UNIVERSIDAD
DE GRANADA

Linearized Laplace Approximation for Modern Deep Learning (II)

Francisco Miguel Castro Macías
fcastro@ugr.es

Visual Information Processing Group (VIP)
Department of Computer Science and Artificial Intelligence (DECSAI)
University of Granada (UGR)

May 5, 2023

Overview

1. Introduction
   1.1 Our goal
   1.2 A short recap
2. The Linearized Laplace Approximation as a Gaussian Linear Model
3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm
4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step
5. Sample-based Lin. Laplace inference

1. Introduction
   1.1 Our goal
   1.2 A short recap

2. The Linearized Laplace Approximation as a Gaussian Linear Model

3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm

4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step

5. Sample-based Lin. Laplace inference

Our goal

$$\underbrace{\text{Sampling-based inference for large linear models, with application to linearised Laplace}}_{\text{Today}}$$

Main references

- **Linearized Laplace Approximation as a Gaussian Linear Model.** Approximate Inference Turns Deep Networks into Gaussian Processes, Khan et al. (2019).
- **Evidence maximization using stochastic approximation.** Sampling-based inference for large linear models, with application to linearised Laplace, Antorán et al. (2022b).

Notation

Let $\mathbf{F} = (F_1, \ldots, F_M) \colon \mathbb{R}^N \to \mathbb{R}^M$ be differentiable and let $\mathrm{d}\mathbf{F}_{\mathbf{x}} \colon \mathbb{R}^N \to \mathbb{R}^M$ be its differential application at $\mathbf{x} \in \mathbb{R}^N$ given by $\mathrm{d}\mathbf{F}_{\mathbf{x}}(\mathbf{v}) = \partial_{\mathbf{x}}\mathbf{F}\mathbf{v}$ for each $\mathbf{v} \in \mathbb{R}^N$. Here, $\partial_{\mathbf{x}}\mathbf{F} \in \mathbb{R}^{M \times N}$ is the Jacobian matrix of $\mathbf{F}$ at $\mathbf{x}$ given by $(\partial_{\mathbf{x}}\mathbf{F})_{ij} = \frac{\partial F_j}{\partial x_i}(\mathbf{x})$. We denote $\nabla\mathbf{F}(\mathbf{x}) = (\partial_{\mathbf{x}}\mathbf{F})^\top \in \mathbb{R}^{N \times M}$.

When $M = 1$ and $\mathbf{F} = F$ is twice differentiable, the Hessian matrix of F at $\mathbf{x}$, denoted by $\nabla^2 F(\mathbf{x}) \in \mathbb{R}^{N \times N}$, is given by $(\nabla^2 F(\mathbf{x}))_{ij} = \frac{\partial^2 F}{\partial x_i \partial x_j}(\mathbf{x})$. If we consider the gradient as $\nabla F \colon \mathbb{R}^N \to \mathbb{R}^N$, then $\nabla^2 F(\mathbf{x}) = \partial_{\mathbf{x}}(\nabla F)^\top$.

Bayesian Deep Learning

Consider a dataset $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n) \colon n \in \{1, \dots, N\}\}$ where $\mathbf{x}_n \in \mathbb{R}^D$ and $\mathbf{y}_n \in \mathcal{Y}^C$ ($\mathcal{Y}$ can be $\mathbb{R}$ or $\{0, 1\}$). Let $\mathbf{f} = [f_1, \dots, f_C] : \mathbb{R}^D \times \mathbb{R}^P \to \mathbb{R}^C$ be a neural network that is trained to minimize the following loss function with respect to $\boldsymbol{\theta} \in \mathbb{R}^P$,

$$\mathcal{L}(\boldsymbol{\theta}, \mathbf{A}, \mathbf{f}) = \sum_n \ell\left(\mathbf{y}_n, \mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})\right) + R(\boldsymbol{\theta}, \mathbf{A}), \quad \widetilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \mathbf{A}, \mathbf{f}),$$

where ℓ is a negative log-likelihood and $\mathbf{A}$ denotes the regularization hyperparameters. Let us cast this under the Bayesian framework. Consider

$$p(\boldsymbol{\theta}, \mathcal{D}, \mathbf{A}; \mathbf{f}) = p(\mathbf{A}) p(\boldsymbol{\theta} \mid \mathbf{A}) \prod_n p(\mathbf{y}_n \mid \mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})) \prod_n p(\mathbf{x}_n),$$

$$p(\boldsymbol{\theta} \mid \mathbf{A}) = \exp(-R(\boldsymbol{\theta}, \mathbf{A}))/Z_R(\mathbf{A}), \quad p(\mathbf{y}_n \mid \mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})) = \exp(-\ell\left(\mathbf{y}_n, \mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})\right))/Z_\ell.$$

Then $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathbf{A}; \mathbf{f}) \propto \exp\left(-\mathcal{L}(\boldsymbol{\theta}, \mathbf{A}, \mathbf{f})\right)$. Therefore, minimizing $\mathcal{L}(\boldsymbol{\theta}, \mathbf{A}, \mathbf{f})$ is equivalent to maximizing $\log p(\boldsymbol{\theta} \mid \mathcal{D}, \mathbf{A}; \mathbf{f})$.

Posterior predictive: Laplace Approximation

We make (probabilistic) predictions for a new input $\mathbf{x}^*$ using the posterior predictive

$$p(\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}; \mathbf{f}) = \int_{\mathbb{R}^P} p(\mathbf{y}^* \mid \mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta})) p(\boldsymbol{\theta} \mid \mathcal{D}; \mathbf{f}) \mathrm{d}\boldsymbol{\theta} = \mathbb{E}_{p(\boldsymbol{\theta} \mid \mathcal{D}; \mathbf{f})} \left[p(\mathbf{y}^* \mid \mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta})) \right].$$

Laplace Approximation (LA).

$$p(\boldsymbol{\theta} \mid \mathcal{D}; \mathbf{f}) \approx \mathcal{N}\left(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}, \boldsymbol{\Sigma}\right),$$

$$\boldsymbol{\Sigma}^{-1} = \nabla^2_{\boldsymbol{\theta}} R(\widetilde{\boldsymbol{\theta}}) + \textstyle\sum_n \nabla^2_{\boldsymbol{\theta}} \ell\left(\mathbf{y}_n, \mathbf{f}(\mathbf{x}_n, \widetilde{\boldsymbol{\theta}})\right).$$

The Linearized Laplace Approximation

We consider the *local linearization of* $\mathbf{f}$ *around* $\widetilde{\boldsymbol{\theta}}$,

$$\mathbf{f}^{\text{lin}}(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{f}(\mathbf{x}, \widetilde{\boldsymbol{\theta}}) + \partial_{\boldsymbol{\theta}}\mathbf{f}(\mathbf{x}, \widetilde{\boldsymbol{\theta}})(\boldsymbol{\theta} - \widetilde{\boldsymbol{\theta}})$$

Linearized Laplace Approximation (LLA)

$$p(\boldsymbol{\theta} \mid \mathcal{D}; \mathbf{f}) \approx p(\boldsymbol{\theta} \mid \mathcal{D}; \mathbf{f}^{\text{lin}}) \approx \mathcal{N}\left(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\text{GGN}}\right),$$

$$\boldsymbol{\Sigma}_{\text{GGN}}^{-1} = \nabla^2_{\boldsymbol{\theta}} R(\widetilde{\boldsymbol{\theta}}) + \textstyle\sum_n \partial_{\boldsymbol{\theta}}\mathbf{f}(\mathbf{x}_n, \widetilde{\boldsymbol{\theta}})^\top \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{\theta}}}(\mathbf{x}_n, \mathbf{y}_n) \partial_{\boldsymbol{\theta}}\mathbf{f}(\mathbf{x}_n, \widetilde{\boldsymbol{\theta}}),$$

$$\boldsymbol{\Lambda}_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{y}_n) = \nabla^2_{\mathbf{f}} \ell(\mathbf{y}_n, \mathbf{f})\big|_{\mathbf{f}=\mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})}$$

Alternative adaptation of LLA, Antorán et al. (2022a)

Objective. Obtain a pair $(\boldsymbol{\theta}^*, \mathbf{A}^*)$ that satisfies

$$\boldsymbol{\theta}^* \in \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \mathbf{A}^*, \mathbf{f}^{\text{in}}), \quad \mathbf{A}^* \in \arg\max_{\mathbf{A}} \mathcal{M}(\mathbf{A}, \boldsymbol{\theta}^*).$$

1. Start with an initial guess of $\mathbf{A}^*$ and minimize $\mathcal{L}(\boldsymbol{\theta}, \mathbf{A}^*, \mathbf{f})$ to obtain $\widetilde{\boldsymbol{\theta}}$.
2. Minimize $\mathcal{L}(\boldsymbol{\theta}, \mathbf{A}^*, \mathbf{f}^{\text{in}})$ obtaining $\boldsymbol{\theta}^*$. In most situations, $\boldsymbol{\theta} \mapsto \mathcal{L}(\boldsymbol{\theta}, \mathbf{A}, \mathbf{f}^{\text{in}})$ will be convex, yielding a minimizer (convergence guaranteed).
3. Maximize $\mathcal{M}(\mathbf{A}, \boldsymbol{\theta}^*)$ obtaining $\mathbf{A}^*$. For every $\boldsymbol{\theta}$, $\mathbf{A} \mapsto \mathcal{M}(\mathbf{A}, \boldsymbol{\theta})$ is concave, yielding a maximizer (convergence guaranteed).
4. If $(\boldsymbol{\theta}^*, \mathbf{A}^*)$ is not a stationary point, move to step 2.

Finally, use the pair $(\boldsymbol{\theta}^*, \mathbf{A}^*)$ to compute the posterior approximation.

1. Introduction
   1.1 Our goal
   1.2 A short recap

2. The Linearized Laplace Approximation as a Gaussian Linear Model

3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm

4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step

5. Sample-based Lin. Laplace inference

The LLA as a GLM

We seek a Gaussian Linear Model (GLM) whose posterior distribution is equal to the LLA approximation. We define the following *pseudo-inputs*,

$$\widetilde{\mathbf{y}}_n = \boldsymbol{\phi}(\mathbf{x}_n)\widetilde{\boldsymbol{\theta}} - \boldsymbol{\Lambda}_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{y}_n)^{-1}\mathbf{r}_{\widetilde{\boldsymbol{\theta}}}(\mathbf{x}_n, \mathbf{y}_n),$$

where $\mathbf{r}_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{y}_n) = \nabla_{\mathbf{f}}\ell(\mathbf{y}, \mathbf{f})\big|_{\mathbf{f}=\mathbf{f}(\mathbf{x}_n,\boldsymbol{\theta})}$ and $\boldsymbol{\phi}(\mathbf{x}) = \partial_{\boldsymbol{\theta}}\mathbf{f}(\mathbf{x}, \widetilde{\boldsymbol{\theta}})$. We consider the *transformed dataset*

$$\widetilde{\mathcal{D}} = \{(\mathbf{x}_1, \widetilde{\mathbf{y}}_1), \ldots, (\mathbf{x}_N, \widetilde{\mathbf{y}}_N)\}.$$

Next, we consider the linear model

$$\widetilde{\mathbf{y}}_n = \boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\theta} + \boldsymbol{\eta}_n, \quad \boldsymbol{\theta} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{A}^{-1}\right), \quad \boldsymbol{\eta}_n \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{\theta}}}(\mathbf{x}_n, \mathbf{y}_n)^{-1}\right) \tag{1}$$

The LLA as a GLM (Khan et al., 2019, Theorem 1).

The Linearized Laplace Approximation $\mathcal{N}\left(\boldsymbol{\theta} \mid \widetilde{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\text{GGN}}\right)$ is equal to the posterior distribution $p(\boldsymbol{\theta} \mid \widetilde{\mathcal{D}})$ of the GLM (1).

1. Introduction
   1.1 Our goal
   1.2 A short recap

2. The Linearized Laplace Approximation as a Gaussian Linear Model

3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm

4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step

5. Sample-based Lin. Laplace inference

Conjugate Gaussian Regression

We study Bayesian Conjugate Gaussian Regression (CGR) with multidimensional outputs,

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\} \subset \mathbb{R}^D \times \mathbb{R}^C,$$
$$\boldsymbol{\phi}\colon \mathbb{R}^D \to \mathbb{R}^{C\times P}, \quad \boldsymbol{\theta} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{A}^{-1}\right), \quad \boldsymbol{\eta}_n \sim \mathcal{N}\left(\mathbf{0}, \mathbf{B}_n^{-1}\right),$$
$$\mathbf{y}_n = \boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\theta} + \boldsymbol{\eta}_n.$$

This can be written

$$\mathbf{Y} = \boldsymbol{\Phi}\boldsymbol{\theta} + \boldsymbol{\eta}, \quad \boldsymbol{\theta} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{A}^{-1}\right), \quad \boldsymbol{\eta} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{B}^{-1}\right),$$
$$\mathbf{Y} = \left[\mathbf{y}_1^\top; \ldots; \mathbf{y}_N^\top\right]^\top \in \mathbb{R}^{NC}, \quad \boldsymbol{\Phi} = \left[\boldsymbol{\phi}(\mathbf{x}_1)^\top; \ldots; \boldsymbol{\phi}(\mathbf{x}_N)^\top\right]^\top \in \mathbb{R}^{NC\times P},$$
$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & & \\ & \ddots & \\ & & \mathbf{B}_N \end{bmatrix} \in \mathbb{R}^{NC\times NC}$$

EM algorithm

Goal. Infer the posterior distribution for the parameters $\boldsymbol{\theta}$ given our observations, under the setting of $\mathbf{A} = \alpha\mathbf{I}$ most likely to have generated the observed data.

Iterative procedure.

(E step) Given $\mathbf{A}$, the posterior for $\boldsymbol{\theta}$ is computed exactly as

$$\Pi = \mathcal{N}\left(\bar{\boldsymbol{\theta}}, \mathbf{H}^{-1}\right), \quad \mathbf{H} = \boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi} + \mathbf{A}, \quad \bar{\boldsymbol{\theta}} = \mathbf{H}^{-1}\boldsymbol{\Phi}^\top \mathbf{B}\mathbf{Y}.$$

(M step) Given Π, the evidence for the model is optimized using

$$\log p(\mathbf{Y}) \geq \underbrace{-\frac{1}{2}\left[\bar{\boldsymbol{\theta}}^\top \mathbf{A}\bar{\boldsymbol{\theta}} + \log\det\left(\mathbf{I} + \mathbf{A}^{-1}\boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\right)\right]}_{\mathcal{M}(\mathbf{A})} + \text{const.}$$

Problem. The E step requires inverting a $P \times P$ matrix. The M step requires evaluating its log-determinant. Both are cubic operations in P. The dual form has cubic complexity in NC.

1. Introduction
   1.1 Our goal
   1.2 A short recap

2. The Linearized Laplace Approximation as a Gaussian Linear Model

3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm

4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step

5. Sample-based Lin. Laplace inference

M step

(M step) Given Π, the evidence for the model is optimized using

$$\log p(\mathbf{Y}) \geq \underbrace{-\frac{1}{2}\left[\bar{\boldsymbol{\theta}}^\top \mathbf{A}\bar{\boldsymbol{\theta}} + \log\det\left(\mathbf{I} + \mathbf{A}^{-1}\boldsymbol{\Phi}^\top\mathbf{B}\boldsymbol{\Phi}\right)\right]}_{\mathcal{M}(\mathbf{A})} + \text{const.}$$

The function $\mathbf{A} \mapsto \mathcal{M}(\mathbf{A})$ is concave. Its maximizer is given by $\nabla_{\mathbf{A}}\mathcal{M}(\mathbf{A}) = \mathbf{0}$, which leads to

$$\bar{\boldsymbol{\theta}}^\top\bar{\boldsymbol{\theta}}\mathbf{A} = \boldsymbol{\Phi}^\top\mathbf{B}\boldsymbol{\Phi}\left(\mathbf{A} + \boldsymbol{\Phi}^\top\mathbf{B}\boldsymbol{\Phi}\right)^{-1} \underbrace{\Longrightarrow}_{\mathrm{Tr}(\cdot)} \bar{\boldsymbol{\theta}}^\top\mathbf{A}\bar{\boldsymbol{\theta}} = \underbrace{\mathrm{Tr}\left(\mathbf{H}^{-1}\boldsymbol{\Phi}^\top\mathbf{B}\boldsymbol{\Phi}\right)}_{\gamma}.$$

The quantity γ is the *effective dimension of the regression problem.* Setting $\mathbf{A} = \alpha\mathbf{I}$ leads to $\alpha = \gamma / \left\|\bar{\boldsymbol{\theta}}\right\|^2$.

M step: computing γ

Using Hutchinson (1989)'s trick,

$$\gamma = \mathrm{Tr}\left(\mathbf{H}^{-1}\mathbf{\Phi}^\top \mathbf{B}\mathbf{\Phi}\right) = \mathbb{E}_{\mathcal{N}(\mathbf{0},\mathbf{H}^{-1})}\left[\boldsymbol{\xi}^\top \mathbf{\Phi}^\top \mathbf{B}\mathbf{\Phi}\boldsymbol{\xi}\right] \approx \frac{1}{K}\sum_{k=1}^{K} \widehat{\boldsymbol{\xi}}_k^\top \mathbf{\Phi}^\top \mathbf{B}\mathbf{\Phi}\widehat{\boldsymbol{\xi}}_k, \quad \widehat{\boldsymbol{\xi}}_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{H}^{-1}\right)$$

Remarks.

- New hyperparameter: the number of samples, K.
- How to sample efficiently from $\mathcal{N}\left(\mathbf{0}, \mathbf{H}^{-1}\right)$?

E step: sampling from the posterior using SGD

Sample-then-optimize (Papandreou and Yuille, 2010), (Antorán et al., 2022b, Appendix C.1).

Let $\mathbf{E} = \left[\boldsymbol{\epsilon}_1^\top; \ldots; \boldsymbol{\epsilon}_N^\top\right]^\top \in \mathbb{R}^{NC}$, with $\boldsymbol{\epsilon}_n \sim \mathcal{N}\left(\mathbf{0}, \mathbf{B}_n^{-1}\right)$, and $\boldsymbol{\theta}^0 \sim \mathcal{N}\left(\mathbf{0}, \mathbf{A}^{-1}\right)$. Then, the minimizer of the following loss is a random variable $\boldsymbol{\xi} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{H}^{-1}\right)$,

$$L(\mathbf{z}) = \underbrace{\frac{1}{2}\left(\boldsymbol{\Phi}\mathbf{z} - \mathbf{E}\right)^\top \mathbf{B}\left(\boldsymbol{\Phi}\mathbf{z} - \mathbf{E}\right)}_{T_1} + \underbrace{\frac{1}{2}\left(\mathbf{z} - \boldsymbol{\theta}^0\right)^\top \mathbf{A}\left(\mathbf{z} - \boldsymbol{\theta}^0\right)}_{T_2}$$

- $\nabla_\mathbf{z} T_1 = \boldsymbol{\Phi}^\top \mathbf{B}\left(\boldsymbol{\Phi}\mathbf{z} - \mathbf{E}\right) = \sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{B}_n \left(\boldsymbol{\phi}(\mathbf{x}_n)\mathbf{z} - \boldsymbol{\epsilon}_n\right) \rightarrow$ It requires stochastic approximation for large datasets.
- $\nabla_\mathbf{z} T_2 = \mathbf{A}\left(\mathbf{z} - \boldsymbol{\theta}^0\right) \rightarrow$ It does not require stochastic approximation.

E step: sampling from the posterior using SGD

An alternative loss is proposed,

$$L'(\mathbf{z}) = \underbrace{\frac{1}{2}\mathbf{z}^\top \boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\mathbf{z}}_{T_1'} + \underbrace{\frac{1}{2}\left(\mathbf{z}-\boldsymbol{\theta}^n\right)^\top \mathbf{A}\left(\mathbf{z}-\boldsymbol{\theta}^n\right)}_{T_2'}, \quad \boldsymbol{\theta}^n = \boldsymbol{\theta}^0 + \mathbf{A}^{-1}\boldsymbol{\Phi}^\top \mathbf{B}\mathbf{E}$$

- $\nabla_{\mathbf{z}} T_1' = \boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\mathbf{z} = \sum_n \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{B}_n \boldsymbol{\phi}(\mathbf{x}_n)\mathbf{z} \rightarrow$ It requires stochastic approximation for large datasets.
- $\nabla_{\mathbf{z}} T_2' = \mathbf{A}\left(\mathbf{z}-\boldsymbol{\theta}^n\right) \rightarrow$ It does not require stochastic approximation.

E step: sampling from the posterior using SGD

Using L' instead of L may lead to a lower minibatch gradient variance[1,2]:

$$\widehat{\mathbf{g}}_n = \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{B}_n \left(\boldsymbol{\phi}(\mathbf{x}_n)\mathbf{z} - \boldsymbol{\epsilon}_n\right), \quad \widehat{\mathbf{g}}'_n = \boldsymbol{\phi}(\mathbf{x}_n)^\top \mathbf{B}_n \boldsymbol{\phi}(\mathbf{x}_n)\mathbf{z},$$

$$\mathrm{Var}(\widehat{\mathbf{g}}_n) - \mathrm{Var}(\widehat{\mathbf{g}}'_n) = \underbrace{\mathrm{Var}\left(\boldsymbol{\Phi}^\top \mathbf{B}\mathbf{E}\right) - 2\,\mathrm{Cov}\left(\boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\mathbf{z}, \boldsymbol{\Phi}^\top \mathbf{B}\mathbf{E}\right)}_{\boldsymbol{\Delta}}$$

Criterion: L' is prefered over L if $\mathrm{Tr}\,\boldsymbol{\Delta} > 0$,

- At initialization, when $\mathbf{z}$ is independent of $\mathbf{E}$, $\mathrm{Tr}\,\boldsymbol{\Delta} = \mathrm{Tr}\left(\boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\right) > 0$.
- At convergence, when $\mathbf{z} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{H}^{-1}\right)$, $\mathrm{Tr}\,\boldsymbol{\Delta} > 0$ if $2\alpha\gamma > \mathrm{Tr}\left(\boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi}\right)$.

[1] $\mathrm{Cov}(\mathbf{x}, \mathbf{y}) = \mathbb{E}[(\mathbf{x} - \mathbb{E}[\mathbf{x}])\,(\mathbf{y} - \mathbb{E}[\mathbf{y}])^\top]$, $\mathrm{Var}(\mathbf{x}) = \mathrm{Cov}(\mathbf{x}, \mathbf{x})$.
[2] Note that $\mathbf{z}$ is a random variable.

1. Introduction
   1.1 Our goal
   1.2 A short recap

2. The Linearized Laplace Approximation as a Gaussian Linear Model

3. Conjugate Gaussian Regression and the EM algorithm
   3.1 Conjugate Gaussian Regression
   3.2 EM algorithm

4. Evidence maximization using stochastic approximation
   4.1 M step
   4.2 E step

5. Sample-based Lin. Laplace inference

Sample-based predictions

Recall that[3]

$$\mathbf{f}^{\text{lin}}(\cdot, \boldsymbol{\theta}) \sim \mathcal{N}\left(\mathbf{f}(\cdot, \widetilde{\boldsymbol{\theta}}), \partial_{\boldsymbol{\theta}}\mathbf{f}(\cdot, \widetilde{\boldsymbol{\theta}})^\top \boldsymbol{\Sigma}_{\text{GGN}} \partial_{\boldsymbol{\theta}}\mathbf{f}(\cdot, \widetilde{\boldsymbol{\theta}})\right).$$

Therefore, for $r\colon \mathbb{R}^C \to \mathcal{Y}^C$,

$$\mathbb{E}_{\hat{\mathbf{f}} \sim \mathbf{f}^{\text{lin}}(\mathbf{x}^*, \boldsymbol{\theta})}\left[r(\mathbf{f})\right] = \frac{1}{K}\sum_{k=1}^{K} r\left(\mathbf{f}(\mathbf{x}^*, \bar{\boldsymbol{\theta}}) + \partial_{\boldsymbol{\theta}}\mathbf{f}(\mathbf{x}^*, \widetilde{\boldsymbol{\theta}})\boldsymbol{\xi}_k\right), \quad \boldsymbol{\xi}_k \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{\text{GGN}}\right).$$

[3] $\boldsymbol{\phi}(\cdot) = \partial_{\boldsymbol{\theta}}\mathbf{f}(\cdot, \widetilde{\boldsymbol{\theta}}), \mathbf{H}^{-1} = \boldsymbol{\Sigma}_{\text{GGN}}$

Sample-based Lin. Laplace inference

Algorithm 1: Sampling-based linearised Laplace hyperparameter learning and inference

Inputs: initial $\alpha > 0$; $k, k' \in \mathbb{N}$, number of samples for stochastic EM and prediction, respectively.

Compute g-prior scaling vector s as in (17)

Sample random regularisers $\theta_1^n, \ldots, \theta_k^n$ per (7)

while α *has not converged* **do**

- Find posterior mode $\bar{\theta}$ by optimising linear model loss $\mathcal{L}(h(\theta, \cdot))$, given in (13)
- Draw posterior samples $\zeta_1 \ldots \zeta_k$ by optimising objective L' with $\theta_1^n, \ldots, \theta_k^n$
- Estimate effective dimension $\hat{\gamma}$, per (5), using samples $\zeta_1 \ldots \zeta_k$
- Update prior precision $\alpha \leftarrow \hat{\gamma} / \|\bar{\theta}\|_2^2$

Sample k' random regularisers $\theta_1^{n'}, \ldots, \theta_{k'}^{n'}$ using optimised α

Draw corresponding posterior samples $\zeta_1', \ldots, \zeta_{k'}'$ using loss L'

Output: posterior samples $\zeta_1', \ldots, \zeta_k'$

Thank you!

References I

Javier Antorán, David Janz, James U Allingham, Erik Daxberger, Riccardo Rb Barbano, Eric Nalisnick, and José Miguel Hernández-Lobato. Adapting the linearised laplace model evidence for modern deep learning. In *International Conference on Machine Learning*, pages 796–821. PMLR, 2022a.

Javier Antorán, Shreyas Padhy, Riccardo Barbano, Eric Nalisnick, David Janz, and José Miguel Hernández-Lobato. Sampling-based inference for large linear models, with application to linearised laplace. *arXiv preprint arXiv:2210.04994*, 2022b.

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

References II

Mohammad Emtiyaz E Khan, Alexander Immer, Ehsan Abedi, and Maciej Korzepa. Approximate inference turns deep networks into gaussian processes. *Advances in neural information processing systems*, 32, 2019.

George Papandreou and Alan L Yuille. Gaussian sampling by local perturbations. *Advances in Neural Information Processing Systems*, 23, 2010.