Aalto University and ELLIS unit Helsinki Seminar on Advances in Probabilistic ML

March 17th, 2022

Bayesian Deep Learning with Linearised Neural Networks

Javier Antorán

UNIVERSITY OF CAMBRIDGE

Talk outline

1. **Preliminaries**: probabilistic inference in neural networks and the linearised Laplace method

2. **Methodological advancement**: adapting the linearised Laplace model evidence for modern deep learning

3. **Case study**: applying linearised Laplace to design a prior for x-ray image tomographic reconstruction

Preliminaries

Preliminaries: open problems in deep learning

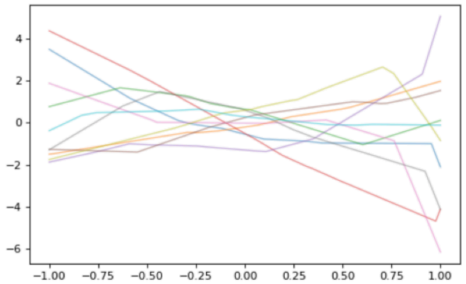
Overconfidence

Training on CIFAR10 – Test on SVHN

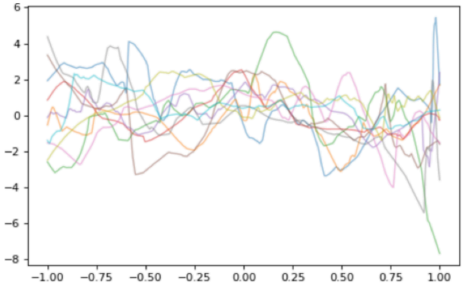
https://vitalab.github.io/article/2019/07/11/overconfident.html

Model Selection

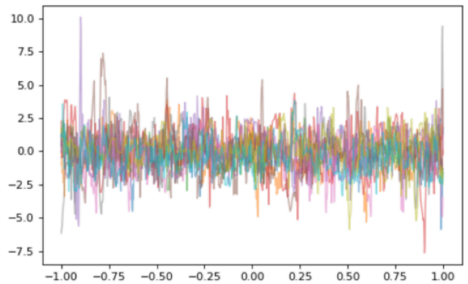
1 Hidden Layer



5 Hidden Layer

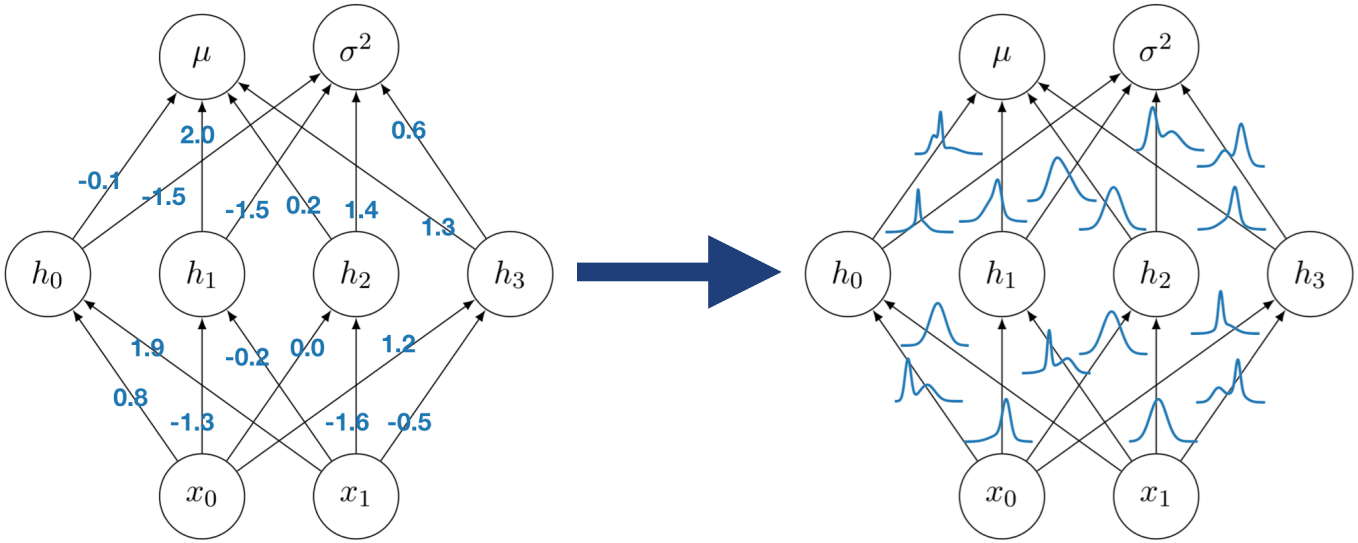


20 Hidden Layer



Preliminaries: probabilistic inference in NNs

1. Place a prior distribution $\pi(\theta)$ over NN parameters.
2. Define some likelihood function $p(y \mid f(\theta, x))$ to characterise the agreement of the NN function $f(\theta, \cdot)$ with the observations (y, x)
3. Update the weight distribution using Bayes' rule

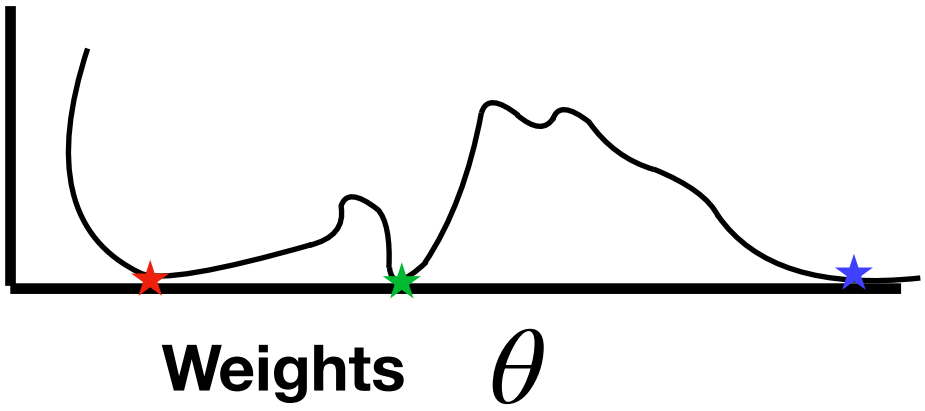


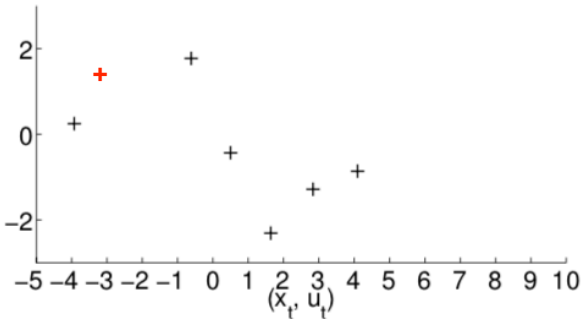
$$\tilde{\theta} \in \operatorname{argmax}_\theta \underbrace{\log p(y \mid f(\theta, x))}_{\text{data fit}} + \underbrace{\log \pi(\theta)}_{\text{regulariser}}$$

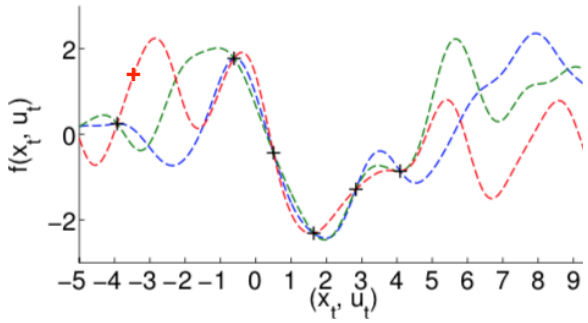
$$p(\theta \mid x, y) = \frac{p(y \mid f(\theta, x))\pi(\theta)}{p(y \mid x)}$$

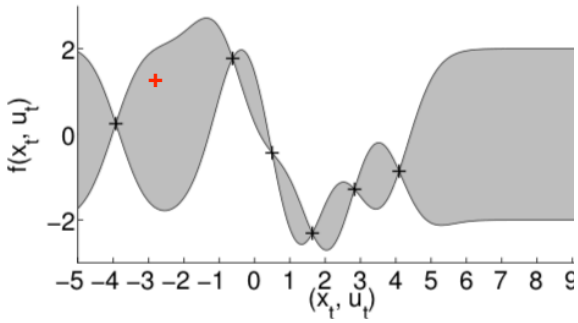
Preliminaries: uncertainty estimation

$$\mathscr{L}_f(\theta) = -\underbrace{\log p(y \mid f(\theta, x))}_{\text{data fit}} - \underbrace{\log \pi(\theta)}_{\text{regulariser}}$$









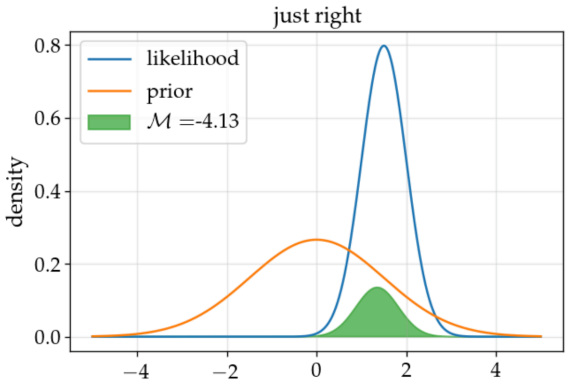
$$p(\theta \mid y, x) = \frac{1}{\exp(\mathscr{M})} \exp(-\mathscr{L}_f(\theta)) \qquad f(\theta, \cdot), \ \theta \sim p(\theta \mid y, x)$$

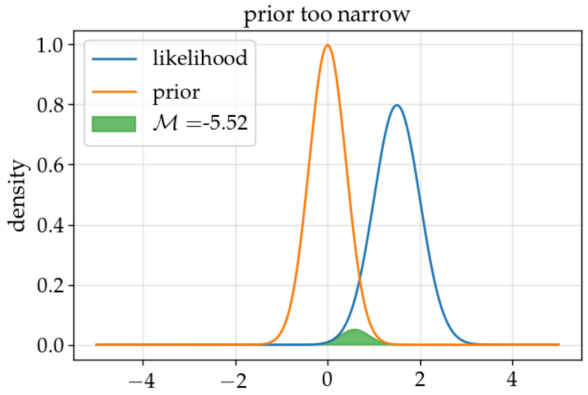
Preliminaries: model selection

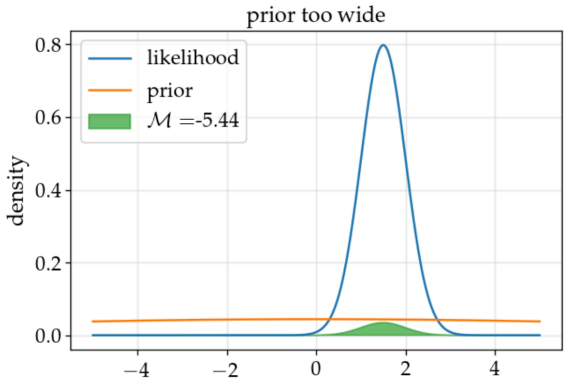
$$\mathscr{L}_f(\theta) = -\log p(y \mid f(\theta, x)) - \log \pi(\theta) \quad p(\theta \mid y, x) = \frac{1}{\exp(\mathscr{M})} \exp(-\mathscr{L}_f(\theta))$$

The normalisation constant, $\mathscr{M}$, is the ***marginal likelihood***, or ***model evidence***. It is the probability that our observations where generated by our prior. It provides an objective for hyperparameter selection without the need for validation data.

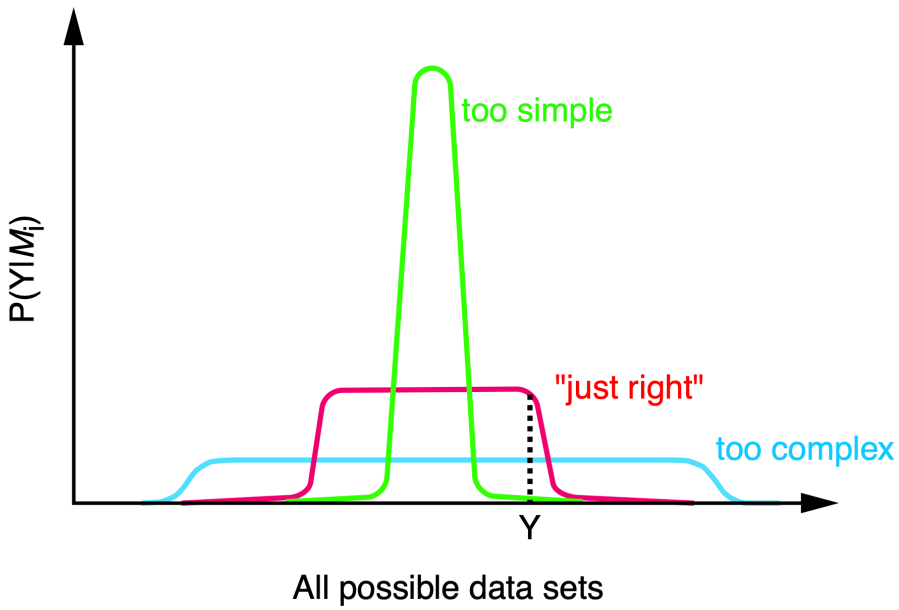
$$\mathscr{M} = \log p(y \mid x) = \log \int p(y \mid f(\theta, x)) d\pi = \log \int \exp(-\mathscr{L}_f(\theta)) d\nu$$

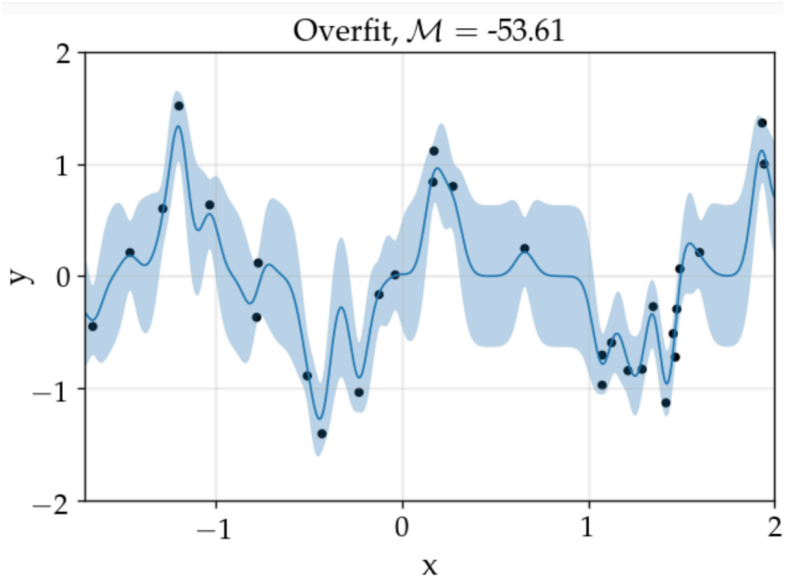


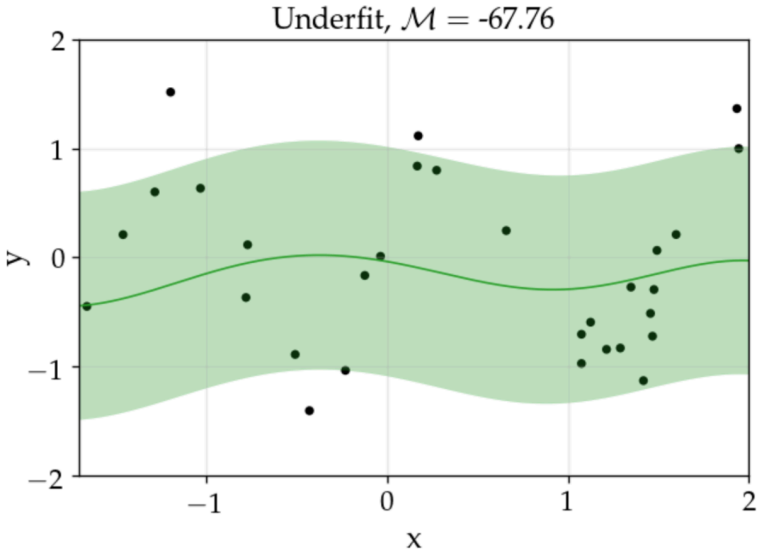


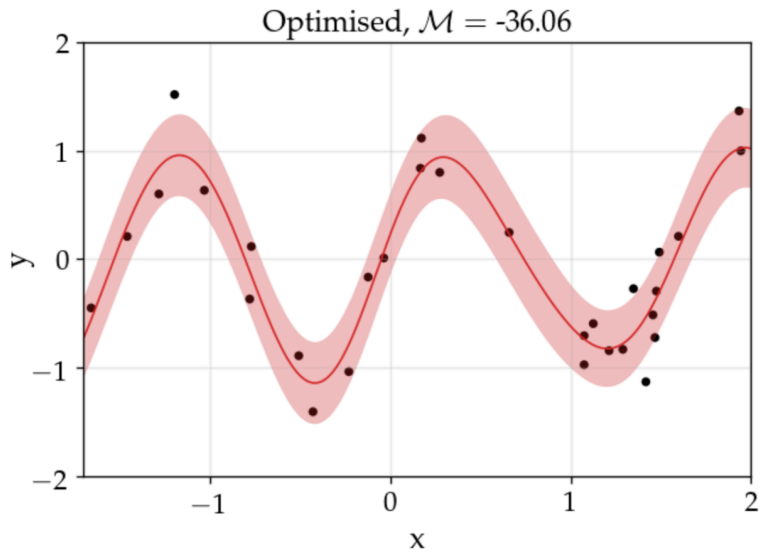


Preliminaries: automatic Occam's razor





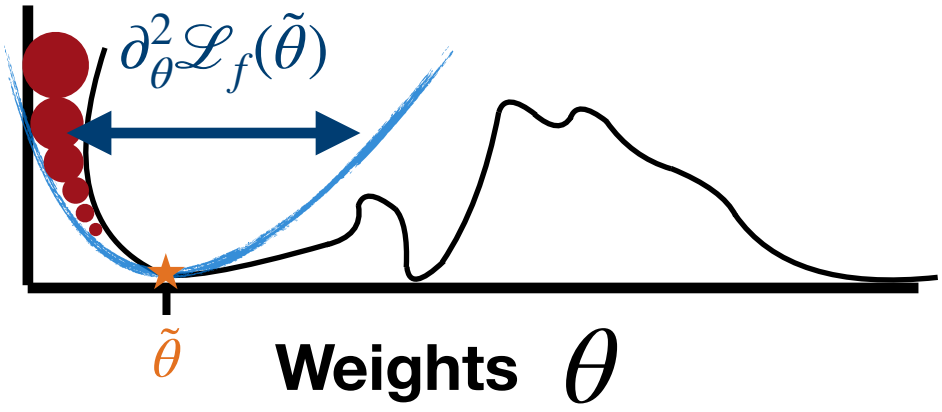




Preliminaries: the Laplace approximation

$$\mathscr{L}_f(\theta) = -\log p(y \,|\, f(\theta, x)) - \log \pi(\theta)$$

$$\mathscr{M} = \log \int \exp(-\mathscr{L}_f(\theta)) d\nu$$



For NNs this integral is intractable

Idea: Find a mode of $\mathscr{L}_f$: $\tilde{\theta}$ and perform 2-order Taylor expansion

$$\mathscr{G}_f(\theta) = \mathscr{L}_f(\tilde{\theta}) + ||\theta - \tilde{\theta}||^2_{\partial^2_\theta \mathscr{L}_f(\tilde{\theta})}$$

By inspection, $\exp(-\mathscr{G}_{f,\tilde{\theta}}(\theta))$ is proportional to $\mathcal{N}(\tilde{\theta}, (\partial^2_\theta \mathscr{L}_f(\tilde{\theta}))^{-1})$ where

$$\partial^2_\theta \mathscr{L}_f(\tilde{\theta})) = \partial^2_\theta \log p(y \,|\, f(\tilde{\theta}, x)) + \partial^2_\theta \log \pi(\tilde{\theta}) \qquad \pi(\theta) \to \mathcal{N}(\theta;\ 0, \Lambda^{-1})$$

Issue: A lot of mass falls in low density region, leading to bad predictions

Preliminaries: the linearised Laplace method

- A Gaussian can be a very poor approximation to the NN posterior
- But it is a very good posterior for a linear model (in some cases exact)
  - $h(\theta, \cdot) = f(\tilde{\theta}, \cdot) + \partial_\theta f(\tilde{\theta}, \cdot)(\theta - \tilde{\theta})$

Affine in θ

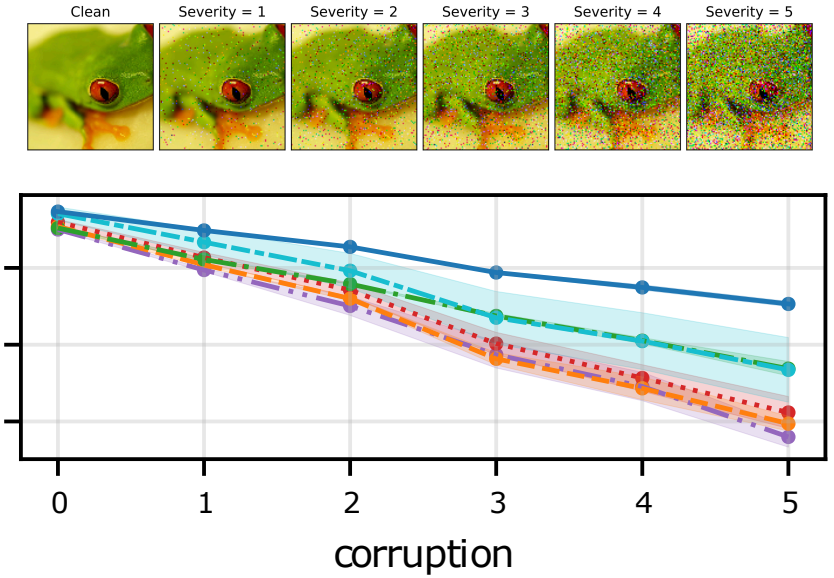
Jacobian acts as basis expansion

- New loss $\mathcal{L}_h(\theta)$ is convex and $\approx \mathcal{G}_h(\theta) = \mathcal{L}_h(\tilde{\theta}) + ||\theta - \tilde{\theta}||^2_{\partial^2_\theta \mathcal{L}_h(\tilde{\theta})}$
- This **conjugate Gaussian-linear model** has:
  - Feature expansion $J(\cdot) = \partial_\theta f(\tilde{\theta}, \cdot)$, Design matrix $H = \partial^2_\theta \mathcal{L}_h(\tilde{\theta})$
  - Closed form predictive posterior and marginal likelihood

[Mackay, 1992]

Some results: image classification under distribution shift

Corrupted CIFAR10 (Ovadia 2019)



Model:
ResNet-18 with **11M** weights

Inference:
Lin Laplace Subnetwork
(Daxberger et. al. 2021)

Baselines:
- MAP
- Diagonal Laplace
- MC Dropout (Gal 2016)
- Deep Ensembles (Lakshminarayanan 2017)
- SWAG (Maddox 2019)

"Bayesian Deep Learning via Subnetwork Inference"

Some questions one might have

- But wait, how did you find a mode of the NN loss to expand around?
  - We didn't, we used **S**GD and hoped for the best
- How did you deal with modern architecture elements, like batchnorm?
  - We used them and hoped for the best
- How did you tune hyperparameters?
  - Cross validation — in fact, the choice of Gaussian prior precision Λ makes a large difference in performance; it controls the size of the errorbars
- What about the model evidence?
  - We could not get it to work, it consistently choose prior precisions that overestimated uncertainty

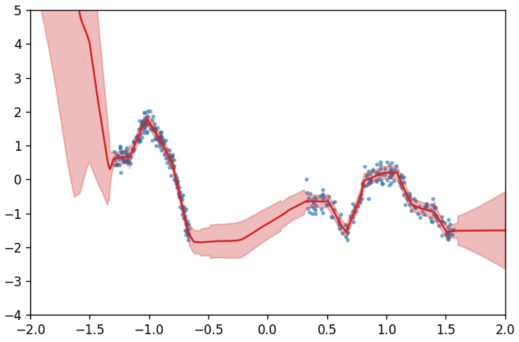
— Why doesn't it work?

Problem illustration

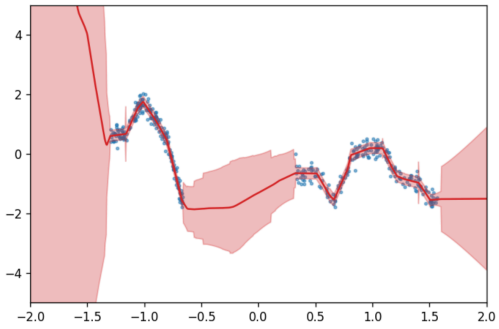
$$\Lambda = \lambda I$$

2 hidden layer, 2600 parameter, MLP with batchnorm

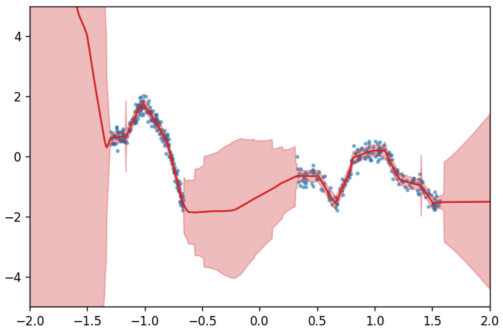
$\lambda = 100$



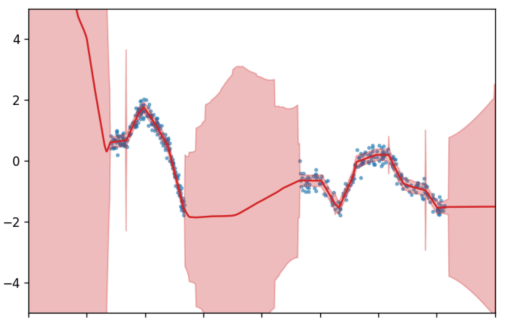
$\lambda = 10$



$\lambda = 5$

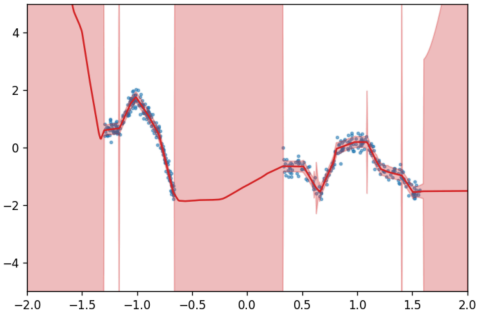


$\lambda = 1$



Largest $\mathcal{M}_{\tilde{\theta}}$

$\lambda = 0.1$



Adapting the linearised Laplace method for modern deep learning

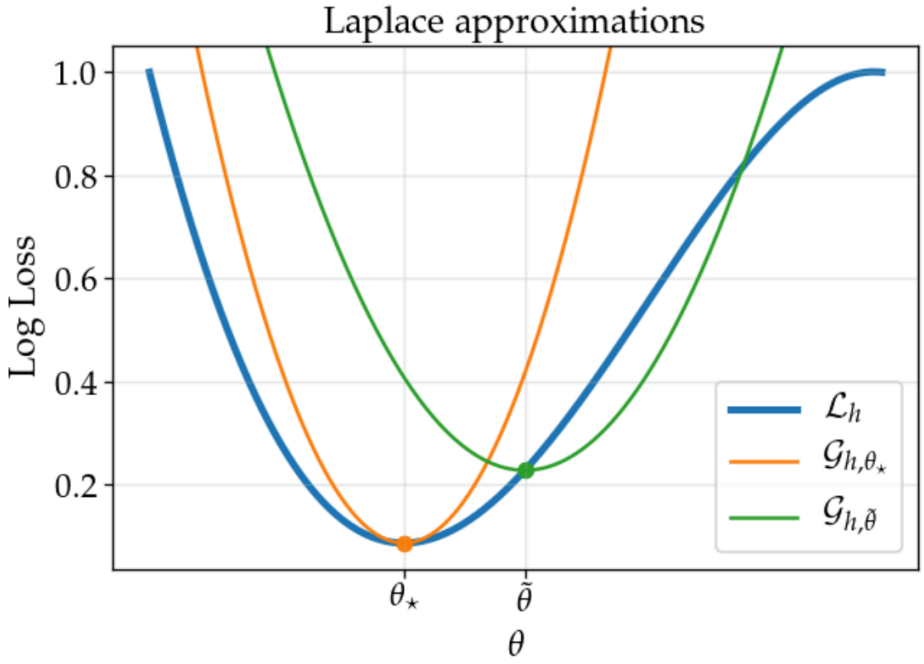
On the difficulty of finding a mode of the loss

- In 1992 Mackay did not use stochastic optimisation, early stopping or normalisation layers
- In modern settings $\tilde{\theta}$ is **not** a stationary point of $\mathscr{L}_f$
- $\partial_\theta \mathscr{L}_f(\tilde{\theta}) \neq 0 \implies \partial_\theta \mathscr{L}_h(\tilde{\theta}) \neq 0$

$$\mathscr{M}_{\tilde{\theta}}(\Lambda) =$$

$$\frac{-1}{2}(||\tilde{\theta}||^2_\Lambda + \text{logdet}(\Lambda^{-1}H + I)) + C$$

Wrong! **Not that wrong?**



The basis function linear model has a well defined optima

- We know that for any regularisation strength Λ, $\mathscr{L}_{h,\Lambda}(\theta)$ is convex
- We know that for any linearisation point θ, $\mathscr{M}_{\theta}(\Lambda)$ is concave

$\implies$ We can find joint stationary point $(\theta_\star, \Lambda_\star)$

$$\theta_\star \in \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_{h,\Lambda_\star}(\theta) \quad \text{and} \quad \Lambda_\star \in \underset{\Lambda}{\operatorname{argmax}}\, \mathcal{M}_{\theta_\star}(\Lambda).$$

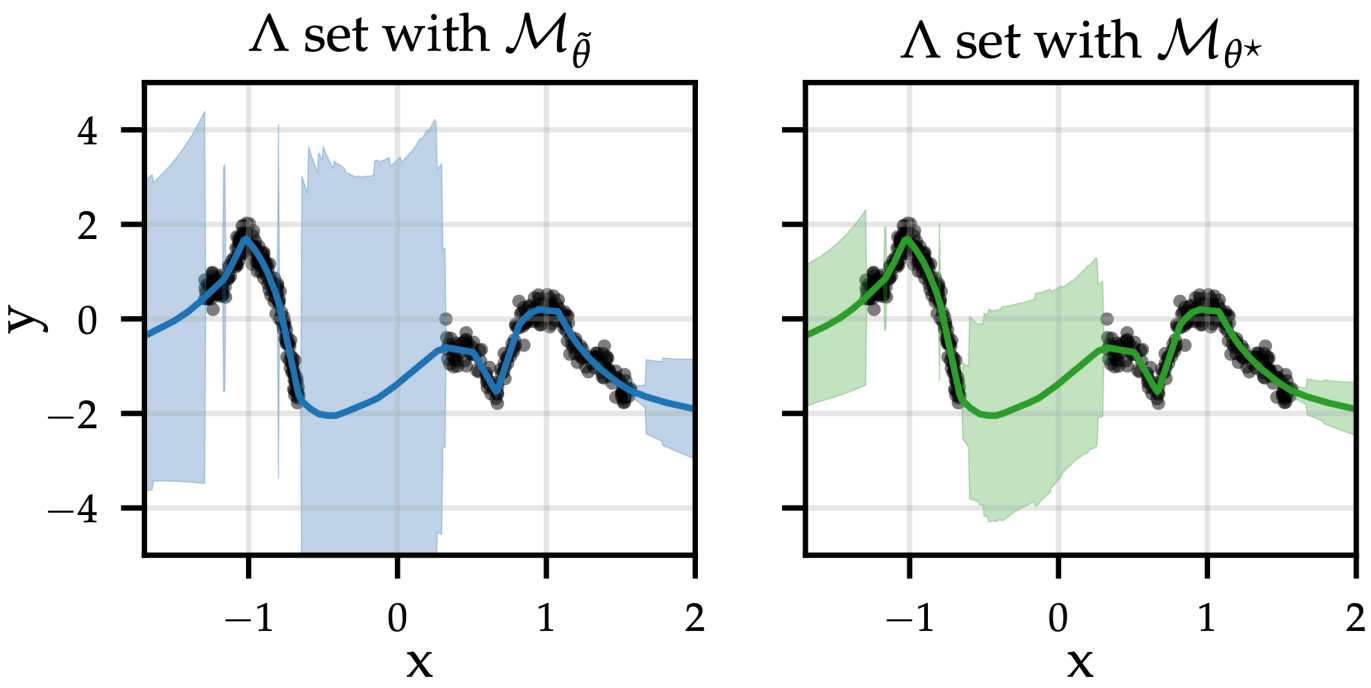
Recommendation 1: Keep $\tilde{\theta}$ as linearisation point but

$$\mathscr{M}_{\tilde{\theta}}(\Lambda) = \frac{-1}{2}(||\tilde{\theta}||^2_\Lambda + \text{logdet}(\Lambda^{-1}H + I)) + C$$

$$\rightarrow$$

$$\mathscr{M}_{\theta_\star}(\Lambda) = \frac{-1}{2}(||\theta_\star||^2_\Lambda + \text{logdet}(\Lambda^{-1}H + I)) + C$$

Following recommendation 1 improves errorbars



Could we find $\tilde{\theta} = \theta_\star$ by optimising our network better?

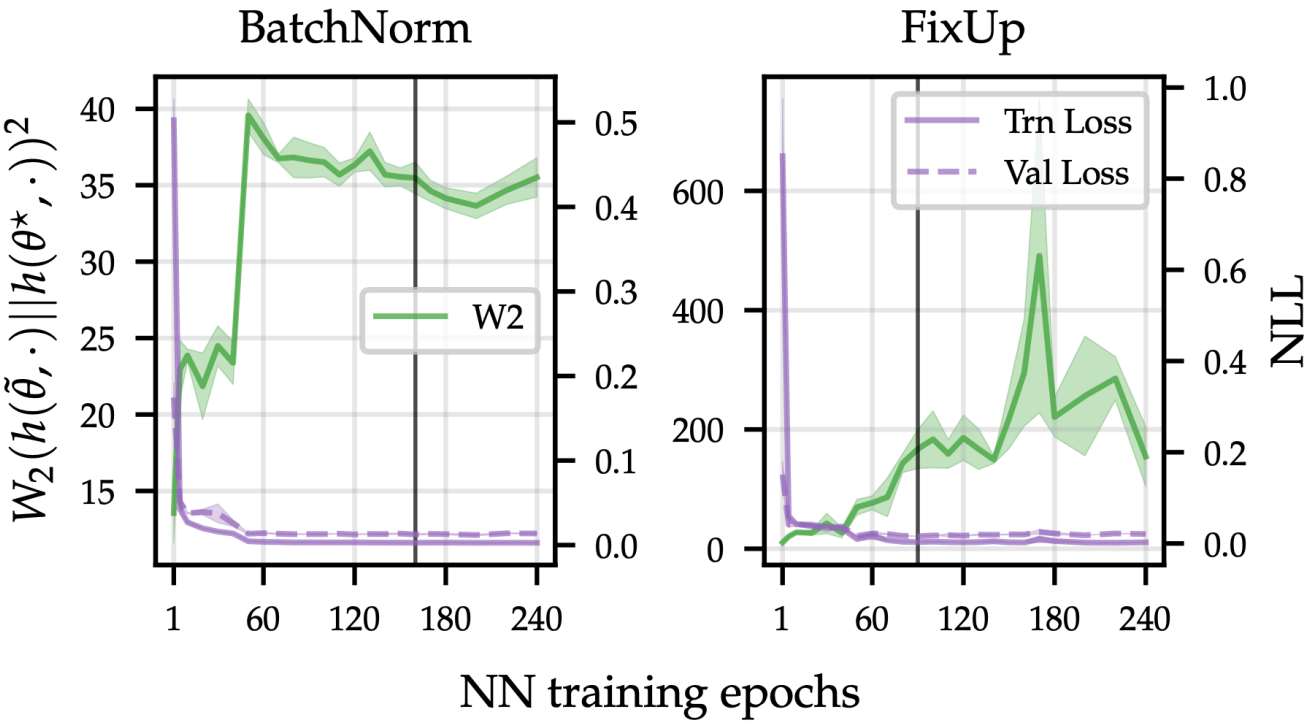


Figure 5. Wasserstein divergence between distributions obtained when employing $\mathcal{M}_{\tilde{\theta}}$ and $\mathcal{M}_{\theta_\star}$ as NN training progresses. The vertical black line indicates optimal early stopping.

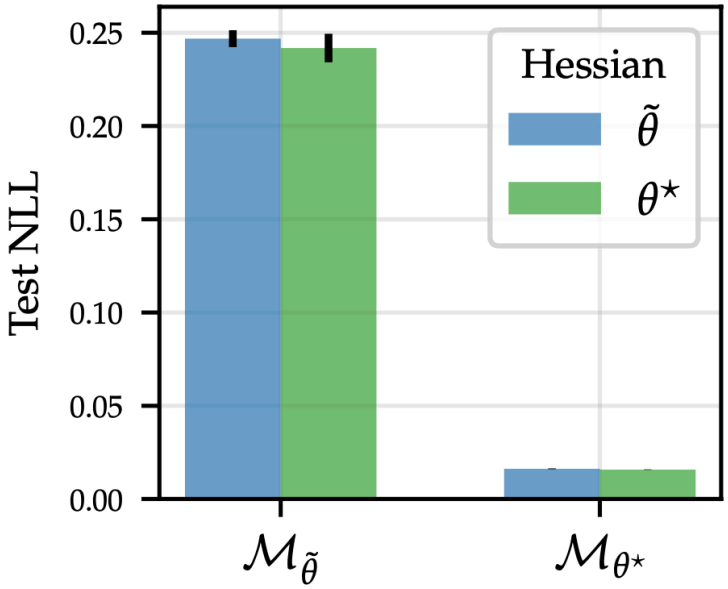
What about the choice of Hessian evaluation point?

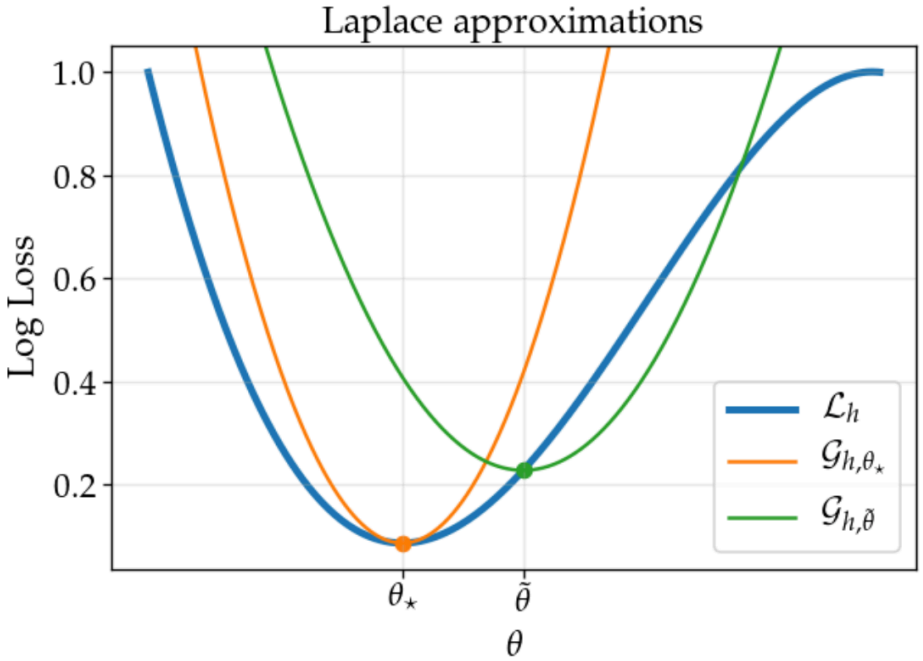
$$\mathscr{M}_{\theta_\star}(\Lambda) =$$

$$\frac{-1}{2}(||\theta_\star||^2_\Lambda + \text{logdet}(\Lambda^{-1}H + I)) + C$$

Not that wrong?

In fact, correct for regression!





ResNet image classification task:
most gains come from setting correct posterior mean in $\mathscr{M}$

Studying the effect of normalisation layers

Normalised networks:

Let $\theta = \theta' + \theta''$ with θ' having zero entries in the place of weights to which normalisation is applied and the opposite is true for θ'', then

$$f(\theta' + \theta'', \cdot) = f(\theta' + k\theta'', \cdot) \quad \text{for} \quad k > 0$$

Definition applies to:

- Batch norm
- Layer norm
- Group norm
- Normalisation-free ResNets

The MAP solution does not exist for normalised networks

Normalisation introduces scale invariance

$$f(\theta' + \theta'', \cdot) = f(\theta' + k\theta'', \cdot) \quad \text{for} \quad k > 0$$

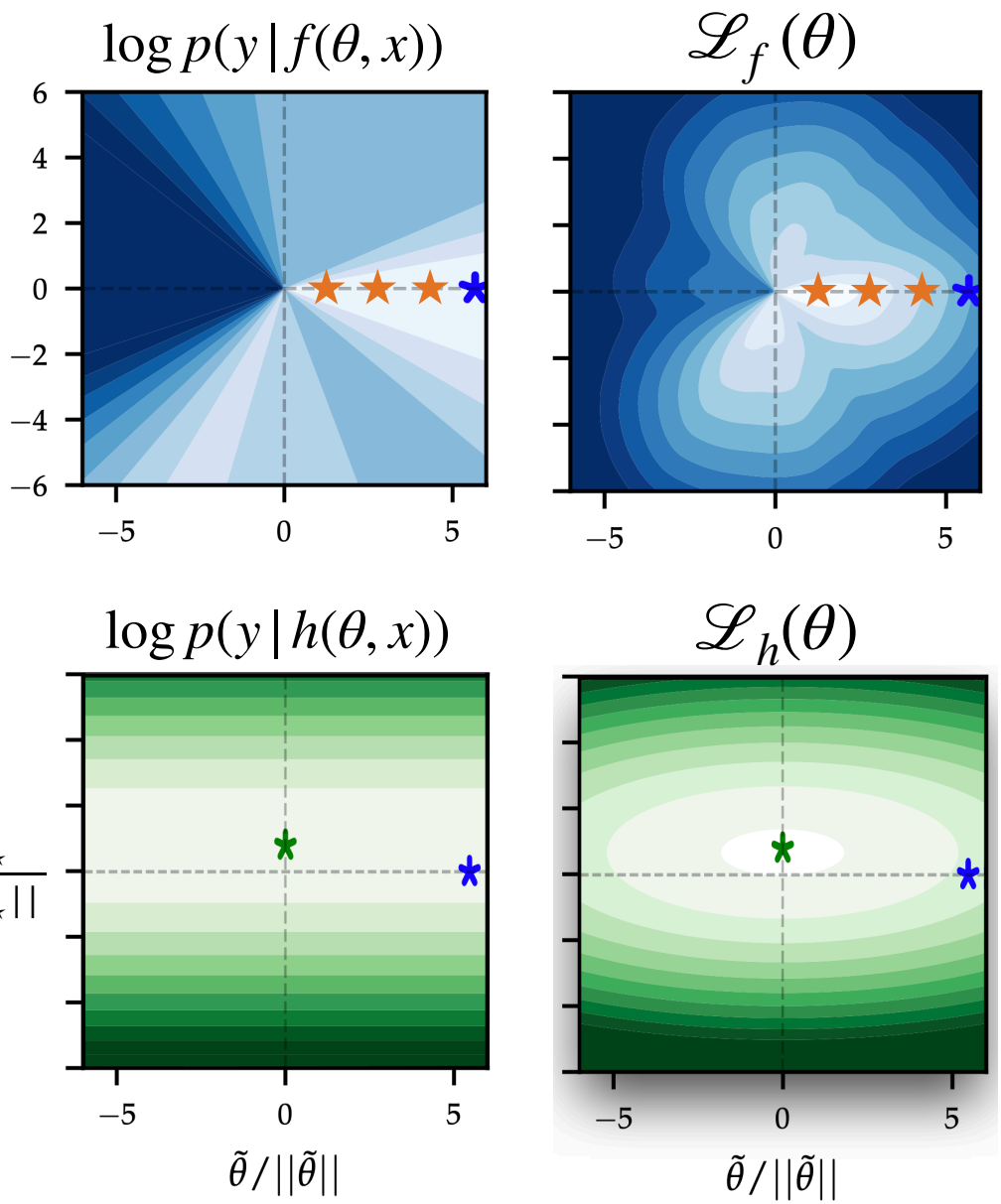
$$\mathscr{L}_f(\theta) = \underbrace{\log p(y \,|\, f(\theta, x))}_{\text{invariant}} + \underbrace{||\theta||^2_\Lambda}_{\text{not invariant}}$$

$$\implies \mathscr{L}_f(\theta' + \theta'') > \mathscr{L}_f(\theta' + \frac{1}{2}\theta'')$$

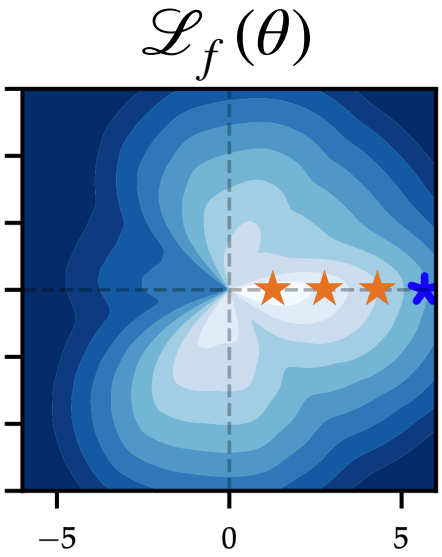
Linearisation point $\tilde{\theta}$ ★ can never be a mode of the posterior since the posterior has no modes

$\mathscr{L}_h(\theta)$ has a well defined mode $\theta_\star$

→ **apply recommendation 1**

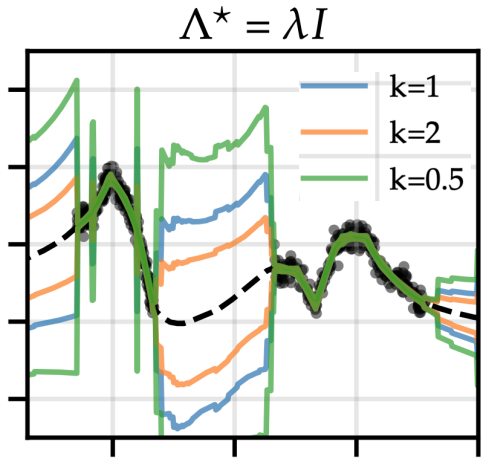


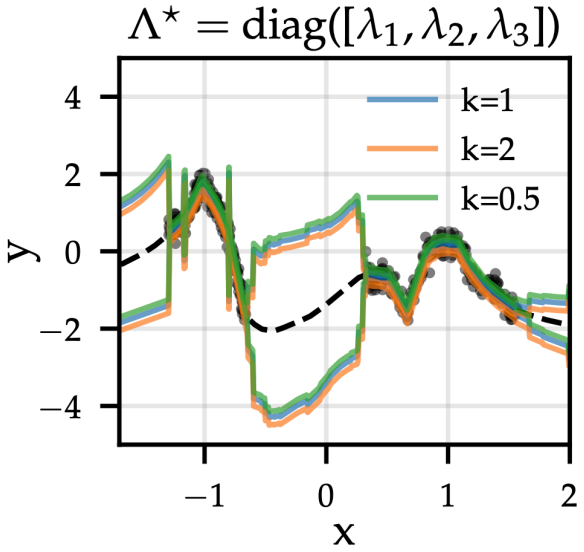
Dependence on scale of linearisation point k



k does not affect NN predictions so it should not affect the predictive variance!

However, in general, it does!





Proposition 3. *For normalised neural networks, using a regulariser of the form* $\|\theta'\|^2_{\Lambda'} + \|\theta''\|^2_{\Lambda''}$ *with* Λ' *and* Λ'' *parametrised independently and chosen according to recommendation 1, the predictive posterior* $h(\theta, \cdot)$, $\theta \sim Q$ *induced by a linearisation point* $\theta' + k\theta''$ *is independent of the choice of* $k > 0$.

Recommendation 2: learn an independent regulariser for each normalised group of weights $\theta^{(n)}$, i.e.

$$\log \pi(\theta) \propto \lambda' ||\theta'||^2 + \sum_n \lambda^{(n)} ||\theta^{(n)}||^2$$

Systematic analysis (46k param models)

Table 1. Validation of recommendations across architectures. All results are reported as negative log-likelihoods (lower is better). In each column, the best performing method is bolded. For each $\mathcal{M}$, if single or multiple λ optimisation performs better it is underlined.

		T-FORMER	CNN	RESNET	PRE-RESNET	FIXUP*	U-NET
$\mathcal{M}_{\theta_\star}$	single λ	**0.162** $\pm$ 0.042	**0.025** $\pm$ 0.000	**0.017** $\pm$ 0.000	0.017 $\pm$ 0.000	**0.055** $\pm$ 0.006	—
	multiple λs	**0.162** $\pm$ 0.042	**0.025** $\pm$ 0.000	**0.016** $\pm$ 0.001	**0.016** $\pm$ 0.000	0.061 $\pm$ 0.005	**-2.240** $\pm$ 0.027
$\mathcal{M}_{\tilde{\theta}}$	single λ	0.310 $\pm$ 0.060	0.253 $\pm$ 0.001	0.252 $\pm$ 0.006	0.220 $\pm$ 0.004	0.153 $\pm$ 0.021	—
	multiple λs	**0.162** $\pm$ 0.042	0.205 $\pm$ 0.002	0.236 $\pm$ 0.005	0.239 $\pm$ 0.004	0.200 $\pm$ 0.018	-1.703 $\pm$ 0.023

Recommendation 1 + Recommendation 2 is best in all cases

* Fixup is a non-scale invariant alternative to normalisation layers so recommendation 2 does not apply

Validation on ResNet-50 (23M parameters)

Table 2. Test negative log-likelihoods for ResNet-50 on CIFAR10.

		BatchNorm	FixUp
$\mathcal{M}_{\theta_\star,\text{simple}}$	single λ	**-0.773** $\pm$ 0.004	-0.744 $\pm$ 0.000
	multiple λs	**-0.778** $\pm$ 0.003	**-0.801** $\pm$ 0.000
$\mathcal{M}_{\theta_\star,\text{full}}$	single λ	-0.645 $\pm$ 0.005	-0.563 $\pm$ 0.002
	multiple λs	-0.639 $\pm$ 0.009	-0.641 $\pm$ 0.001
$\mathcal{M}_{\tilde{\theta}}$	single λ	-0.269 $\pm$ 0.004	-0.387 $\pm$ 0.000
	multiple λs	-0.271 $\pm$ 0.004	-0.437 $\pm$ 0.000

Recommendation 1 + Recommendation 2 is best in all cases

We employ standard KFAC approximation for scalable Hessian computations

Wrapping up: treat your NNs as kernels!

- Linearised Laplace should not be naively applied to modern NNs.
  - Every linearisation point $\tilde{\theta}$ defines a tangent linear model. Linearised Laplace uses this model to provide errorbars. Choosing hyperparameters using this model's evidence avoids pathologies.

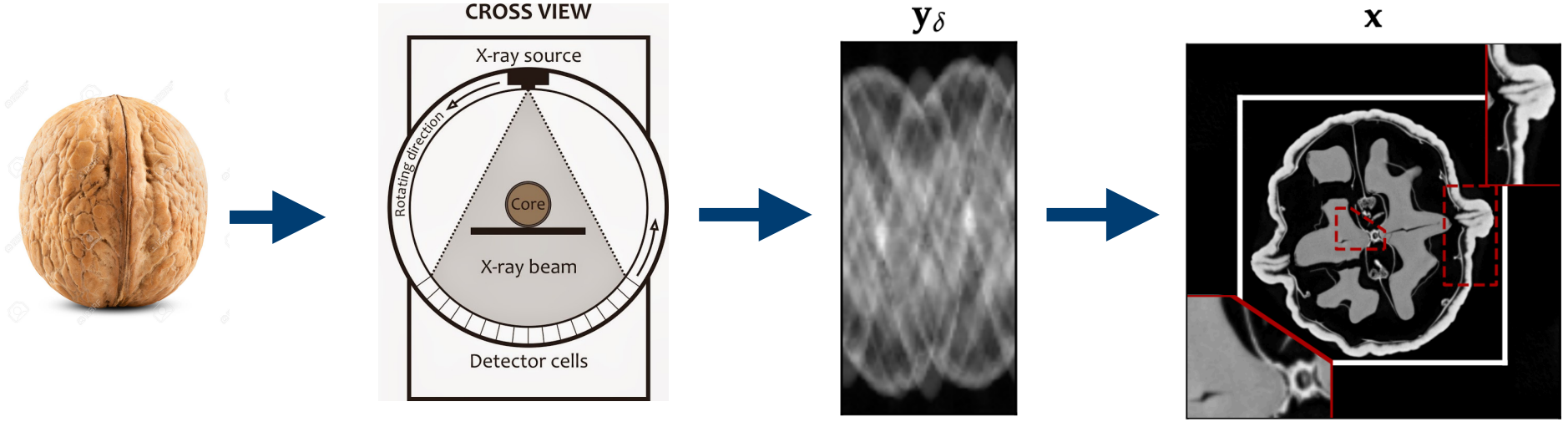
- Is the tangent linear model a good surrogate for the NN?
  - For NNs with linear dense output layers, $f(\tilde{\theta}, \cdot)$ is in the linear span of the Jacobian basis expansion $J(\cdot) = \partial_\theta f(\tilde{\theta}, \cdot)$

- Furthermore, for normalised networks with dense output layers:
  - Linearisation simplifies to $h(\theta, \cdot) = J(\cdot)\theta,\ \theta \sim \mathcal{N}(0, \Lambda)$ and thus induces a GP prior $f \sim \mathrm{GP}(0,\ J\Lambda^{-1}J^T)$

Case study: probabilistic inference with linearised neural networks for X-ray image reconstruction

A brief primer on inverse problems

- Consider the setting $y_\delta = Ax + \eta, \quad \eta \sim \mathcal{N}(0, \sigma_y^2 I)$ and
- We observe $y_\delta \in \mathbb{R}^{d_y}$ and are tasked with recovering $x \in \mathbb{R}^{d_x}$, and $d_x \gg d_y$



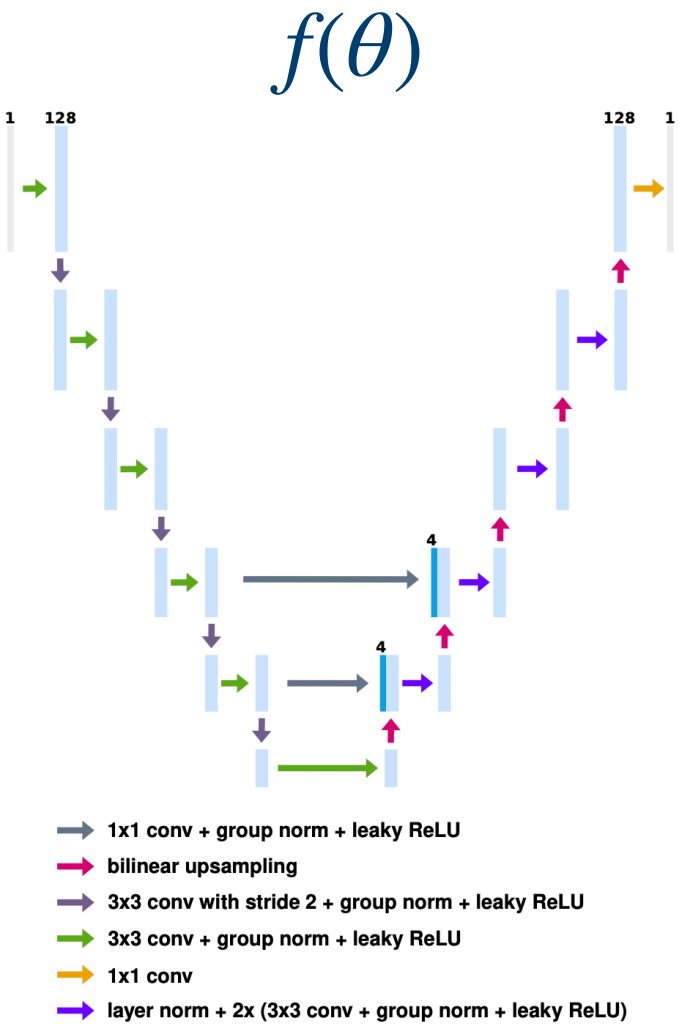
- Clearly the problem is ill posed
  - Traditionally, x is estimated through regularised reconstruction
  - Can we design a Bayesian prior $p(x)$ to solve this task?

The “deep image prior” for inverse problems

- Standard solution: “Deep image prior”

$$\operatorname{argmin}_\theta \underbrace{(y_\delta - Af(\theta))^2}_{\text{data fit}} + \lambda \underbrace{\mathrm{TV}(f(\theta))}_{\text{classical regulariser}}$$

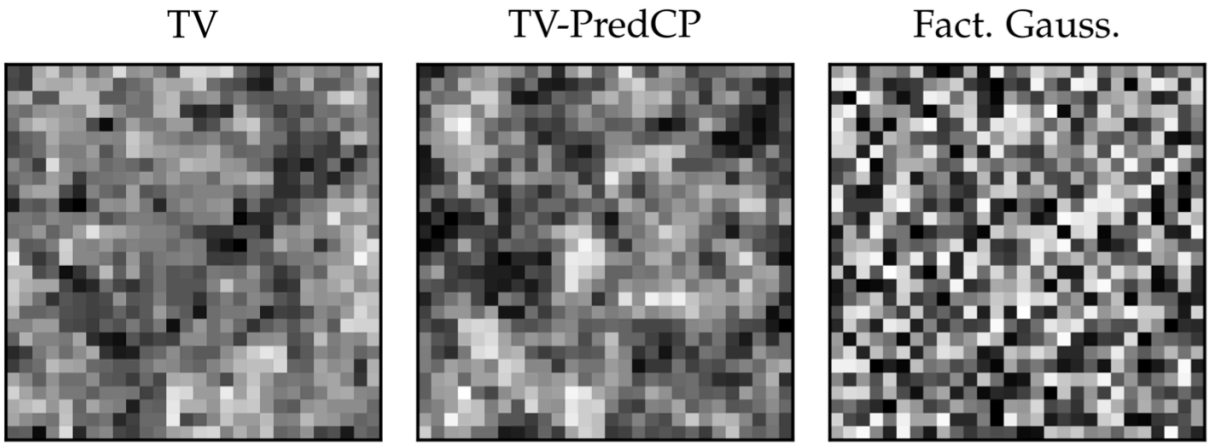
Can be interpreted as a MAP objective given a prior that constrains reconstructions to be the output of a U-net and have low TV



From regularised reconstruction to Bayesian inference

- Can build unnormalised prior $p(f) \propto \exp(-\lambda \mathrm{TV}(f))$
  - Normalising constant does not admit closed form
  - Hessian is 0 almost everywhere $\Longrightarrow$ can't use Laplace
- **Idea**: build surrogate Gaussian prior with a covariance kernel that enforces TV smoothness

$$f \sim N(0, K(\Lambda)), \quad \Lambda \sim p(\Lambda) = \mathrm{Exp}(TV(f); \lambda) \left| \frac{\partial \mathrm{TV}(f)}{\partial \Lambda} \right|$$



Building a probabilistic deep image prior

1. Train U-net with standard objective: $(y_\delta - Af(\theta))^2 + \mathrm{TV}(f(\theta))$

2. Linearise around some acceptable parameter setting $\tilde{\theta}$

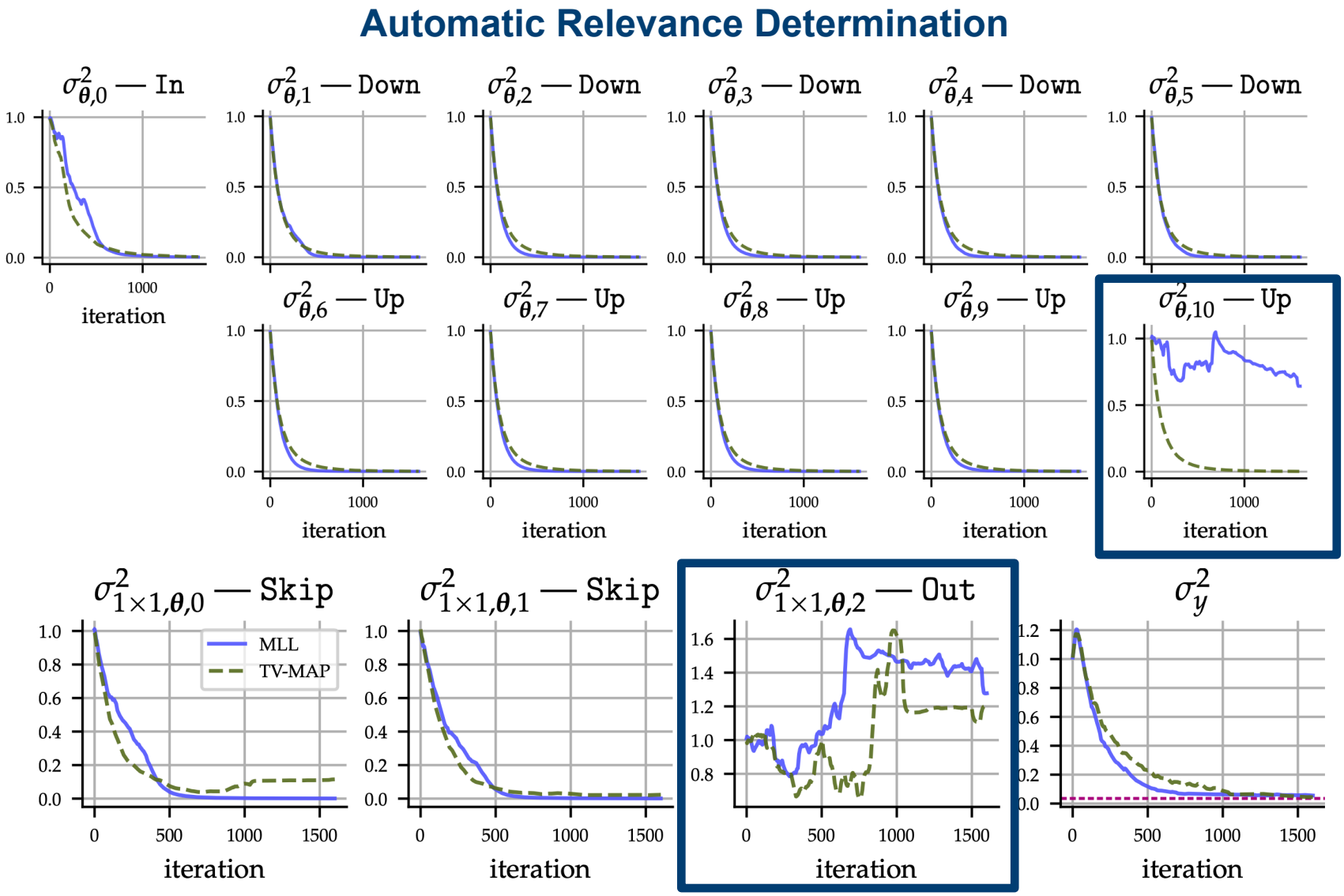
3. Build Bayesian hierarchical model

$$y_\delta \sim \mathcal{N}(Af,\, \sigma_y^2 I), \quad f \sim \mathcal{N}(0,\, J\Lambda^{-1}J^T), \quad \Lambda \sim p(\Lambda) = \mathrm{Exp}(TV(f);\lambda) \left| \frac{\partial \mathrm{TV}(f)}{\partial \Lambda} \right|$$

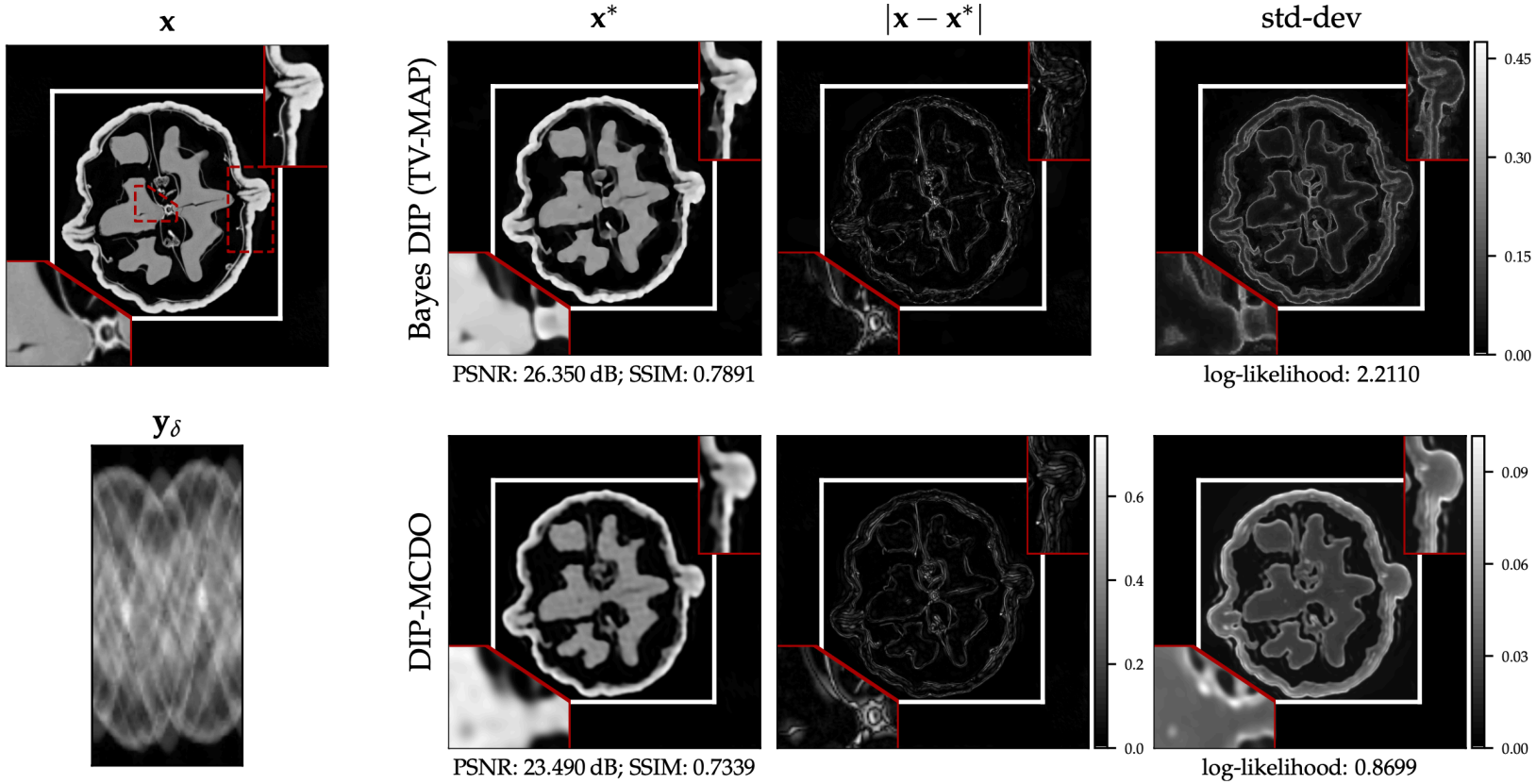
4. Optimise hyperparameters with marginal likelihood

5. Make predictions (cheap because $d_y \ll d_x, d_\theta$)

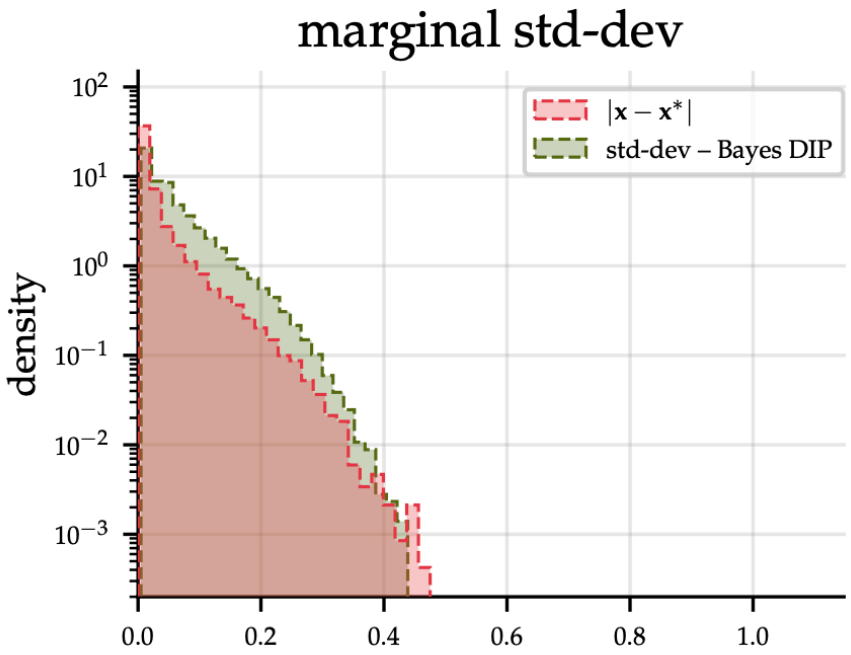
Optimising hyperparameters with the marginal likelihood



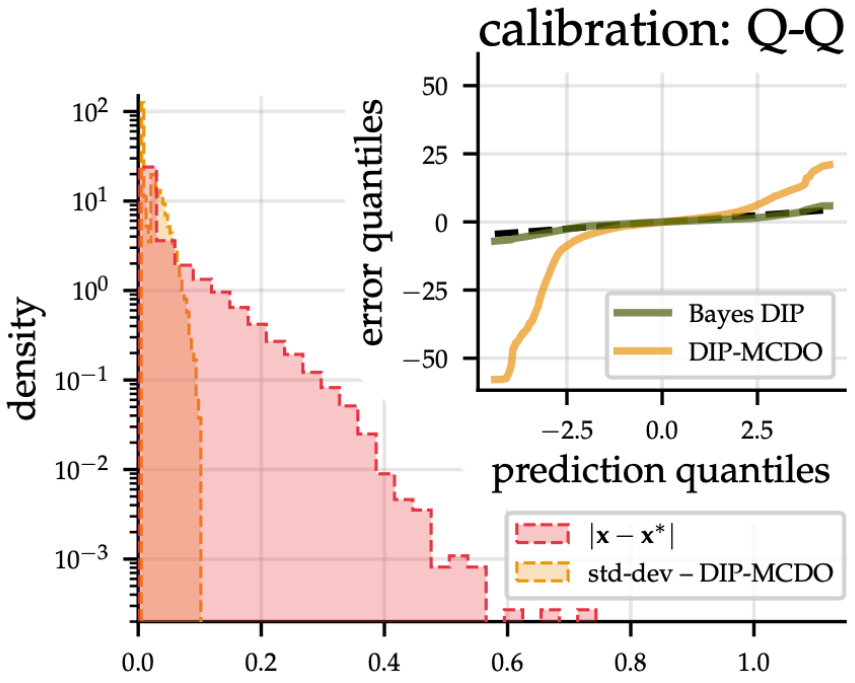
Some results



Calibration comparison



Lin Laplace



Dropout

Take-home message 2: use deep learning to build specialised kernels

- We can obtain very powerful task-specific kernels by training a NN to solve a task and then linearising it.

- Once the network is trained, we the tangent linear model $f \sim \mathrm{GP}(0,\, J\Lambda^{-1}J^T)$ provides us with uncertainty estimates and a model selection objective.

Thank you to all my collaborators!

James Allingham

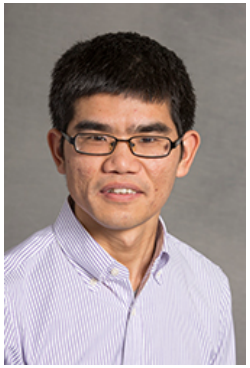
David Janz

Erik Daxberger

Riccardo Barbano

Johannes Leuschner

Bangti Jin



Eric Nalisnick

José Miguel Hernández-Lobato

Papers discussed

- Preliminaries: Daxberger et. al. “Bayesian Deep Learning via Subnetwork Inference”, ICML 2021
- Antorán et. al. “Adapting the Linearised Laplace Model Evidence for Modern Deep Learning”, will be released in next couple of weeks
- Antorán et. al. “A Probabilistic Deep Image Prior for Computational Tomography”, https://arxiv.org/pdf/2203.00479.pdf