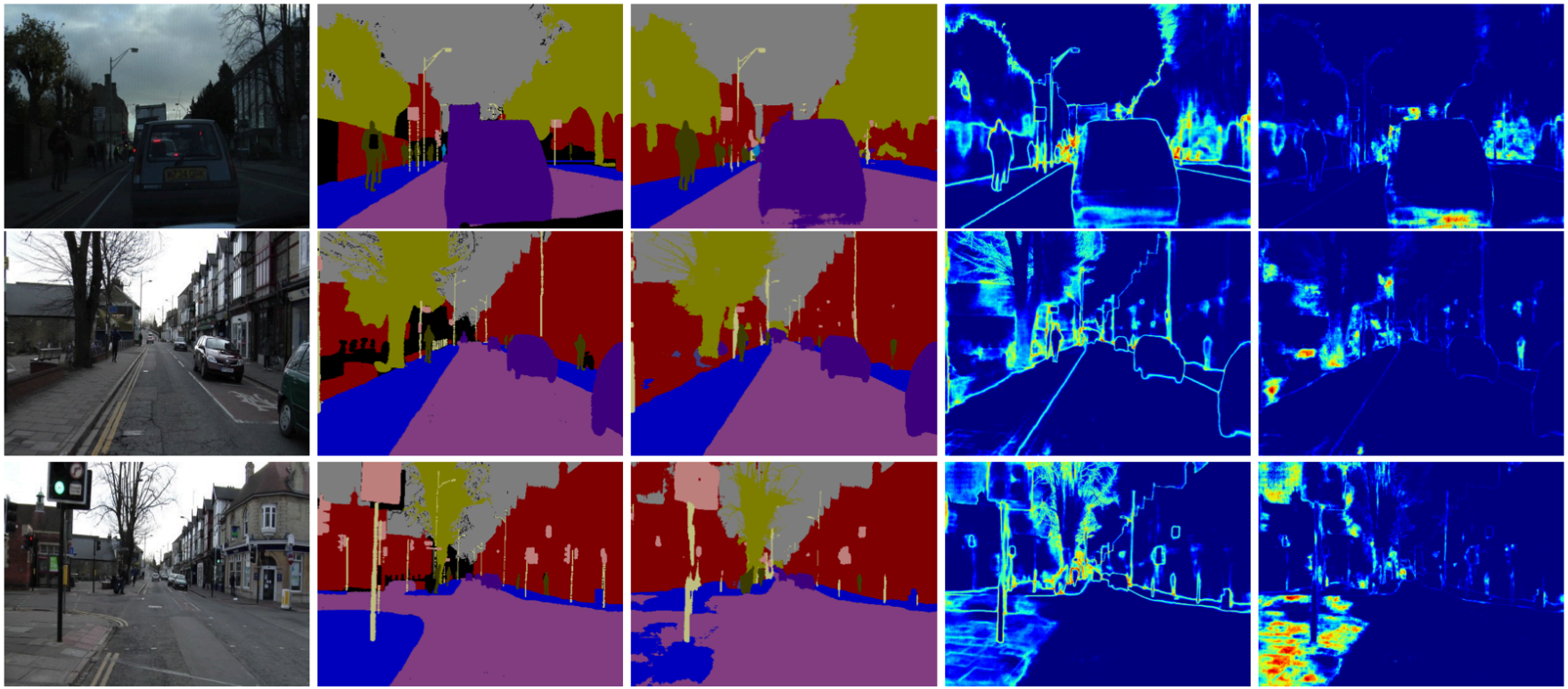
Advances in Laplace Inference for Reliable and Interpretable Bayesian Deep Learning

Kouroche Bouchiat

ETH Zürich

March 25th, 2025

Motivation



(a) Input Image (b) Ground Truth (c) Semantic Segmentation (d) Aleatoric Uncertainty (e) Epistemic Uncertainty

Figure: Uncertainty for semantic segmentation. (Fig. 1 of *Kendall et al., 2017*)

Laplace approximation: Notation

We will be focusing on supervised learning over inputs $\boldsymbol{x}_{1:n}$ and labels $y_{1:n}$.

Neural network: Let $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ be a neural network with parameters $\boldsymbol{\theta}$.

$$f_{\boldsymbol{\theta}} \triangleq f^{(L)}_{\boldsymbol{\theta}^{(L)}} \circ \cdots \circ f^{(1)}_{\boldsymbol{\theta}^{(1)}}, \;\; \boldsymbol{\theta} \triangleq \texttt{Concat}\left[\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(L)}\right] \in \mathbb{R}^P, \;\; \boldsymbol{\theta}^{(\ell)} \in \mathbb{R}^{D_{\text{in}} \cdot D_{\text{out}}}$$

Probabilistic inference: Update our belief over $\boldsymbol{\theta}$ after seeing $\{\boldsymbol{x}_{1:n}, y_{1:n}\}$.

$$\underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n})}_{\text{posterior dist.}} = \frac{\overbrace{p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta})}^{\text{likelihood}} \overbrace{p(\boldsymbol{\theta})}^{\text{prior dist.}}}{\underbrace{\int p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}_{\text{marginal likelihood}}}$$

Laplace approximation: VI and MAP estimation

Variational inference: Goal is to approximate the **posterior distribution**.

$$p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n}) = \frac{p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{\int p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, d\boldsymbol{\theta}} \approx q(\boldsymbol{\theta} \mid \boldsymbol{\lambda}) \triangleq q_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$$

MAP estimation: Rely on SGD to find a maximum *a posteriori* estimate.

$$\begin{aligned}
\hat{\boldsymbol{\theta}} &= \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n}) \\
&= \arg\min_{\boldsymbol{\theta}} -\log p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n}) \\
&= \arg\min_{\boldsymbol{\theta}} -\log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta})
\end{aligned}$$

Laplace approximation: Taylor expansion

Laplace's method (*P.S. Laplace, 1774*): Let $\psi(\boldsymbol{\theta}) \triangleq \log p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n})$.
Take the 2nd-order Taylor expansion of $\psi(\boldsymbol{\theta})$ around the MAP estimate $\hat{\boldsymbol{\theta}}$.

$$\psi(\boldsymbol{\theta}) \approx \psi(\hat{\boldsymbol{\theta}}) + (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}) \cancel{\left[\ldots\right]} + \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^\top \left[\nabla^2_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]_{\hat{\boldsymbol{\theta}}} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$$

Compare this to the log-p.d.f. of a multivariate Normal distribution (MVN).

$$\log \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu}) + \text{const.}$$

Laplace approximation: Mean and covariance

The **Laplace approximation** q of p is a MVN (mean $\hat{\boldsymbol{\theta}}$, cov. matrix $\hat{\boldsymbol{\Sigma}}$).

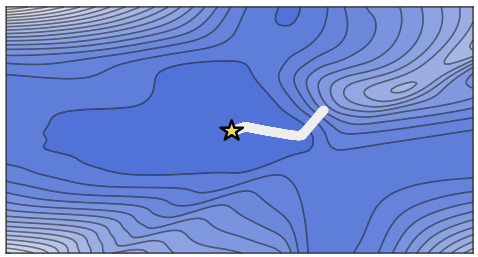
$$q(\boldsymbol{\theta}) \triangleq \mathcal{N}(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\Sigma}}) \quad \hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta}) \quad \hat{\boldsymbol{\Sigma}} = -\left[\nabla^2_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]^{-1}_{\hat{\boldsymbol{\theta}}}$$

- Beware! $\hat{\boldsymbol{\Sigma}}$ needs to be *symmetric* and *positive semi-definite*.
- However $\psi(\boldsymbol{\theta})$ being twice continuously diff. around $\hat{\boldsymbol{\theta}}$ is sufficient:
  - Symmetric: Order of differentiation does not matter.
  - Positive semi-definite: Hessian taken at $\hat{\boldsymbol{\theta}}$ is negative semi-definite.

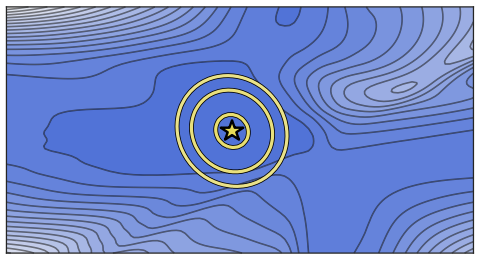
Laplace approximation: Predictive posterior

How do we make predictions using our Laplace approximation?

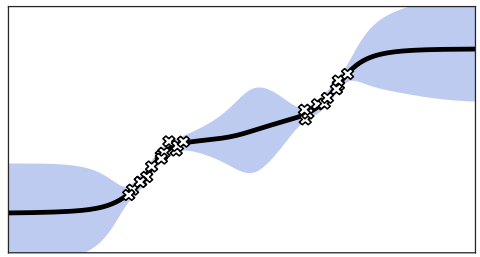
$$
\begin{aligned}
p(y^\star \mid \boldsymbol{x}^\star, \boldsymbol{x}_{1:n}, y_{1:n}) &= \int p(y^\star \mid \boldsymbol{x}^\star, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \boldsymbol{x}_{1:n}, y_{1:n})\, d\boldsymbol{\theta} \\
&\approx \int p(y^\star \mid \boldsymbol{x}^\star, \boldsymbol{\theta})\, q(\boldsymbol{\theta})\, d\boldsymbol{\theta} \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\theta} \sim q} \Big[p(y^\star \mid \boldsymbol{x}^\star, \boldsymbol{\theta}) \Big]
\end{aligned}
$$



(a) Step 1: Find MAP



(b) Step 2: Fit approx.



(c) Step 3: Predict!

Figure: Fig. 1 of (*Daxberger et al., 2021*)

Laplace approximation: Bayesian neural networks?

LA hinges on inverting Hessian of the log-**posterior distribution** w.r.t. $\boldsymbol{\theta}$

$$\hat{\boldsymbol{\Sigma}} = -\left[\nabla^2_{\boldsymbol{\theta}} \psi(\boldsymbol{\theta})\right]^{-1}_{\hat{\boldsymbol{\theta}}} = -\left[\nabla^2_{\boldsymbol{\theta}} \log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta}) + \nabla^2_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta})\right]^{-1}_{\hat{\boldsymbol{\theta}}}$$

- Known to work with small neural networks. (*D.J.C. MacKay, 1992*)
- However the Hessian quickly becomes problematic as P grows.
  - Auto-diff. frameworks help but Hessian still needs $\mathcal{O}(P^2)$ storage.
  - Inversion can also be difficult. (compute & numerical instability)
  - Not guaranteed to be negative semi-definite! (e.g. saddle points)

Hessian approximation: Fisher information matrix

Fisher information: Avoid 2^{nd}-order differentiation and focus on the score.

$$\hat{\boldsymbol{\Sigma}} \approx \boldsymbol{\Sigma}_{\text{Fish.}}, \quad \boldsymbol{\Sigma}_{\text{Fish.}} \triangleq - \left[\sum_{i=1}^{n} \mathop{\mathbb{E}}_{y \sim p(\dots)} \left[\left[\nabla_{\boldsymbol{\theta}} \log p(y \mid \boldsymbol{x}_i, \boldsymbol{\theta}) \right]_{\hat{\boldsymbol{\theta}}}^{2} \right] + \dots \right]^{-1}$$

Note: Guaranteed positive semi-definite. (cf. info. geometry, natural grad.)

Hessian approximation: Gen. Gauss-Newton matrix

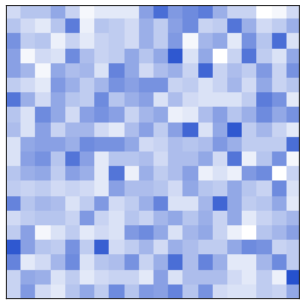
Gen. Gauss-Newton (GGN): Let $\mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}) \triangleq \left[\nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x})\right]_{\hat{\boldsymbol{\theta}}}$ denote Jacobian.

$$\hat{\boldsymbol{\Sigma}} \approx \boldsymbol{\Sigma}_{\text{GGN}},\ \boldsymbol{\Sigma}_{\text{GGN}} \triangleq -\left[\sum_{i=1}^{n} \mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \left[\nabla_f^2 \log p(y_i \mid f)\right]_{\hat{f}} \mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top + \dots\right]^{-1}$$

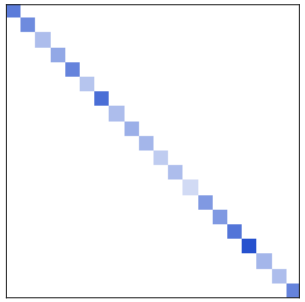
Note: Equivalent to the Fisher for most common log-likelihoods.

Hessian approximation: Matrix factorization

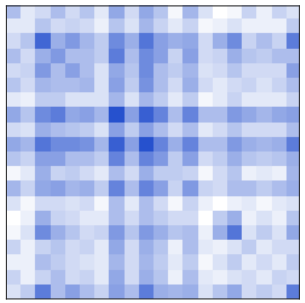
The full covariance matrices $\mathbf{\Sigma}_{\text{Fish.}}$ and $\mathbf{\Sigma}_{\text{GGN}}$ are still quadratic in the number of parameters P, making them difficult to store and difficult to invert.



Diagonal factorization: Keep the matrix diagonal and ignore the off-diagonal elements. Lightweight but highest tradeoff in approximation fidelity.

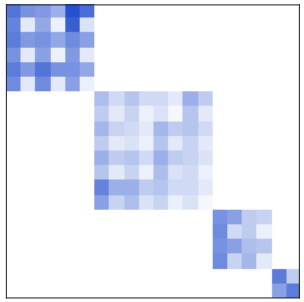


Low-rank factorization: Use a low-rank approximation, such as truncated singular value decomposition (SVD). Can be combined with the diagonal factorization.



Hessian approximation: Kronecker factorization

Block-diagonal factorization: Factorize across neural network layers by approximating diagonal blocks for each layer ℓ, i.e. making the assumption of layer independence.



Kronecker factorization (*Ritter et al., 2018*)**:** The approximate blocks for a single data point can be formulated as a Kronecker product (K-FAC).

$$
\begin{aligned}
-\left[\mathbf{\Sigma}_{\text{GGN}}^{(\ell)}\right]^{-1} &= \sum_{i=1}^{n} \left[\mathbf{A}_i^{(\ell)} \otimes \mathbf{B}_i^{(\ell)}\right] \\
&\approx \left[\sum_{i=1}^{n} \mathbf{A}_i^{(\ell)}\right] \otimes \left[\sum_{i=1}^{n} \mathbf{B}_i^{(\ell)}\right] \triangleq \mathbf{A}^{(\ell)} \otimes \mathbf{B}^{(\ell)}
\end{aligned}
$$

Note: $\mathbf{A}^{(\ell)} \in \mathbb{R}^{D_{\text{out}}^{(\ell)} \times D_{\text{out}}^{(\ell)}}$, $\mathbf{B}^{(\ell)} \in \mathbb{R}^{D_{\text{in}}^{(\ell)} \times D_{\text{in}}^{(\ell)}}$ and both are pos. semi. definite.

Hessian approximation: Predictive misspecification

Misspecification? Used the GGN but now the predictive is underfitting?

① Data $\mathcal{D}$ and model

Prior: $p\left(\begin{bmatrix} w \\ b \end{bmatrix}\right) = \mathcal{N}(\mathbf{0}, \mathbf{I}_2)$

Model:

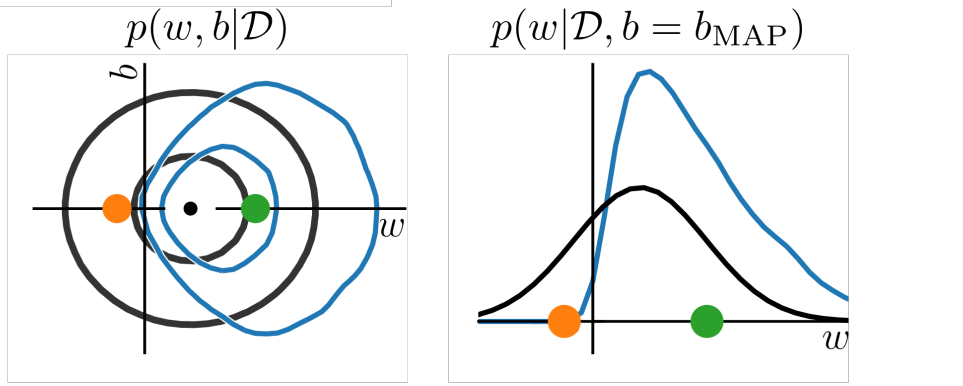
$f(x; w, b) = 5 \tanh(wx + b)$

Likelihood:

$p([y = 1] \mid f) = \texttt{sigmoid}(f)$

The BNN underfits because some samples can give extremely wrong predictions (e.g. sample shown in orange). How do we correct this?

② Posterior inference



③ Posterior predictive distribution

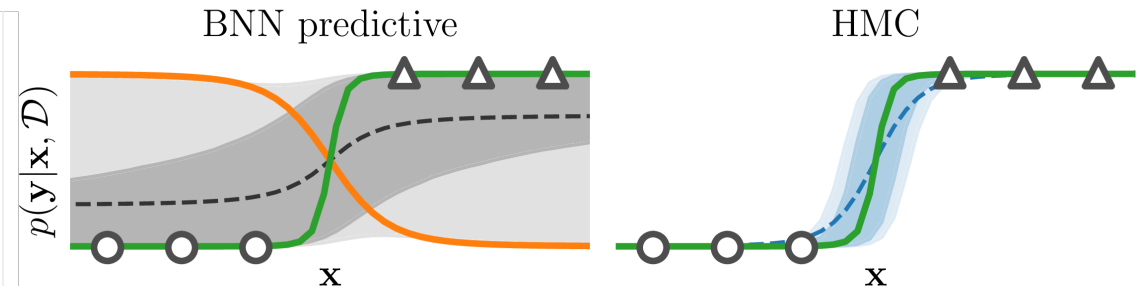


Figure: Fig. 3 of (*Immer et al., 2021a*)

Linearized Laplace: Linearization of $f_{\boldsymbol{\theta}}$

Recall that $\hat{\boldsymbol{\Sigma}}$ depends on the Hessian of the log-**posterior dist.** w.r.t. $\boldsymbol{\theta}$.

$$\hat{\boldsymbol{\Sigma}} = -\left[\nabla^2_{\boldsymbol{\theta}} \log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta}) + \nabla^2_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta})\right]^{-1}_{\hat{\boldsymbol{\theta}}}$$

Focus on the log-**likelihood**. Let $\mathcal{H}_{\boldsymbol{\theta}}(\boldsymbol{x}) \triangleq \left[\nabla^2_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(x)\right]$ denote Hessian.

$$\nabla_{\boldsymbol{\theta}} \log p(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}) = \mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top \left[\nabla_f \log p(y_i \mid f)\right]$$

$$\nabla^2_{\boldsymbol{\theta}} \log p(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}) = \mathcal{H}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top \left[\nabla_f \log p(y_i \mid f)\right] \underbrace{-\mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \left[\nabla^2_f p(y_i \mid f)\right]_{\hat{f}} \mathcal{J}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top}_{\text{GGN approximation!}}$$

Linearized Laplace: Linearization of $f_{\boldsymbol{\theta}}$

GGN approx. makes the assumption that $\mathcal{H}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top \left[\nabla_f \log p(y_i \mid f)\right]$ is zero.

Condition 1: The *residual* $\left[\nabla_f \log p(y_i \mid f)\right]$ vanishes for all data points. This is not desired, as it would indicate overfitting, nor is it very realistic.

Condition 2: The Hessian $\mathcal{H}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ vanishes for all input points. This is true for linear networks, and we can enforce it by linearizing our network!

Local linearization of $f_{\boldsymbol{\theta}}$ (*Immer et al., 2021a*)**:** Applying GGN approx. to the Hessian of the likelihood turns underlying model from BNN to GLM.

$$f_{\boldsymbol{\theta}}^{\text{lin.}}(\boldsymbol{x}) = f_{\boldsymbol{\theta}}(\boldsymbol{x}) + (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\mathcal{J}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x})$$

Linearized Laplace: GP formulation

Function-space equivalence (*Immer et al., 2021a*): This linearized model in weight-space is equivalent to a Gaussian process (GP) in function-space.

$$
\begin{aligned}
m(\boldsymbol{x}) &= \mathop{\mathbb{E}}_{\boldsymbol{\theta}}\left[f_{\boldsymbol{\theta}}^{\text{lin.}}(\boldsymbol{x})\right] = f_{\hat{\boldsymbol{\theta}}}^{\text{lin.}}(\boldsymbol{x}) \\
k(\boldsymbol{x}, \boldsymbol{x}') &= \mathrm{Cov}_{\boldsymbol{\theta}}\left[f_{\boldsymbol{\theta}}^{\text{lin.}}(\boldsymbol{x}),\, f_{\boldsymbol{\theta}}^{\text{lin.}}(\boldsymbol{x}')\right] \\
&= \mathcal{J}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}) \boldsymbol{\Sigma}_{\text{GGN}} \mathcal{J}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x})^{\top}
\end{aligned}
$$

Function-space predictive (*Immer et al., 2021a*): Leads to a closed form for sampling from the network. Use Monte-Carlo simulation for predictive.

$$
\begin{aligned}
p(f^{\star} \mid \boldsymbol{x}^{\star}, \boldsymbol{x}_{1:n}, y_{1:n}) &= \mathcal{N}(f^{\star}; f_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}^{\star}), \mathcal{J}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}^{\star}) \boldsymbol{\Sigma}_{\text{GGN}} \mathcal{J}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}^{\star})^{\top}) \\
p(y^{\star} \mid \boldsymbol{x}^{\star}, \boldsymbol{x}_{1:n}, y_{1:n}) &= \mathop{\mathbb{E}}_{f^{\star}}\left[p(y^{\star} | f^{\star})\right]
\end{aligned}
$$

Linearized Laplace: Demonstration

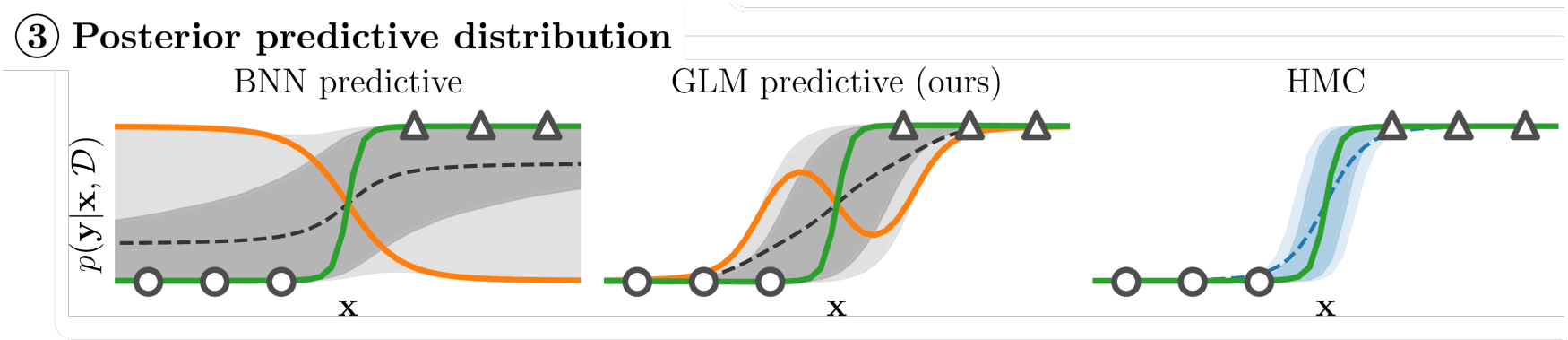


Figure: Using GLM as the underlying model. (Fig. 3 of *Immer et al., 2021a*)

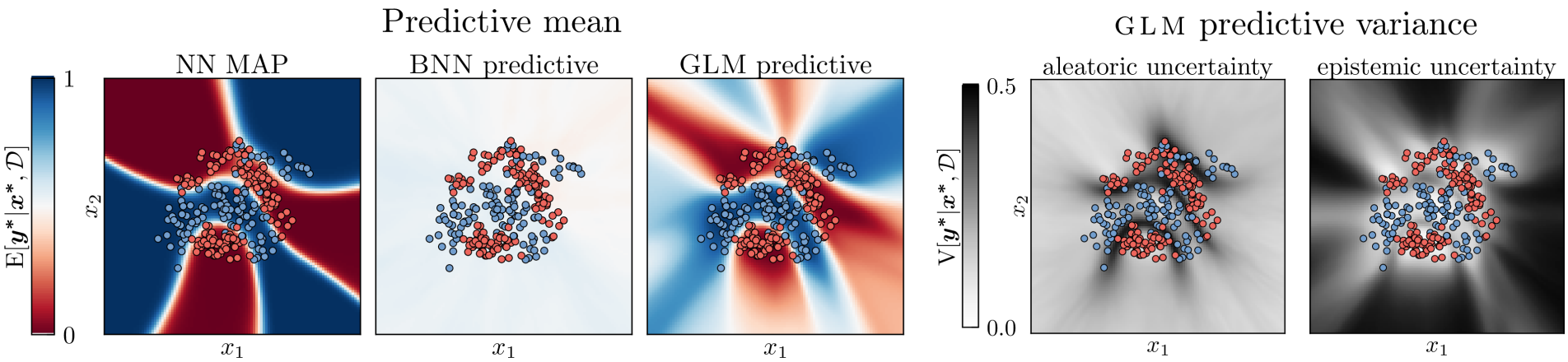


Figure: Mean and uncertainty on banana dataset. (Fig. 4 of *Immer et al., 2021a*)

Linearized Laplace: Model selection

Marginal likelihood (*Immer et al., 2021b*) **:** Likelihood of model $\mathcal{M}$.
Overly simple or complex models have low-probability (Occam's razor).

$$\log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \mathcal{M}) = \int \log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta} \mid \mathcal{M})\, d\boldsymbol{\theta}$$

$$\approx \log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \hat{\boldsymbol{\theta}}) + \log p(\hat{\boldsymbol{\theta}} \mid \mathcal{M}) - \frac{1}{2} \log \left| \frac{1}{2\pi} \boldsymbol{\Sigma}^{-1} \right|$$

Note: Model $\mathcal{M}$ can be choice of architecture, hyper-parameters, etc.

Linearized Laplace: Model selection

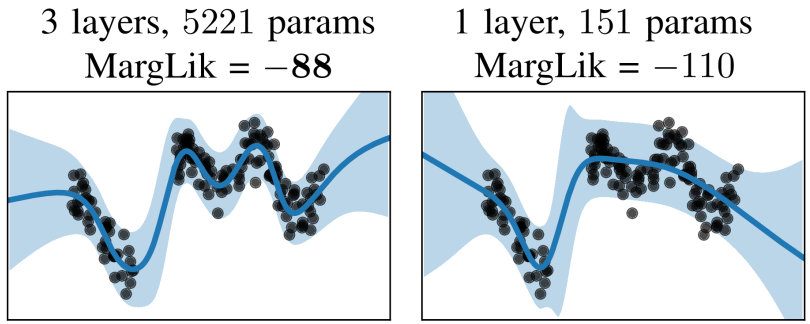


Figure: Marginal likelihood on toy regression. (Fig. 1 of *Immer et al., 2021b*)

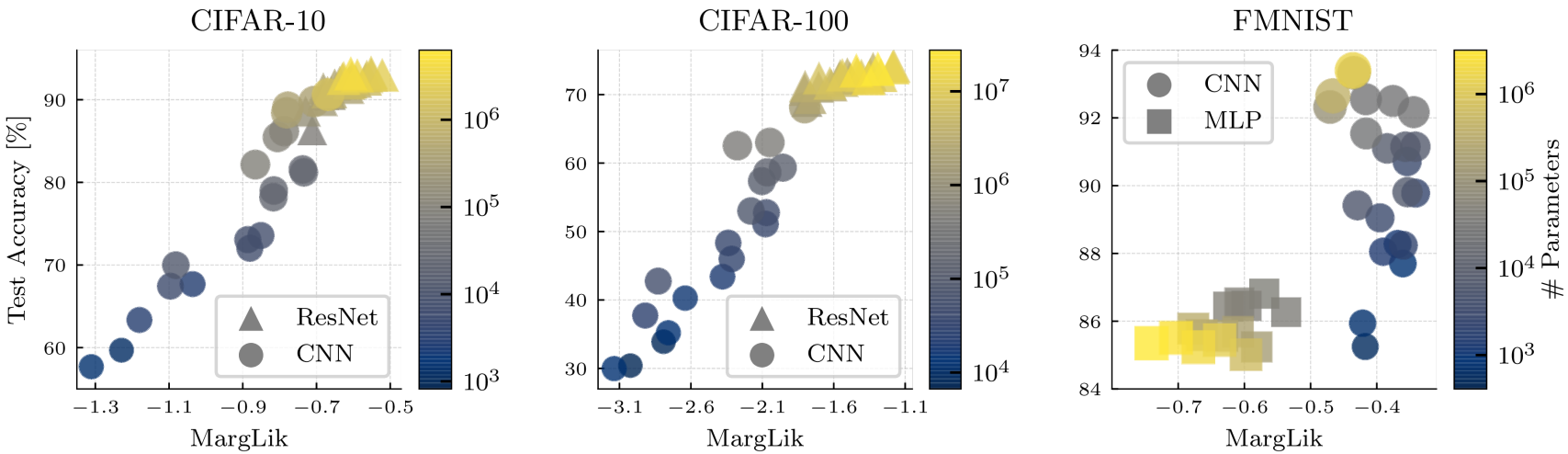


Figure: Marginal likelihood and test accuracy (Fig. 2 of *Immer et al., 2021b*)

Linearized Laplace: Hyperparam. optimization

Hyperparam. optimization (*Immer et al., 2021*): Diff. hyperparams $\mathcal{M}^{\partial}$. Alternate between optimizing params. $\boldsymbol{\theta}$ and hyperparams. $\mathcal{M}^{\partial}$ online.

$$\mathcal{M}^{\partial}_{t+1} \leftarrow \mathcal{M}^{\partial}_{t} + \gamma \nabla_{\mathcal{M}^{\partial}} \log q(y_{1:n} \mid \boldsymbol{x}_{1:n}, \mathcal{M})$$

Note: a.k.a. empirical Bayes or type-II maximum likelihood estimation.

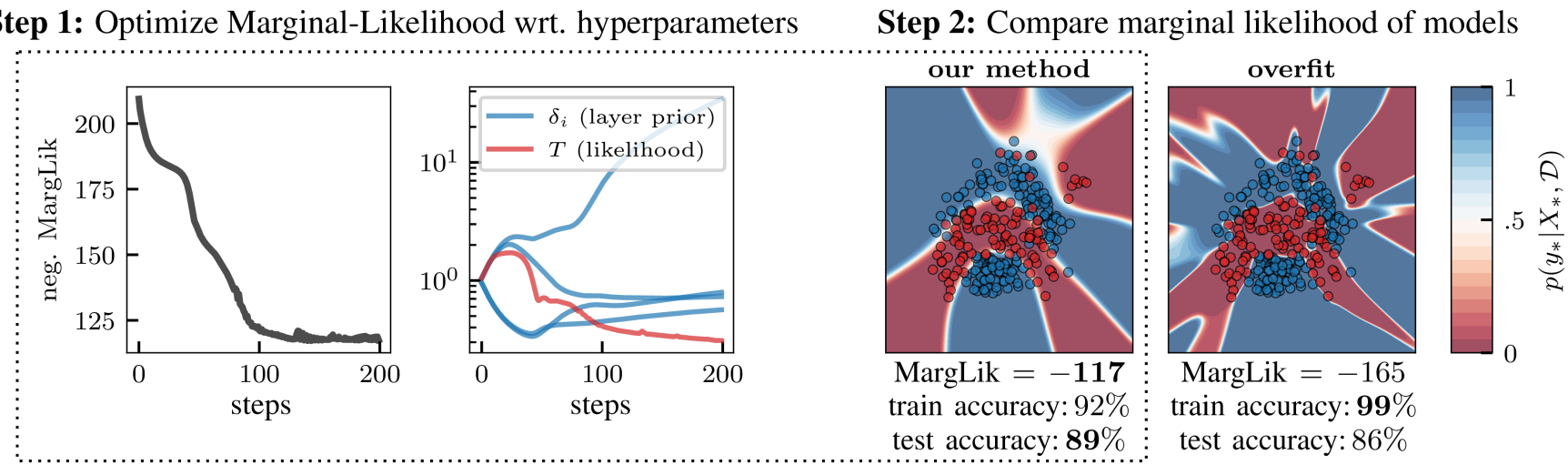


Figure: Optimizing marg. lik. in banana dataset. (Fig. 3 of *Immer et al., 2021b*)

Linearized Laplace: PyTorch package

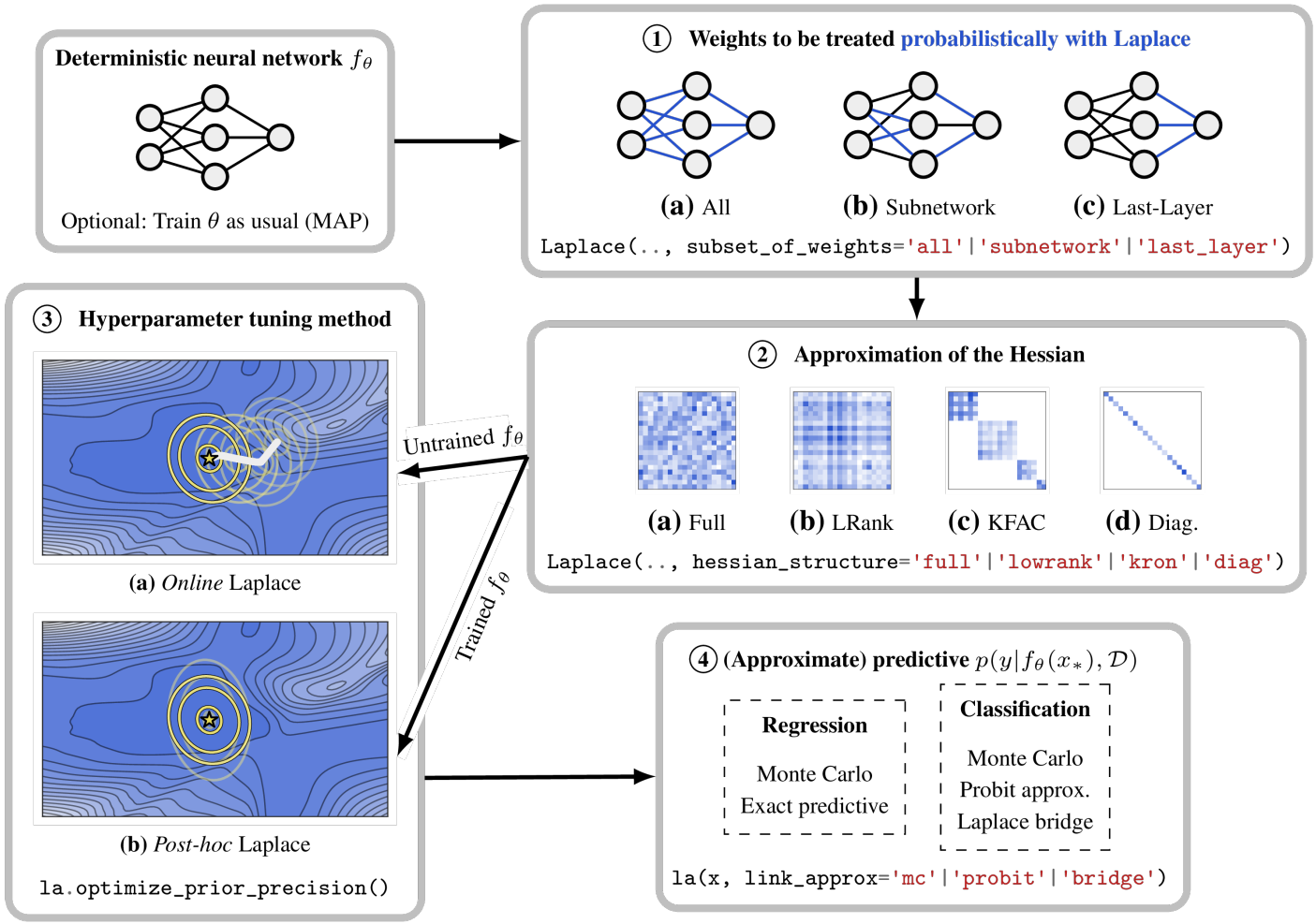


Figure: Overview of `laplace-torch`. (Fig. 2 of *Daxberger et al., 2021*)

Laplace-approximated NAMs: Introduction

Focus on tabular data, d columns of numer. or categ. features $x_1, \ldots, x_d$.

Neural additive models (*Agarwal et al., 2021*): Handle input columns indep. in separate sub-networks to observe the response as we vary inputs.

$$\mathbb{E}[g(y) \mid \boldsymbol{x}] = f_{\boldsymbol{\theta}}(\boldsymbol{x}) \triangleq f^{(1)}_{\boldsymbol{\theta}_1}(x_1) + f^{(2)}_{\boldsymbol{\theta}_2}(x_2) + \cdots + f^{(d)}_{\boldsymbol{\theta}_d}(x_d)$$

Laplace-approximated NAMs (*Bouchiat et al., 2024*): Swap the point estimates with Bayesian neural networks and use linearized Laplace approx.

$$f^{\text{lin.}}_{\boldsymbol{\theta}}(\boldsymbol{x}) \triangleq f^{(1),\,\text{lin.}}_{\boldsymbol{\theta}_1}(x_1) + f^{(2),\,\text{lin.}}_{\boldsymbol{\theta}_2}(x_2) + \cdots + f^{(d),\,\text{lin.}}_{\boldsymbol{\theta}_d}(x_d)$$
$$f^{(j),\,\text{lin.}}_{\boldsymbol{\theta}_j}(x_j) \triangleq f^{(j)}_{\hat{\boldsymbol{\theta}}_j}(x_j) + (\boldsymbol{\theta}_j - \hat{\boldsymbol{\theta}}_j)\mathcal{J}_{\hat{\boldsymbol{\theta}}_j}(x_j)$$

Laplace-approximated NAMs: Introduction

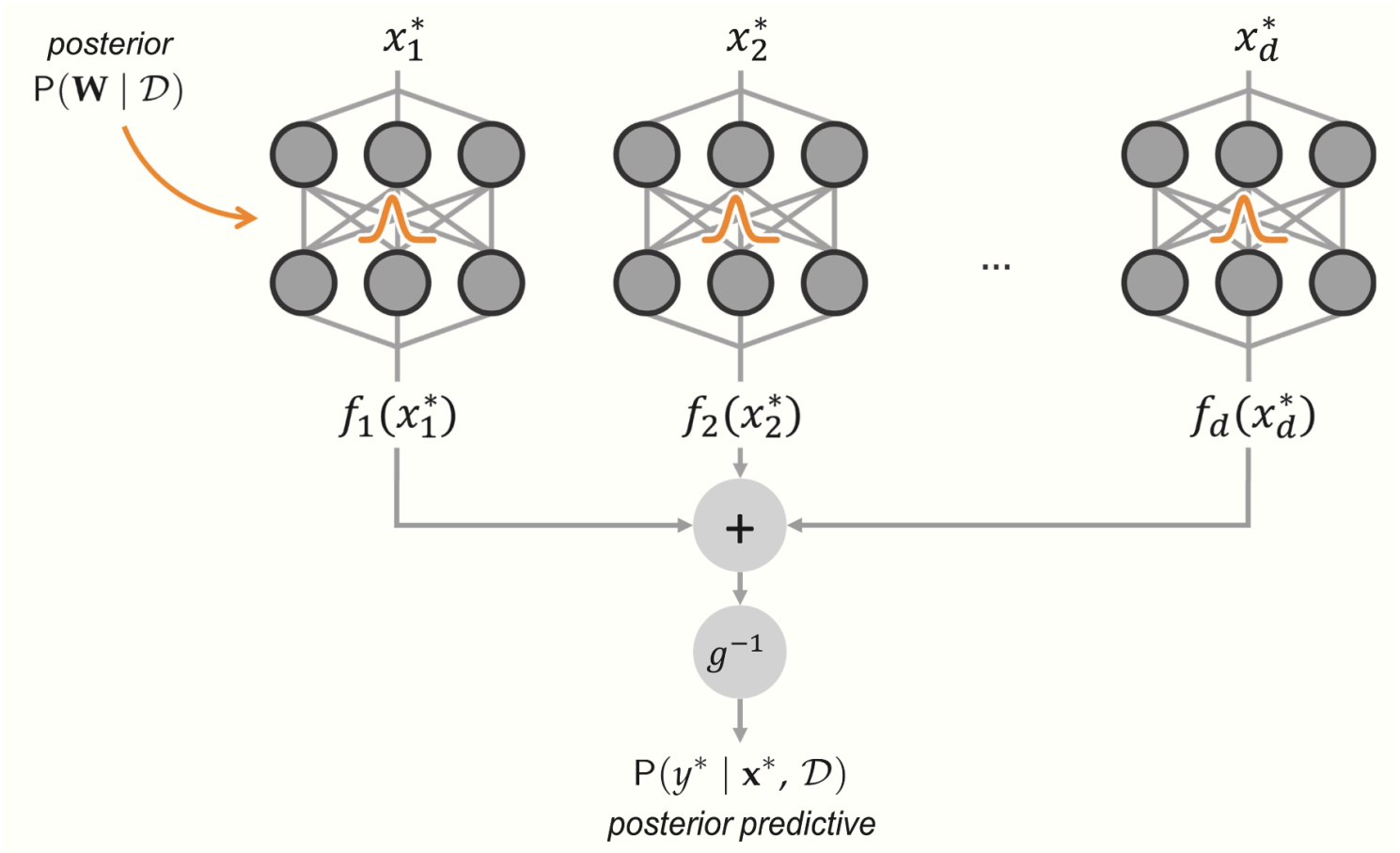


Figure: Diagram of LA-NAM architecture in (*Bouchiat et al., 2024*).

Laplace-approximated NAMs: Introduction

Laplace-approximated NAMs (*Bouchiat et al., 2024*): The independence of the subnetworks leads to factorized block-diagonal posterior covariance.

$$q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\text{GGN}}), \quad \boldsymbol{\Sigma}_{\text{GGN}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\text{GGN}}^{(1)} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \boldsymbol{\Sigma}_{\text{GGN}}^{(d)} \end{bmatrix}$$

Block factorization leads to factorized marg. lik. (*Immer et al., 2023*).

$$\begin{aligned} \log p(y_{1:n} \mid \boldsymbol{x}_{1:n}, \mathcal{M}) &\approx \log p(y_{1:n}, \hat{\boldsymbol{\theta}} \mid \boldsymbol{x}_{1:n}, \mathcal{M}) + \frac{1}{2} \left| \frac{1}{2\pi} \boldsymbol{\Sigma}_{\text{Full}}^{-1} \right| \\ &\geq \log p(y_{1:n}, \hat{\boldsymbol{\theta}} \mid \boldsymbol{x}_{1:n}, \mathcal{M}) + \frac{1}{2} \sum_j \left| \frac{1}{2\pi} \boldsymbol{\Sigma}_{\text{GGN}}^{(j)} \right| \end{aligned}$$

Laplace-approximated NAMs: Toy example

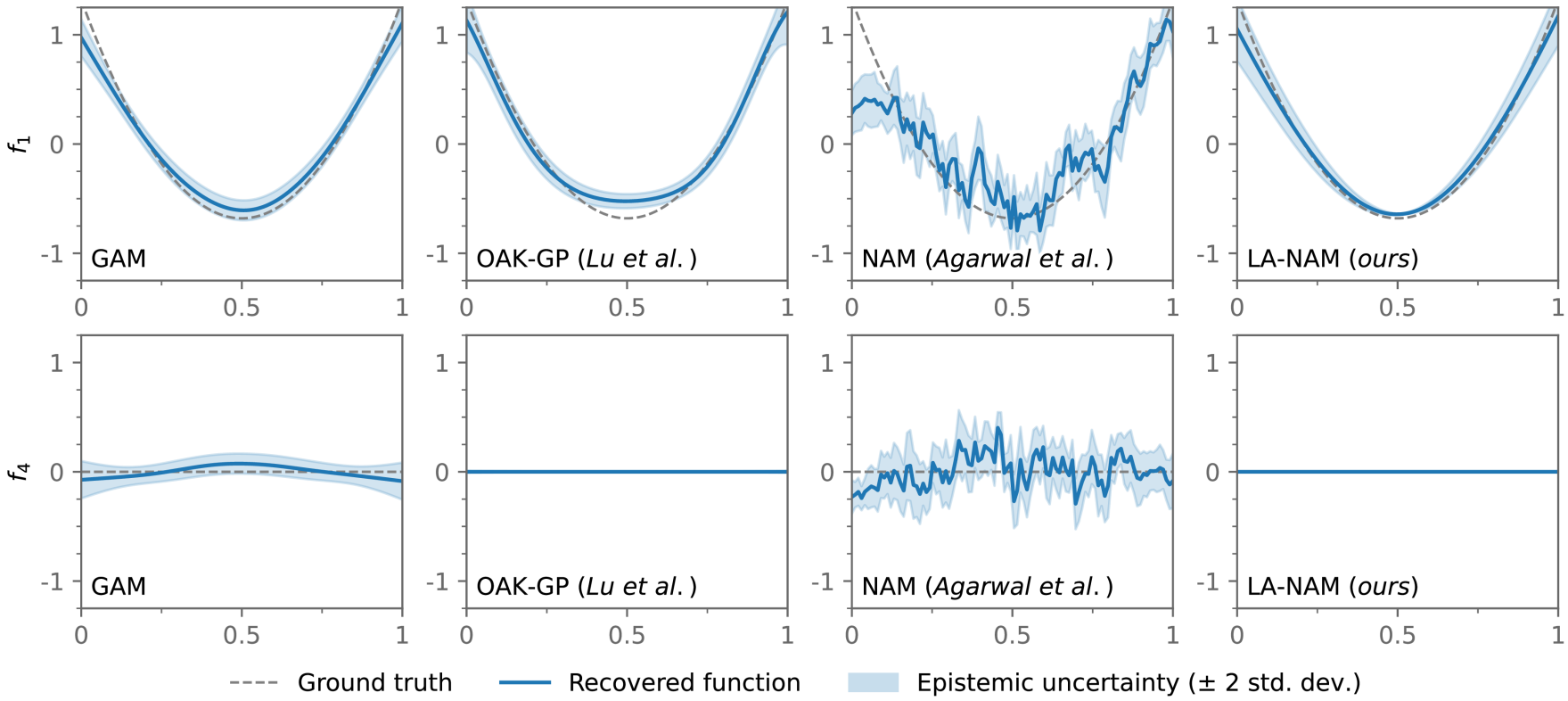


Figure: Demonstration on toy dataset. (Fig. 1 of *Bouchiat et al., 2024*)

Laplace-approximated NAMs: MIMIC-III dataset

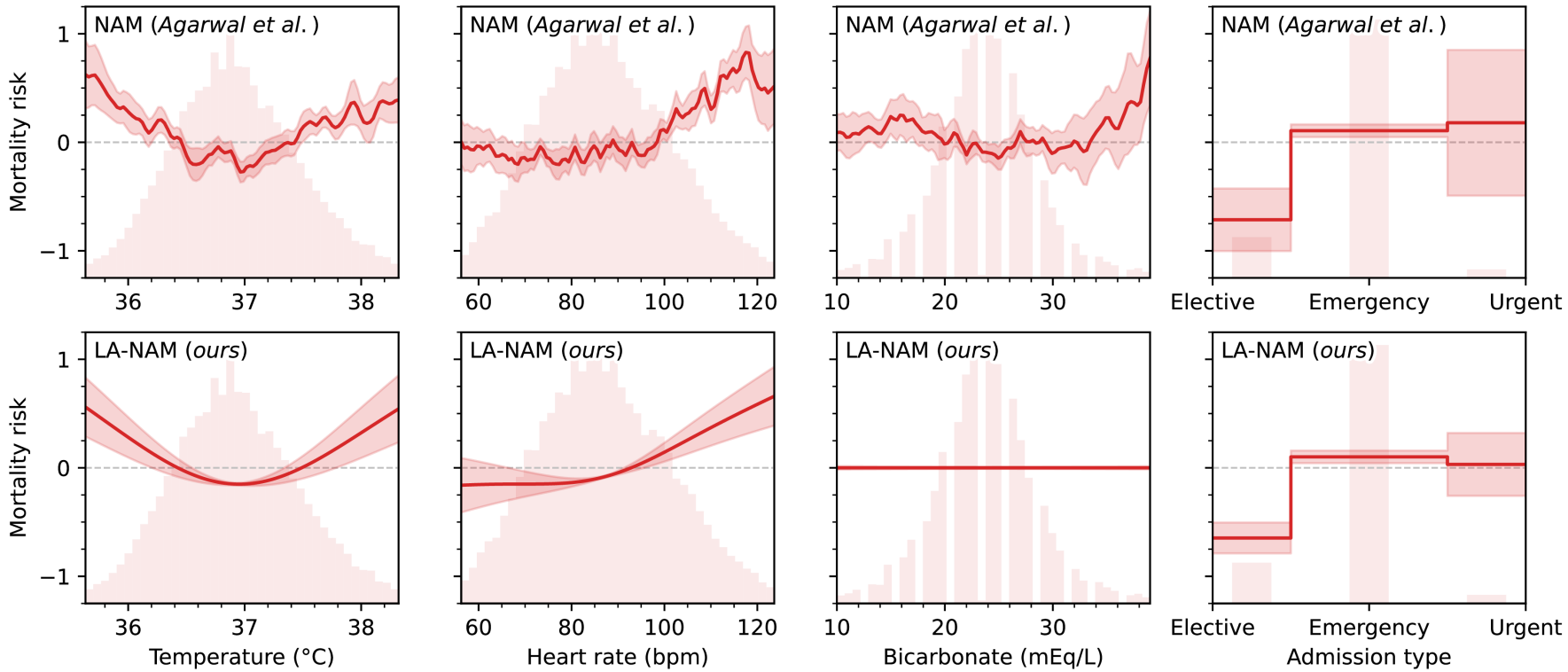
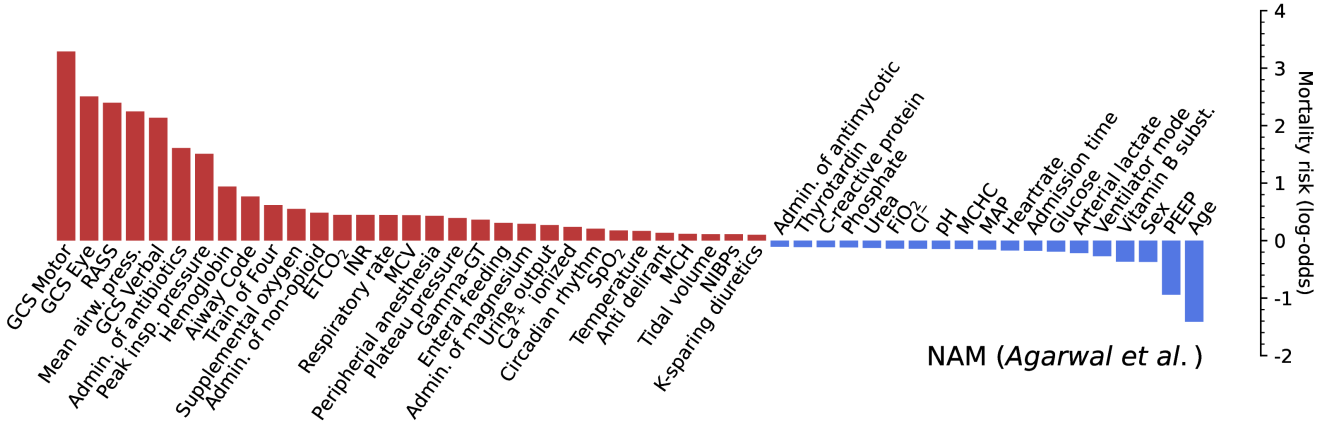
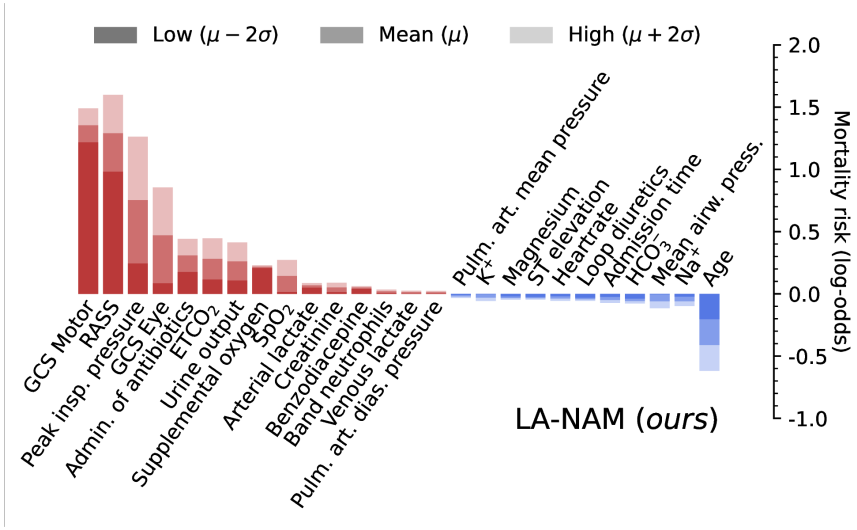


Figure: Application to MIMIC-III ICU mortality. (Fig. 2 of *Bouchiat et al., 2024*)

Laplace-approximated NAMs: MIMIC-III dataset



(a) Local predictions of NAM. (Fig. 3 of *Bouchiat et al., 2024*)



(b) Local predictions of LA-NAM. (Fig. 4 of *Bouchiat et al., 2024*)

Laplace-approximated NAMs: Feature interaction

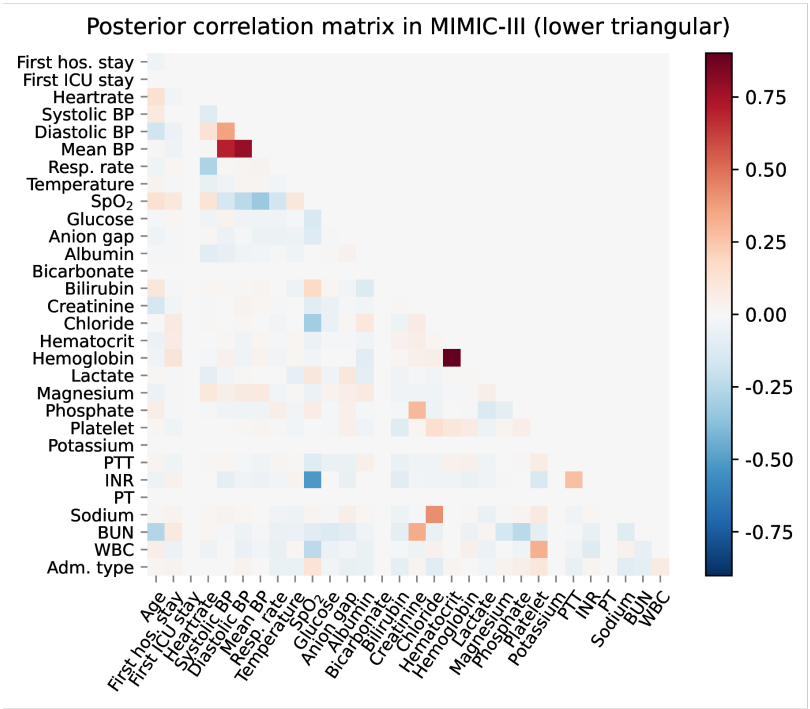
Laplace-approximated NAMs (*Bouchiat et al., 2024*): The subnetworks are not necessarily mutually independent in the true posterior distribution.

Determine mut. inf. using scalar marginal variances $\sigma_j^2, \sigma_{j'}^2$ and covariance $\sigma_{j,j'}^2$ with a last-layer Laplace approximation on output weights θ_j and $\theta_{j'}$.

$$\begin{aligned} \mathsf{I}(\boldsymbol{\theta}_j; \boldsymbol{\theta}_{j'}) &= \mathsf{H}(\boldsymbol{\theta}_j) + \mathsf{H}(\boldsymbol{\theta}_{j'}) - \mathsf{H}(\boldsymbol{\theta}_j, \boldsymbol{\theta}_{j'}) \\ &\approx \tfrac{1}{2} \log \left[\sigma_j^2 \sigma_{j'}^2 (\sigma_j^2 \sigma_{j'}^2 - \sigma_{j,j'}^2)^{-1} \right] \\ &= \tfrac{1}{2} \log \left[1 - \mathsf{Corr}(\theta_j, \theta_{j'})^2 \right]^{-1} \end{aligned}$$

Select top-k, append their $f_{\boldsymbol{\theta}_{j,j'}}^{(j,j')}(x_j, x_{j'})$ interaction subnetworks, fine-tune.

Laplace-approximated NAMs: Feature interaction



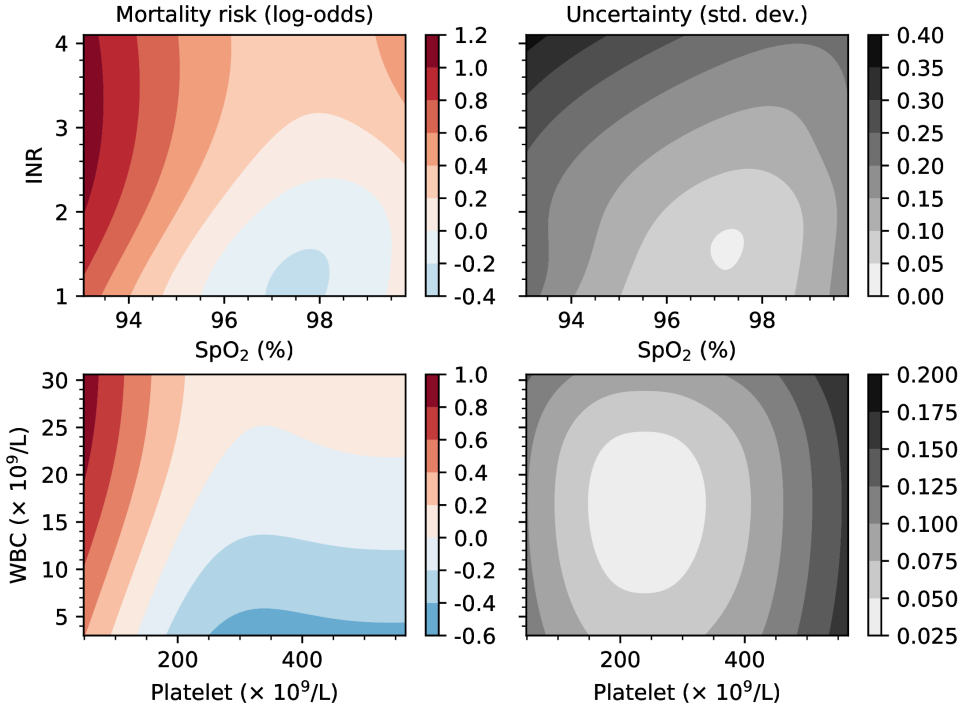


Figure: Feature interaction of LA-NAM. (Fig. 5 of *Bouchiat et al., 2024*)

References (i)

- P.S. Laplace, 1774, *"Mémoire sur la probabilité des causes par les évènements"* in Mémoire de l'Académie Royale des Sciences de Paris.
- D.J.C. MacKay, 1992, *"A Practical Bayesian Framework for Backpropagation Networks"* in Neural Computation.
- Kendall et al., 2017, *"What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?"* in NeurIPS 2017.
- Ritter et al., 2018, *"A Scalable Laplace Approximation for Neural Networks"* in ICLR 2018.
- Immer et al., 2021a, *"Improving predictions of Bayesian neural nets via local linearization"* in Artificial Intelligence and Statistics 2021.

References (ii)

- Immer et al., 2021b, *"Scalable Marginal Likelihood Estimation for Model Selection in Deep Learning"* in ICML 2021.
- Agarwal et al., 2021, *"Neural Additive Models: Interpretable Machine Learning with Neural Nets"* in NeurIPS 2021.
- Daxberger et al., 2021, *"Laplace Redux – Effortless Bayesian Deep Learning"* in NeurIPS 2021.
- Immer et al., 2023, *"Stochastic Marginal Likelihood Gradients using Neural Tangent Kernels"* in ICML 2023.
- Bouchiat et al., 2024, *"Improving Neural Additive Models with Bayesian Principles"* in ICML 2024.