf{g}} \leftarrow \widehat{\nabla}\ell_{\mathrm{data}}(\boldsymbol{\theta})$ by ordinary backpropagation.
3	**Hessian:** $\widehat{\mathbf{h}} \leftarrow \widehat{\mathbf{g}} \odot (\boldsymbol{\theta} - \mathbf{m})/\boldsymbol{\sigma}^{\circ 2}$.
4	$\mathbf{g} \leftarrow \beta_1 \mathbf{g} + (1 - \beta_1)\widehat{\mathbf{g}}, \quad \bar{\mathbf{g}} \leftarrow \mathbf{g}/(1 - \beta_1^t)$.
5	$\mathbf{h} \leftarrow \beta_2 \mathbf{h} + (1 - \beta_2)\widehat{\mathbf{h}} + \frac{1}{2}(1 - \beta_2)^2(\mathbf{h} - \widehat{\mathbf{h}})^{\circ 2}/(\mathbf{h} + \delta)$.
6	$\mathbf{m} \leftarrow \mathbf{m} - \alpha_t(\bar{\mathbf{g}} + \delta\mathbf{m})/(\mathbf{h} + \delta), \quad \boldsymbol{\sigma} \leftarrow 1/\sqrt{\lambda_{\mathrm{ivon}}(\mathbf{h} + \delta)}$.
7	end for; return $\mathbf{m}, \boldsymbol{\sigma}$.

Algorithm 2: NN structure-aware IVON

Req. Learning rates $\{\alpha_t\}$, damping/weight decay $\delta > 0$, momenta $\beta_1, \beta_2 \in [0, 1)$, Hessian init $h_0 > 0$, effective sample size λ_{ivon}.

Init. $\mathbf{m} \leftarrow$ NN weights, $\mathbf{h} \leftarrow h_0$, $\mathbf{g} \leftarrow \mathbf{0}$, $\boldsymbol{\sigma} \leftarrow 1/\sqrt{\lambda_{\text{ivon}}(\mathbf{h} + \delta)}$.

1 for $t = 1, 2, \ldots$ do

2 **Forward:** Propagate Gaussian moments using affine rules and Diag-FO or Diag-SO activation projections (Appendix F).

3 **Backward:** Differentiate the projected computation by AD with stop-gradient to obtain $\widehat{\mathbf{g}}$ and $\widehat{\mathbf{h}}$.

4 $\mathbf{g} \leftarrow \beta_1 \mathbf{g} + (1 - \beta_1)\widehat{\mathbf{g}}, \quad \bar{\mathbf{g}} \leftarrow \mathbf{g}/(1 - \beta_1^t)$.

5 $\mathbf{h} \leftarrow \beta_2 \mathbf{h} + (1 - \beta_2)\widehat{\mathbf{h}} + \frac{1}{2}(1 - \beta_2)^2(\mathbf{h} - \widehat{\mathbf{h}})^{\circ 2}/(\mathbf{h} + \delta)$.

6 $\mathbf{m} \leftarrow \mathbf{m} - \alpha_t(\bar{\mathbf{g}} + \delta\mathbf{m})/(\mathbf{h} + \delta), \quad \boldsymbol{\sigma} \leftarrow 1/\sqrt{\lambda_{\text{ivon}}(\mathbf{h} + \delta)}$.

7 end for; return $\mathbf{m}, \boldsymbol{\sigma}$.

H.2 Stochastic Propagation Maps and Monte Carlo IVON

The framework of Section 3 uses deterministic propagation maps for distributional states. Here we extend the same constrained formulation to *stochastic propagation maps*, showing how Monte Carlo IVON fits the forward–adjoint and BLR site structure. The main observation is at the level of the computational constraints: conditioning on posterior sampling noise gives an ordinary NN computation, and averaging its adjoint pullbacks recovers the expected loss-site natural parameters.

Stochastic forward constraints. For a diagonal Gaussian posterior, write $\mathbf{v} = \boldsymbol{\sigma}^{\circ 2}$ and reparameterize the weights as $\boldsymbol{\theta}(\boldsymbol{\mu}, \boldsymbol{\epsilon}) = \mathbf{m} + \sqrt{\mathbf{v}} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathbf{m}$ and $\mathbf{v}$ are functions of the Gaussian EF expectation parameter $\boldsymbol{\mu}$. Treat parameter sampling as a node of the computation graph. Conditional on $\boldsymbol{\epsilon}$, each activation is then computed using the sampled parameters. This gives stochastic local constraints

$$\mathbf{z}_v^n(\boldsymbol{\epsilon}) = \Psi_{v,\boldsymbol{\epsilon}}^n\big(\mathbf{z}_{\text{pa}(v)}^n(\boldsymbol{\epsilon}); \boldsymbol{\mu}\big). \tag{19}$$

Here $\mathbf{z}_v^n(\boldsymbol{\epsilon})$ denotes the sampled node state, rather than a Gaussian moment summary. The parameter-sampling node is included among the maps $\Psi_{v,\boldsymbol{\epsilon}}^n$. In particular, a sampled convolutional filter is reused at every spatial location where that parameter occurs; resampling it independently at different uses would change the model.

Stochastic lifted Lagrangian. At the output node, $\ell_n^{\text{out}}(\mathbf{z}_{n,\text{out}}(\boldsymbol{\epsilon})) = \ell_n(\boldsymbol{\theta}(\boldsymbol{\mu}, \boldsymbol{\epsilon}))$. Replacing the projected data losses by expectations over these sampled networks gives the stochastic analogue of (6):

$$\begin{aligned}\mathfrak{L}_{\text{MC}}(q, \boldsymbol{\mu}, \boldsymbol{\lambda}, \mathbf{z}, \boldsymbol{\xi}) = \mathcal{R}_h(q) + \mathcal{L}_0(\boldsymbol{\mu}) + \sum_n \mathbb{E}_{\boldsymbol{\epsilon}}[\ell_n^{\text{out}}(\mathbf{z}_{n,\text{out}}(\boldsymbol{\epsilon}))] \\ + \langle \boldsymbol{\lambda}, \boldsymbol{\mu} - \mathbb{E}_q[\mathbf{T}(\boldsymbol{\theta})] \rangle + \sum_{n,v} \mathbb{E}_{\boldsymbol{\epsilon}}\Big[\langle \boldsymbol{\xi}_{v,\boldsymbol{\epsilon}}^n, \Psi_{v,\boldsymbol{\epsilon}}^n(\mathbf{z}_{\text{pa}(v)}^n(\boldsymbol{\epsilon}); \boldsymbol{\mu}) - \mathbf{z}_v^n(\boldsymbol{\epsilon}) \rangle \Big].\end{aligned} \tag{20}$$

The statistical moment constraint is unchanged, and its stationary posterior agrees with the Gaussian law used by the sampling node; computational consistency now holds for each realization of the sampling noise. Stationarity in the sampled states gives the same backward adjoint recursion as (7), applied to the sampled graph. For a fixed $\boldsymbol{\epsilon}$, these adjoints are ordinary backpropagation sensitivities.

Taking the derivative with respect to the shared expectation parameter $\boldsymbol{\mu}$ gives the loss-site natural parameter contribution

$$\widehat{\boldsymbol{\lambda}}_n = -\mathbb{E}_{\boldsymbol{\epsilon}}\left[\sum_v \big(D_{\boldsymbol{\mu}} \Psi_{v,\boldsymbol{\epsilon}}^n\big)^* \boldsymbol{\xi}_{v,\boldsymbol{\epsilon}}^n \right] = -\nabla_{\boldsymbol{\mu}} \mathbb{E}_q[\ell_n(\boldsymbol{\theta})]. \tag{21}$$

The equality holds when the sampled graph represents the original NN and derivatives may be interchanged with expectations. Consequently, evaluating the bracketed pullback on a sample gives an unbiased Monte Carlo estimate of the BLR site, and the same prior-plus-data aggregation applies. This is the stochastic counterpart of (8), rather than a separate variational principle.

Recovery of IVON's estimators. For completeness, let $\widehat{\ell}_{n,\boldsymbol{\epsilon}}(\mathbf{m}, \mathbf{v}) := \ell_n(\mathbf{m} + \sqrt{\mathbf{v}} \odot \boldsymbol{\epsilon})$. The sample-wise gradient and diagonal-Hessian quantities obtained by the adjoint pullback are

$$\widehat{\mathbf{g}}_{n,\boldsymbol{\epsilon}} := \nabla_{\mathbf{m}} \widehat{\ell}_{n,\boldsymbol{\epsilon}} = \nabla_{\boldsymbol{\theta}} \ell_n(\boldsymbol{\theta}), \qquad \widehat{\mathbf{h}}_{n,\boldsymbol{\epsilon}} := 2\nabla_{\mathbf{v}} \widehat{\ell}_{n,\boldsymbol{\epsilon}} = \widehat{\mathbf{g}}_{n,\boldsymbol{\epsilon}} \odot \frac{\boldsymbol{\theta} - \mathbf{m}}{\mathbf{v}}. \tag{22}$$

Their expectations recover $\nabla_{\mathbf{m}} \mathbb{E}_q[\ell_n]$ and $2\nabla_{\mathbf{v}} \mathbb{E}_q[\ell_n] = \mathbb{E}_q[\mathrm{diag}(\nabla^2_{\boldsymbol{\theta}} \ell_n)]$ by the Gaussian identities, under the usual regularity conditions. In canonical diagonal-Gaussian expectation coordinates $\boldsymbol{\mu} = (\mathbf{m}, \mathbf{m}^{\circ 2} + \mathbf{v})$, the corresponding sample-wise site is

$$\widehat{\boldsymbol{\lambda}}_{n,\boldsymbol{\epsilon}} = \left(-\widehat{\mathbf{g}}_{n,\boldsymbol{\epsilon}} + \mathbf{m} \odot \widehat{\mathbf{h}}_{n,\boldsymbol{\epsilon}}, -\tfrac{1}{2} \widehat{\mathbf{h}}_{n,\boldsymbol{\epsilon}} \right), \qquad \mathbb{E}_{\boldsymbol{\epsilon}}[\widehat{\boldsymbol{\lambda}}_{n,\boldsymbol{\epsilon}}] = \widehat{\boldsymbol{\lambda}}_n. \tag{23}$$

The connection to our formulation is that they arise from the same constrained forward–adjoint construction, now with sample-dependent rather than moment-propagation maps. The stochastic quantities are unbiased for the indicated expected loss derivatives.

Relation to deterministic Gaussian projection. In the stochastic formulation, the expected loss is evaluated by averaging adjoint pullbacks through sampled networks; in NN structure-aware IVON, it is approximated by propagating Gaussian layer states and taking the adjoint of the resulting projected computation. The former is exact in expectation when sampling respects all parameter dependencies, while the latter can avoid parameter-sampling variance but introduces projection bias. The framework also permits mixed stochastic and analytic maps, although approximating any part of the forward computation generally changes the target expected loss.

I Numerical Experiment Details

I.1 Configuration Details

We implemented standard IVON and the proposed NN structure-aware variants in JAX [Bradbury et al., 2018] and Equinox [Kidger and Garcia, 2021]. All methods use the same outer IVON update; Diag-FO and Diag-SO replace its Monte Carlo gradient and diagonal-Hessian estimates with projected Gaussian moment-forward and adjoint-backward computations. We use factorized Gaussian weight posteriors and diagonal Gaussian layer states. For MNIST and Fashion-MNIST, we use two-hidden-layer MLPs with widths $(128, 64)$; for UCI regression, we use one-hidden-layer MLPs with width 50. For CIFAR-10, we use a four-layer CNN with channels $(16, 16, 32, 32)$, 3×3 convolutions, average pooling, and a linear classifier. All networks use GeLU activations. The experiments were conducted on V100 GPUs.

All methods were trained for 50 epochs with learning rate 0.01 and $\beta_1 = 0.9$. For MNIST and Fashion-MNIST, we used minibatch size 64, initial Hessian estimate $h_0 = 0.1$, $\beta_2 = 0.9999$, and weight decay 10^{-5}. CIFAR-10 uses the same settings except $\beta_2 = 0.99999$. For classification, we applied MixUp augmentation with $\alpha = 0.1$. For UCI regression, we used minibatch size 32, $h_0 = 10$, $\beta_2 = 0.999$, and weight decay 10^{-3}. We used a heteroscedastic Gaussian likelihood, clipping the predicted log variance to $[-5, 5]$ and the IVON update direction to $[-0.1, 0.1]$ elementwise.

Standard IVON used one Monte Carlo weight sample per minibatch during training. At test time, we used 1000 Monte Carlo samples for IVON, and the structure-aware methods evaluated predictive quantities using 1000 samples from projected Gaussian output states. We report test accuracy, negative log predictive density (NLPD), and expected calibration error (ECE) for classification, and test RMSE and NLPD for regression. UCI metrics are reported on standardized target scales. All tabulated results are from epoch 50 and report mean $\pm$ sample standard deviation over 20 independent runs.

The code is available here.

I.2 UCI Regression Test Performance

Table 6 compares standard IVON with Diag-FO on eight UCI regression datasets. Diag-FO gives lower mean test RMSE and NLPD on all eight datasets, with larger improvements on Naval and Yacht. These results show that our NN structure-aware IVON can improve predictive performance without changing the outer IVON update.

Table 6: UCI regression test performance on standardized targets at epoch 50 (mean ± sample standard deviation over 20 runs). Lower RMSE and NLPD are better. Best mean results are bolded.

	Test RMSE ↓		Test NLPD ↓	
Dataset	IVON	Diag-FO	IVON	Diag-FO
Boston	0.402 ± 0.104	$\mathbf{0.381 \pm 0.107}$	0.365 ± 0.158	$\mathbf{0.302 \pm 0.151}$
Concrete	0.394 ± 0.042	$\mathbf{0.390 \pm 0.042}$	0.391 ± 0.123	$\mathbf{0.374 \pm 0.120}$
Energy	0.276 ± 0.029	$\mathbf{0.269 \pm 0.027}$	-0.239 ± 0.110	$\mathbf{-0.397 \pm 0.104}$
Kin8nm	0.373 ± 0.020	$\mathbf{0.366 \pm 0.017}$	0.335 ± 0.046	$\mathbf{0.318 \pm 0.046}$
Naval	0.084 ± 0.020	$\mathbf{0.054 \pm 0.016}$	-1.133 ± 0.193	$\mathbf{-1.379 \pm 0.072}$
Power	0.261 ± 0.011	$\mathbf{0.254 \pm 0.009}$	0.078 ± 0.047	$\mathbf{0.044 \pm 0.030}$
Wine	0.810 ± 0.067	$\mathbf{0.801 \pm 0.055}$	1.185 ± 0.098	$\mathbf{1.182 \pm 0.099}$
Yacht	0.377 ± 0.074	$\mathbf{0.337 \pm 0.084}$	0.335 ± 0.140	$\mathbf{0.099 \pm 0.157}$

I.3 Comparison of Diag-FO and Diag-SO

We compare the two sampling-free activation projections using the same evaluation metrics. As shown in Tables 7 and 8, they yield similar test performance across classification and regression datasets.

Table 7: Diag-FO versus Diag-SO on classification test sets at epoch 50 (mean ± sample standard deviation over 20 runs). Higher accuracy and lower NLPD/ECE are better.

Dataset	Method	Accuracy (%) ↑	NLPD ↓	ECE ↓
MNIST	Diag-FO	98.18 ± 0.09	0.0824 ± 0.0057	0.0311 ± 0.0056
	Diag-SO	98.17 ± 0.10	0.0834 ± 0.0057	0.0322 ± 0.0056
Fashion-MNIST	Diag-FO	88.95 ± 0.36	0.3238 ± 0.0069	0.0105 ± 0.0028
	Diag-SO	88.92 ± 0.31	0.3236 ± 0.0071	0.0111 ± 0.0027
CIFAR-10	Diag-FO	74.62 ± 0.49	0.7480 ± 0.0116	0.0196 ± 0.0055
	Diag-SO	74.65 ± 0.47	0.7473 ± 0.0114	0.0200 ± 0.0056

Table 8: Diag-FO versus Diag-SO on UCI regression test sets at epoch 50, on standardized targets (mean ± sample standard deviation over 20 runs). Lower is better.

	Test RMSE ↓		Test NLPD ↓	
Dataset	Diag-FO	Diag-SO	Diag-FO	Diag-SO
Boston	0.381 ± 0.107	0.381 ± 0.107	0.302 ± 0.151	0.302 ± 0.151
Concrete	0.390 ± 0.042	0.390 ± 0.042	0.374 ± 0.120	0.374 ± 0.120
Energy	0.269 ± 0.027	0.269 ± 0.027	-0.397 ± 0.104	-0.397 ± 0.104
Kin8nm	0.366 ± 0.017	0.367 ± 0.017	0.318 ± 0.046	0.324 ± 0.049
Naval	0.054 ± 0.016	0.058 ± 0.018	-1.379 ± 0.072	-1.367 ± 0.076
Power	0.254 ± 0.009	0.253 ± 0.010	0.044 ± 0.030	0.040 ± 0.032
Wine	0.801 ± 0.055	0.801 ± 0.055	1.182 ± 0.099	1.182 ± 0.098
Yacht	0.337 ± 0.084	0.337 ± 0.084	0.099 ± 0.157	0.099 ± 0.157

J Additional Experimental Results

We provide further experiments on the accuracy and computational cost of our moment-based approximations, comparisons with AdamW and DVI, the stop-gradient convention, and deeper networks. The additional classification comparisons and ablations report results over 10 independent runs, separately from the 20-run results in Section 7 and Appendix I. Approximation-quality and runtime measurements follow the protocols specified below.

J.1 Approximation Quality

We first assess how accurately the deterministic moment projections approximate the posterior-expected loss, gradient, and diagonal Hessian used by IVON. We use a synthetic regression task with width-64 MLPs of varying depth L. For each quantity, a high-accuracy Monte Carlo (MC) reference is computed using 10^5 samples from the Gaussian weight posterior maintained by IVON. The MC estimates converge to the targeted posterior expectations as the sample size increases.

For the loss, we report the relative error $\frac{|\widehat{\mathcal{L}}-\mathcal{L}_{\mathrm{MC}}|}{|\mathcal{L}_{\mathrm{MC}}|}$. For the gradient and Hessian estimates, we report cosine similarity with the corresponding MC references, $\cos(g,\hat{g}) = \frac{\langle g,\hat{g}\rangle}{\|g\|\,\|\hat{g}\|}$, $\cos(h,\hat{h}) = \frac{\langle h,\hat{h}\rangle}{\|h\|\,\|\hat{h}\|}$. Here, g and h denote the MC reference gradient and diagonal-Hessian quantities, while $\hat{g}$ and $\hat{h}$ denote their projected counterparts. Tables 9 and 10 report Diag-FO/Diag-SO at random initialization and after fitting the network with IVON, respectively.

Table 9: Approximation quality at random initialization. Values are reported as Diag-FO/Diag-SO. The reference is obtained using 10^5 posterior samples from IVON. Lower relative loss error is better, whereas higher cosine similarity is better.

L	Rel. loss ↓	$\cos(g)$ ↑	$\cos(h)$ ↑
2	7.55×10^{-3} / 1.18×10^{-3}	1.000 / 1.000	0.999 / 0.999
4	1.70×10^{-2} / 9.45×10^{-3}	1.000 / 1.000	0.990 / 0.990
8	2.27×10^{-1} / 1.24×10^{-1}	0.963 / 0.967	0.964 / 0.962

Table 10: Approximation quality after fitting the network using IVON. Values are reported as Diag-FO/Diag-SO.

L	Rel. loss ↓	$\cos(g)$ ↑	$\cos(h)$ ↑
2	4.64×10^{-2} / 5.44×10^{-2}	0.997 / 0.997	0.995 / 0.995
4	2.84×10^{-1} / 3.07×10^{-1}	1.000 / 1.000	0.996 / 0.996
8	3.49×10^{-1} / 9.18×10^{-2}	0.516 / 0.533	0.906 / 0.909

For $L = 2$ and $L = 4$, the projected gradient and diagonal-Hessian directions are closely aligned with the MC references. Approximation quality deteriorates at $L = 8$, especially after fitting: gradient cosine similarity drops to about 0.52, although the diagonal-Hessian cosine remains above 0.90. Diag-SO reduces the fitted $L = 8$ loss error, but has little effect on the gradient and Hessian directions. This illustrates how projection error can accumulate with depth.

J.2 Computational Cost

We measure per-step runtime and peak device memory, including the forward and backward computations, using the synthetic MLPs from Section J.1. Results are normalized at each depth to one-sample IVON (time and memory equal to 1.00). Entries are reported as IVON/Diag-FO/Diag-SO.

Table 11: Normalized optimization-step runtime and peak device memory for the synthetic MLP experiment. Values are reported as IVON/Diag-FO/Diag-SO, with one-sample IVON normalized to 1.00 separately for each depth.

L	Time ↓	Peak memory ↓
2	1.00 / 1.06 / 1.05	1.00 / 1.17 / 1.17
4	1.00 / 1.23 / 1.28	1.00 / 1.17 / 1.17
8	1.00 / 1.40 / 1.41	1.00 / 1.17 / 1.17

The deterministic variants are somewhat more expensive than one-sample IVON in this setting. Their runtime overhead increases with depth, reaching about $1.4\times$ at $L = 8$, while peak memory is approximately $1.17\times$ across depths. Diag-FO and Diag-SO have similar computational cost.

J.3 Additional Classification Results

We compare Diag-FO with AdamW, standard IVON, and deterministic variational inference (DVI) [Wu et al., 2019] on Fashion-MNIST and CIFAR-10, using 10 independent runs. We use the same two-hidden-layer MLP for Fashion-MNIST and four-convolution-layer CNN for CIFAR-10 as in Appendix I. DVI maintains a diagonal Gaussian variational posterior, propagates approximate activation moments, and optimizes its mean and variance parameters using Adam. For this comparison, its posterior is initialized as in the IVON variants. The test metrics are the same as in the main classification experiments.

Table 12: Fashion-MNIST test performance at epoch 50 (mean $\pm$ sample standard deviation over 10 runs). Higher accuracy and lower NLPD/ECE are better; best displayed means are bolded.

Method	Accuracy (%) $\uparrow$	NLPD $\downarrow$	ECE $\downarrow$
AdamW	88.98 ± 0.18	0.3808 ± 0.0079	0.0376 ± 0.0037
DVI	89.01 ± 0.13	0.3309 ± 0.0036	0.0380 ± 0.0037
IVON	88.54 ± 0.24	0.3286 ± 0.0055	0.0249 ± 0.0055
Diag-FO	$\mathbf{89.08 \pm 0.25}$	$\mathbf{0.3211 \pm 0.0050}$	$\mathbf{0.0102 \pm 0.0019}$

Table 13: CIFAR-10 test performance at epoch 50 (mean $\pm$ sample standard deviation over 10 runs). Higher accuracy and lower NLPD/ECE are better; best displayed means are bolded.

Method	Accuracy (%) $\uparrow$	NLPD $\downarrow$	ECE $\downarrow$
AdamW	72.77 ± 0.51	0.8870 ± 0.0177	0.0752 ± 0.0075
DVI	72.68 ± 0.72	0.8218 ± 0.0182	0.0848 ± 0.0050
IVON	74.34 ± 0.64	0.7636 ± 0.0138	0.0380 ± 0.0061
Diag-FO	$\mathbf{74.70 \pm 0.53}$	$\mathbf{0.7478 \pm 0.0133}$	$\mathbf{0.0209 \pm 0.0047}$

Diag-FO obtains the highest mean accuracy and lowest mean NLPD and ECE among the reported methods on both datasets.

J.4 Effect of the Constant-Coefficient Convention

Our projected backward computation uses the constant-coefficient (stop-gradient) convention: the Gaussian-averaged coefficients are evaluated at the current forward state and held fixed when differentiating the covariance closure. We compare this proposed Diag-FO rule ("stop gradient") with differentiation through the coefficients ("no stop gradient") on Fashion-MNIST and CIFAR-10. Both variants use the same architectures, hyperparameters, and 10 random seeds.

Table 14: Effect of stopping gradients through the Gaussian-averaged moment coefficients on Fashion-MNIST at epoch 50. Mean $\pm$ sample standard deviation over 10 runs.

Diag-FO variant	Accuracy (%) $\uparrow$	NLPD $\downarrow$	ECE $\downarrow$
Stop gradient	89.08 ± 0.25	0.3211 ± 0.0050	0.0102 ± 0.0019
No stop gradient	89.15 ± 0.19	0.3200 ± 0.0052	0.0111 ± 0.0029

Table 15: Effect of stopping gradients through the Gaussian-averaged moment coefficients on CIFAR-10 at epoch 50. Mean $\pm$ sample standard deviation over 10 runs.

Diag-FO variant	Accuracy (%) $\uparrow$	NLPD $\downarrow$	ECE $\downarrow$
Stop gradient	74.70 ± 0.53	0.7478 ± 0.0133	0.0209 ± 0.0047
No stop gradient	74.64 ± 0.57	0.7499 ± 0.0150	0.0206 ± 0.0047

The two variants perform similarly within run-to-run variability. The stop-gradient convention is therefore motivated primarily by the intended local backward approximation, rather than by a consistent empirical advantage over differentiating through the coefficients.

J.5 Performance with Deeper Networks

We also compare standard IVON and Diag-FO using depth-8 networks on Fashion-MNIST and CIFAR-10, retaining the remaining training and evaluation settings. We report test metrics over 10 runs.

Table 16: Test performance for depth-8 networks on Fashion-MNIST at epoch 50 (mean $\pm$ sample standard deviation over 10 runs).

Method	Accuracy (%) ↑	NLPD ↓	ECE ↓
IVON	87.95 ± 0.43	0.3529 ± 0.0079	0.0303 ± 0.0056
Diag-FO	87.42 ± 0.59	0.3796 ± 0.0144	0.0200 ± 0.0059

Table 17: Test performance for depth-8 networks on CIFAR-10 at epoch 50 (mean $\pm$ sample standard deviation over 10 runs).

Method	Accuracy (%) ↑	NLPD ↓	ECE ↓
IVON	77.46 ± 0.49	0.6714 ± 0.0123	0.0508 ± 0.0069
Diag-FO	78.35 ± 0.58	0.6503 ± 0.0160	0.0148 ± 0.0028

The effect of increased depth depends on the task. Compared with the shallower networks, the depth-8 models show worse test accuracy and NLPD on Fashion-MNIST, but improved performance on CIFAR-10. At depth 8, IVON achieves higher accuracy and lower NLPD on Fashion-MNIST, while Diag-FO obtains lower ECE. On CIFAR-10, Diag-FO improves all three test metrics. Together with Section J.1, these results suggest that approximation errors can accumulate with depth, but do not necessarily lead to a collapse in optimization and predictive performance.

J.6 Runtime Scaling with Monte Carlo Sample Size

Finally, we compare Diag-FO with IVON using different numbers of Monte Carlo weight samples per update. Tables 18 and 19 report normalized runtime / peak device memory for MLPs and CNNs. One-sample IVON is normalized to $1.00/1.00$ separately for each architecture and depth.

Table 18: Normalized runtime / peak device memory for MLPs. One-sample IVON is normalized to $1.00/1.00$ separately for each depth.

Method	$L = 2$	$L = 4$	$L = 8$
IVON, MC$= 1$	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00
IVON, MC$= 2$	1.05 / 1.00	1.17 / 1.00	1.20 / 1.00
IVON, MC$= 4$	1.23 / 1.00	1.61 / 1.00	2.06 / 1.00
IVON, MC$= 6$	1.34 / 1.00	2.07 / 1.00	2.49 / 1.00
Diag-FO	1.06 / 1.17	1.23 / 1.17	1.40 / 1.17

Table 19: Normalized runtime / peak device memory for CNNs. One-sample IVON is normalized to $1.00/1.00$ separately for each depth.

Method	$L = 2$	$L = 4$	$L = 8$
IVON, MC$= 1$	1.00 / 1.00	1.00 / 1.00	1.00 / 1.00
IVON, MC$= 2$	1.36 / 1.00	1.71 / 1.00	1.65 / 1.00
IVON, MC$= 4$	2.25 / 1.00	2.79 / 1.00	2.71 / 1.00
IVON, MC$= 6$	2.83 / 1.00	3.68 / 1.00	3.78 / 1.00
Diag-FO	1.66 / 1.00	2.13 / 1.00	2.02 / 1.00

Across all reported MLP and CNN configurations, the per-step runtime of Diag-FO consistently falls between that of two- and four-sample IVON. For example, at $L = 8$ the normalized MLP runtimes are 1.40 for Diag-FO, compared with 1.20 and 2.06 for two- and four-sample IVON, respectively. This illustrates the computational trade-off: deterministic moment propagation is more expensive than one-sample IVON, but becomes relatively more efficient when multiple posterior samples are required.

K Limitations and Future Directions

Gaussian approximations and projection bias. Our current implementation uses factorized Gaussian weight posteriors and diagonal Gaussian layer states. Gaussian approximations are most justified for wide affine preactivations, where many weakly dependent contributions are aggregated. The diagonal approximation is more restrictive, as it neglects correlations between activations. These assumptions may become less accurate in narrow or deep networks, or when dependencies accumulate during training. Indeed, Section J.1 shows that projection errors can increase with depth. Nevertheless, the deeper-network experiments in Section J.5 show that Diag-FO can remain competitive despite substantial approximation errors. Moment-based estimates trade Monte Carlo variance for projection bias, and an imperfect estimate can still provide useful optimization directions.

Structured weight uncertainty. The diagonal weight posterior is primarily a computational choice. Extending IVON to structured covariances is challenging because correlated uncertainty and curvature-based preconditioning must be handled during training, rather than estimated only after optimization as in post-hoc Laplace approximations. Recent methods, such as SOAP Bubbles/EVON [Minut et al., 2026], demonstrate scalable structured weight uncertainty. Combining such posteriors with our layer-state projections and adjoints is a promising direction.

Extension to more complex architectures. The forward–adjoint formulation in Section 3 applies to general NN computation graphs, but its practical implementation requires suitable moment-propagation maps and their adjoints. Dense and convolutional layers are affine operations and admit relatively simple Gaussian moment rules, as illustrated by our MLP and CNN experiments. However, the diagonal approximation neglects correlations induced by convolutional parameter sharing. Attention and normalization pose additional challenges: query–key interactions are bilinear, while softmax and normalization couple coordinates and can produce strongly non-Gaussian states. Existing sampling-free prediction methods simplify these computations [Li et al., 2025], and recent work has introduced improved propagation and calibration techniques [Wieczorek et al., 2026]. Our training-time setting additionally requires efficient backward adjoints through these projections, including gradient and curvature information. Richer covariance representations or hybrid analytic–Monte Carlo projections may improve accuracy, but introduce additional computational and memory costs. Developing such projections and their adjoints for larger architectures remains an important direction for future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our extension of BLR and recovery and connection with other methods are discussed in Section 3, Section 4. Comparison with IVON and experimental results are provided in Section 6 and Section 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: This is explained in the conculusion section and App K.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Provided in Appendix sections like App E.5

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Configurations are detailed in App I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The link is provided in App I.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: This is explained in App I.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We have reported standard deviations accorss independent runs.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: This is covered in App I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper mainly provides theoretical framework.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited in App I.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: NA

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.