Wilkinson, William; Särkkä, Simo; Solin, Arno

Bayes-Newton Methods for Approximate Bayesian Inference with PSD Guarantees

Published in:
Journal of Machine Learning Research

Published: 01/03/2023

Document Version
Publisher's PDF, also known as Version of record

Journal of Machine Learning Research 24 (2023) 1-50 Submitted 11/21; Revised 3/23; Published 3/23

Bayes–Newton Methods for Approximate Bayesian Inference with PSD Guarantees

William J. Wilkinson WILLIAM.WILKINSON@AALTO.FI
Department of Computer Science
Aalto University
Finland

Simo Särkkä SIMO.SARKKA@AALTO.FI
Department of Electrical Engineering and Automation
Aalto University
Finland

Arno Solin ARNO.SOLIN@AALTO.FI
Department of Computer Science
Aalto University
Finland

Editor: Pierre Alquier

Abstract

We formulate natural gradient variational inference (VI), expectation propagation (EP), and posterior linearisation (PL) as generalisations of Newton's method for optimising the parameters of a Bayesian posterior distribution. This viewpoint explicitly casts inference algorithms under the framework of numerical optimisation. We show that common approximations to Newton's method from the optimisation literature, namely Gauss–Newton and quasi-Newton methods (*e.g.*, the BFGS algorithm), are still valid under this 'Bayes–Newton' framework. This leads to a suite of novel algorithms which are guaranteed to result in positive semi-definite (PSD) covariance matrices, unlike standard VI and EP. Our unifying viewpoint provides new insights into the connections between various inference schemes. All the presented methods apply to any model with a Gaussian prior and non-conjugate likelihood, which we demonstrate with (sparse) Gaussian processes and state space models.

Keywords: Approximate Bayesian inference, optimisation, variational inference, expectation propagation, Gaussian processes.

1. Introduction

When performing approximate Bayesian inference in probabilistic models, the need to strike a balance between accuracy and efficiency under varying use cases has led to the development of numerous schemes. Typically these have been derived from different viewpoints, and from various communities of researchers with different priorities. For example, linearisation-based methods have been intensively studied in the signal processing literature due to their intuitive nature when applied to nonlinear dynamical systems (Bell, 1994; Särkkä, 2013; García-Fernández et al., 2016). Attempts to generalise these methods beyond signal processing motivated the invention of expectation propagation (EP, Minka, 2001) in the

machine learning community as an alternative to variational inference (VI, Sato, 2001; Blei et al., 2017). However, the Laplace approximation (Tierney and Kadane, 1986) arguably remains the most popular approach for performing inference in probabilistic machine learning models due to its simplicity.

Despite their differing backgrounds, we will show here that all of these schemes can be viewed under the framework of numerical optimisation (Nocedal and Wright, 2006), namely as generalisations of Newton's method. Explicitly, we show that they all reduce to either Newton's method or the Gauss–Newton method under certain conditions. By making these links to the optimisation literature explicit, we gain new insights into the type of approximations used and the connections between the methods. For example, we show that natural gradient VI is a limiting case of power EP, we discuss the connection between variational inference and Newton's method, and we show that when approximations are applied to Newton's method the extended Kalman smoother is recovered. We also derive an improved version of the posterior linearisation algorithm (García-Fernández et al., 2016) based on our insights.

Furthermore, we show that our optimisation viewpoint provides the means by which to derive new variants of VI and EP by showing that Gauss–Newton (Björck, 1996) and quasi-Newton (Broyden, 1967) approximations remain valid in these cases. These methods address stability issues by ensuring that updates to the approximate posterior always result in positive semi-definite (PSD) covariance matrices. Such stability issues have previously hindered the use of approximate inference in cases where the likelihood is not log-concave (Challis and Barber, 2013). We also present an alternative approach to PSD constraints based on Riemannian gradients (Lin et al., 2020).

Finally, we demonstrate that all the methods outlined here can be used to perform inference in Gaussian processes (Rasmussen and Williams, 2006) and their variants, as well as state space models. Our experiments show that for some complicated non-conjugate models our methods can significantly improve prediction accuracy relative to a simple heuristic approach.

Our main contributions, which apply generally to cases where the approximate posterior is chosen to be Gaussian, can be summarised as follows:

- We present natural gradient variational inference, power expectation propagation, and posterior linearisation under a unified 'Bayes–Newton' framework based on numerical optimisation. We argue that this presentation makes the connections between methods more explicit than in previous work.

- We utilise this framework to derive novel Gauss–Newton methods for approximate inference, including a Gauss–Newton approximation to variational inference which guarantees the posterior covariance is PSD.

- We derive a (damped) quasi-Newton method for approximate inference based on the application of BFGS updates to local likelihood terms. This leads to novel quasi-Newton approximations for variational inference, expectation propagation, and posterior linearisation, all of which guarantee the posterior covariance is PSD.

- We discuss PSD constraints for variational inference based on Riemannian gradients, and show that similar constraints can be applied to expectation propagation and

posterior linearisation. We also discuss heuristic methods for ensuring PSD covariances, and present case studies comparing the proposed methods.

2. Background

Despite their varying motivations and attributes, there has been much work pointing out the connections between different approximate inference schemes. In this section, we review this past work and set out the notation to be used throughout this paper to unify the discussed methods.

2.1 Related Work

Casting approximate inference as optimisation is a viewpoint explicitly used when performing variational inference (Blei et al., 2017), in which gradient ascent is typically applied to a lower bound of the model likelihood (see Section 4.1). The derivation of *natural gradient ascent* for VI (Amari, 1998; Khan and Lin, 2017) has made clear further connections to optimisation methods that incorporate second-order information, namely Newton's method. Notably, Khan and Rue (2021) show how VI generalises a large class of machine learning algorithms, including Newton's method, the Laplace approximation, and many gradient-based optimisation algorithms such as the Adam optimiser (Kingma and Ba, 2014; Khan et al., 2018).

The VI formulation of inference in Khan and Rue (2021) is extremely general, so much so that they refer to it as the 'Bayesian Learning Rule', but it ignores the various other approaches to approximate inference, namely linearisation-based methods and EP. This is likely because it less clear how such approaches can be cast as optimisation. However, it was shown by Bell (1994) that the iterated extended Kalman smoother (EKS) is equivalent to applying the Gauss–Newton algorithm to a nonlinear state space model. García-Fernández et al. (2016) further discuss how posterior linearisation, which is a generalisation of all nonlinear Kalman smoothers, can also be seen as a Gauss–Newton type of method.

Viewing EP as a form of gradient-based optimisation is less straight-forward, but the connections between EP and VI have been discussed at length: VI is in fact a special case of Power EP (PEP, Minka, 2004, 2005), and this relationship remains valid in settings such as sparse Gaussian process models (Bui et al., 2017) and when using natural gradients (Bui et al., 2018). By considering EP in the large-data limit, Dehaene and Barthelmé (2018) show that EP does have a deep connection to Newton's method. However, our work shows that these connections are more general than this limiting case (see Section 4.2).

Nickisch and Rasmussen (2008) provide a presentation of some of the methods discussed here with a unifying aim but do not discuss linearisation-based methods and their viewpoint does not allow for the derivation of Gauss–Newton and quasi-Newton extensions. Jylänki et al. (2011) similarly discuss a variety of approaches and explore ways to deal with the instability of inference, particularly EP, when using non-log-concave likelihoods which result in non-PSD covariance updates. They suggest a complicated scheme involving double-loop algorithms, *ad hoc* fixes, and adaptive learning rates.

Applying PSD constraints in EP has long been an open research question, with most methods being based on similar double-loop algorithms (Opper and Winther, 2005; Seeger and Nickisch, 2011). Lin et al. (2020) present a method for enforcing PSD constraints in VI

based on Riemannian gradients, and we discuss this approach in Section 7, where we also derive a similar approach for EP. On the other hand, PL and its variants are guaranteed to result in PSD covariance matrices, but are often less accurate than VI and EP since the PL parameter updates are based on first-order derivative information only (see Section 5.1 for discussion).

We also explore the use of quasi-Newton algorithms for inference. The quasi-Newton approach has been used previously to improve the computational scaling of Gaussian process regression (Leithead and Zhang, 2007). However, as discussed in Section 6.1, the application of low-rank updates to a large full-rank covariance matrix can be a very poor approximation, and hence our proposed method is instead based on updates to local terms, with a focus on accurate inference and PSD guarantees. It is also important to distinguish our work from Bayesian interpretations of quasi-Newton methods (Hennig and Kiefel, 2013; Hennig et al., 2015), which aim to characterise uncertainty about the optimisation procedure itself, rather than use the quasi-Newton algorithm for Bayesian inference as we do.

In this paper, we consider models with a Gaussian prior and non-Gaussian observation model, and all of our case studies in Section 8 are based on Gaussian processes (Rasmussen and Williams, 2006). Approximate inference has been intensively studied for such models, and we outline all the connections between our work and this body of literature in Section 8.1, Section 8.2, and Section 8.3, where we discuss Gaussian processes, sparse Gaussian processes, and state space models respectively. Challis and Barber (2013) list many other modelling scenarios that fit our specification, including Bayesian generalised linear models, binary logistic regression, and independent component analysis. Many of the presented methods could also be extended beyond the Gaussian case to any exponential family distribution, but we do not discuss such extensions here.

2.2 Model Definition and Notation

We consider models with a matrix-valued latent variable, $\mathbf{F} \in \mathbb{R}^{N \times D}$, and observed data, $\mathbf{Y} \in \mathbb{R}^{N \times D_{\mathbf{y}}}$. We use their vectorised form, letting $\mathbf{f} = \operatorname{vec}(\mathbf{F}) \in \mathbb{R}^{ND \times 1}$, $\mathbf{y} = \operatorname{vec}(\mathbf{Y}) \in \mathbb{R}^{ND_{\mathbf{y}} \times 1}$, and at all times we abuse notation by indexing them as follows: $\mathbf{f}_n = \mathbf{F}_n^\top \in \mathbb{R}^{D \times 1}$, and $\mathbf{y}_n = \mathbf{Y}_n^\top \in \mathbb{R}^{D_{\mathbf{y}} \times 1}$. For a block-diagonal matrix $\overline{\mathbf{C}}$, we also refer to its n-th block as $\overline{\mathbf{C}}_{n,n}$. This allows us to use vector notation throughout, whilst recognising that all methods are extendable to matrix-valued data (see Section 8.4 for an example).

We assume a Gaussian prior for $\mathbf{f}$ with a non-conjugate, *i.e.*, non-Gaussian, observation model for $\mathbf{y}$ (which we will refer to as the *likelihood*) that factorises as follows,

$$\begin{aligned} \mathbf{f} &\sim p(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \boldsymbol{\mu}, \mathbf{K}), && \text{(prior)} \\ \mathbf{y} \,|\, \mathbf{f} &\sim p(\mathbf{y} \,|\, \mathbf{f}) = \prod_{n=1}^{N} p(\mathbf{y}_n \,|\, \mathbf{f}_n). && \text{(likelihood)} \end{aligned} \tag{1}$$

Our main motivation for such a model is Gaussian processes (see Section 8.1), where the prior is constructed using a *mean function*, $\mu(\cdot)$, and a *kernel*, $\kappa(\cdot, \cdot)$, applied to some input features, $\mathbf{X}$: $\boldsymbol{\mu} = \mu(\mathbf{X})$, $\mathbf{K} = \kappa(\mathbf{X}, \mathbf{X})$. However, as discussed in Section 2.1, our methods are not limited to this case, so we maintain a more general presentation.

We explore methods for computing a Gaussian approximation, $q(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \mathbf{m}, \mathbf{C})$, to the non-Gaussian posterior, $p(\mathbf{f} \,|\, \mathbf{y})$:

$$q(\mathbf{f}) \approx p(\mathbf{f} \,|\, \mathbf{y}) \propto p(\mathbf{f}) \prod_{n=1}^{N} p(\mathbf{y}_n \,|\, \mathbf{f}_n). \tag{2}$$

Without loss of generality we can assume that $q(\mathbf{f})$ factorises in the same way as the true posterior by approximating the non-Gaussian likelihood with a factorisable *unnormalised* Gaussian function, $t(\mathbf{f}) = \prod_n t(\mathbf{f}_n)$. This function describes the effect of the observations on the latent variables, and whilst it is only explicitly written as a function of $\mathbf{f}$, it's parameters will be set based on $\mathbf{y}$. This results in an approximate posterior of the form,

$$q(\mathbf{f}) \propto p(\mathbf{f}) \prod_{n=1}^{N} t(\mathbf{f}_n). \tag{3}$$

We emphasise that this parametrisation is not a restriction or limitation on the approximate posterior since, as we will show, updates to $q(\mathbf{f})$ (*e.g.*, via gradient-based methods) always implicitly contain a contribution from the prior and a factorised contribution from the true likelihood. That is to say, the approximate posterior is *fully characterised* by the prior and an approximate likelihood. We denote $\overline{\mathbf{m}}$ and $\overline{\mathbf{C}}$ as the mean and covariance of $t(\mathbf{f})$ respectively, $t(\mathbf{f}) = z\mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}})$, where $\overline{\mathbf{C}} \in \mathbb{R}^{ND \times ND}$ is a block-diagonal matrix with block size D, and $z = \prod_{n=1}^{N} z_n$ is the unnormalised Gaussian constant. The above construction is very general, and we will show that it directly enables all approximate inference methods to be cast as local parameter update rules for $\overline{\mathbf{m}}$ and the block-diagonal elements of $\overline{\mathbf{C}}$.

Letting $\boldsymbol{\lambda} = [\boldsymbol{\lambda}^{(1)} \in \mathbb{R}^{ND \times 1}, \boldsymbol{\lambda}^{(2)} \in \mathbb{R}^{ND \times ND}]$ be the *natural parameters*, we parametrise the model densities as follows,

$$\begin{aligned}
&\text{Prior:} && p(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \boldsymbol{\mu}, \mathbf{K}), && \boldsymbol{\lambda}^{(1)}_{\text{prior}} = \mathbf{K}^{-1}\boldsymbol{\mu}, && \boldsymbol{\lambda}^{(2)}_{\text{prior}} = -\frac{1}{2}\mathbf{K}^{-1}, \\
&\text{Approximate likelihood:} && t(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}}), && \overline{\boldsymbol{\lambda}}^{(1)} = \overline{\mathbf{C}}^{-1}\overline{\mathbf{m}}, && \overline{\boldsymbol{\lambda}}^{(2)} = -\frac{1}{2}\overline{\mathbf{C}}^{-1}, \\
&\text{Approximate posterior:} && q(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \mathbf{m}, \mathbf{C}), && \boldsymbol{\lambda}^{(1)} = \mathbf{C}^{-1}\mathbf{m}, && \boldsymbol{\lambda}^{(2)} = -\frac{1}{2}\mathbf{C}^{-1}.
\end{aligned} \tag{4}$$

Due to conjugacy, we have $\boldsymbol{\lambda}^{(1)} = \boldsymbol{\lambda}^{(1)}_{\text{prior}} + \overline{\boldsymbol{\lambda}}^{(1)}$ and $\boldsymbol{\lambda}^{(2)} = \boldsymbol{\lambda}^{(2)}_{\text{prior}} + \overline{\boldsymbol{\lambda}}^{(2)}$.

We will now introduce our unifying perspective on approximate inference. It is important to note that in Section 3 and Section 4 we do not propose any new methods: we present a view of *existing* algorithms under a numerical optimisation framework. Later, in Section 5, Section 6, and Section 7, we derive entirely *novel* algorithms motivated by this viewpoint.

3. Newton's Method and the Laplace Approximation as Bayesian Inference

Newton's method (Nocedal and Wright, 2006) is a very general approach for finding the optimum of a function, or the mode of a distribution. It can be used to perform *maximum a*

posteriori (MAP) estimation in a Bayesian model by letting the optimisation target be the log-posterior, $\mathcal{L}(\mathbf{f}) = \log p(\mathbf{f} \,|\, \mathbf{y})$. Using the model and notation from above we have,

$$\begin{aligned} \mathcal{L}(\mathbf{f}) &= \log p(\mathbf{f} \,|\, \mathbf{y}) = \log p(\mathbf{y} \,|\, \mathbf{f}) + \log p(\mathbf{f}) - \log p(\mathbf{y}), \\ \nabla_{\mathbf{f}} \mathcal{L}(\mathbf{f}) &= \nabla_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f}) - \mathbf{K}^{-1}(\mathbf{f} - \boldsymbol{\mu}), \\ \nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{f}) &= \nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f}) - \mathbf{K}^{-1}. \end{aligned} \tag{5}$$

We then iterate the following *online* Newton updates (see Appendix A),

$$\begin{aligned} \mathbf{C}_{k+1}^{-1} &= (1-\rho)\mathbf{C}_k^{-1} - \rho\, \nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{m}_k), \\ \mathbf{m}_{k+1} &= \mathbf{m}_k + \rho\, \mathbf{C}_{k+1}\, \nabla_{\mathbf{f}} \mathcal{L}(\mathbf{m}_k), \end{aligned} \tag{6}$$

where k is the iteration number. The term 'online' refers to the fact that $\mathbf{C}_k$ is updated in a damped fashion using learning rate ρ. When $\rho = 1$, this reduces to standard Newton's method. The iterates $\mathbf{m}_k$ converge to the fixed point, $\mathbf{m}^*$, the posterior mode. This MAP estimate can be transformed into a full approximate inference scheme by using the inverse Hessian of the objective, $(-\nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{m}^*))^{-1}$, as the posterior covariance estimate, which is known as the *Laplace approximation* (Tierney and Kadane, 1986). This is a natural choice since the iterates $\mathbf{C}_k$ converge to this quantity. The approximate posterior is then given by $q(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \mathbf{m} = \mathbf{m}^*, \mathbf{C} = (-\nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{m}^*))^{-1})$.

Examining the form of the Hessian in Equation (5) we can see that the running estimate of $\mathbf{C}^{-1}$ contains additive contributions from the prior precision and a block-diagonal term depending on the likelihood: $\nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{f}) = \nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f}) - \mathbf{K}^{-1} = \nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f}) + 2\boldsymbol{\lambda}^{(2)}_{\text{prior}}$. For notational convenience we define $\nabla_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k) := \nabla_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f})|_{\mathbf{f}=\mathbf{m}_k}$ to be the Jacobian w.r.t. $\mathbf{f}$ evaluated at the posterior mean estimate. Utilising the property that $\mathbf{C}^{-1} = \mathbf{K}^{-1} + \overline{\mathbf{C}}^{-1}$, and similarly $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{\text{prior}} + \overline{\boldsymbol{\lambda}}$, the Newton updates can be rewritten in terms of the natural parameters as

$$\begin{aligned} \boldsymbol{\lambda}^{(2)}_{k+1} := -\frac{1}{2}\mathbf{C}^{-1}_{k+1} &= -(1-\rho)\frac{1}{2}\mathbf{C}_k^{-1} - \rho\,\frac{1}{2}\left(\mathbf{K}^{-1} - \nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k)\right) \\ &= \boldsymbol{\lambda}^{(2)}_{\text{prior}} + \underbrace{(1-\rho)\overline{\boldsymbol{\lambda}}^{(2)}_k + \rho\,\frac{1}{2}\nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k)}_{\overline{\boldsymbol{\lambda}}^{(2)}_{k+1}}, \\ \boldsymbol{\lambda}^{(1)}_{k+1} := \mathbf{C}^{-1}_{k+1}\mathbf{m}_{k+1} &= \mathbf{C}^{-1}_{k+1}\mathbf{m}_k + \rho\,\nabla_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k) - \rho\,\mathbf{K}^{-1}(\mathbf{m}_k - \boldsymbol{\mu}) \\ &= \boldsymbol{\lambda}^{(1)}_{\text{prior}} + \underbrace{(1-\rho)\overline{\boldsymbol{\lambda}}^{(1)}_k + \rho\left(\nabla_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k) - \nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k)\,\mathbf{m}_k\right)}_{\overline{\boldsymbol{\lambda}}^{(1)}_{k+1}}, \end{aligned} \tag{7}$$

where we define $\boldsymbol{\lambda}^{(1)}_k$, $\boldsymbol{\lambda}^{(2)}_k$ and $\overline{\boldsymbol{\lambda}}^{(1)}_k$, $\overline{\boldsymbol{\lambda}}^{(2)}_k$ respectively to be the natural parameters of the approximate posterior and approximate likelihood at iteration k. A more detailed derivation is given in Appendix A. Observing that the prior contribution in Equation (7) is fixed, with only the terms depending on the likelihood being updated across iterations, leads to the following remark:

Remark 1 *Only the block-diagonal entries of the posterior precision are updated when iterating Newton's method, since* $\nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{m}_k)$ *is block-diagonal and the prior* $\boldsymbol{\lambda}^{(2)}_{prior}$ *is fixed. Therefore Newton's method / the Laplace approximation can be written as a combination of local (likelihood) and global (posterior) updates.*

This fact, that Newton's method can be seen as iterative updates to a likelihood component, can be further clarified by defining the log likelihood to be a *surrogate target* for optimisation. The combination of this with global conjugate updates ensures that the likelihood is always updated using the latest global information. We denote this surrogate target $\overline{\mathcal{L}}(\mathbf{f}_n)$, and its Jacobian $\mathbf{J}_k$ and Hessian $\mathbf{H}_k$ (which is a block-diagonal matrix) are, for all $n = 1, 2, \ldots, N$,

$$\left.\begin{array}{ll} \overline{\mathcal{L}}(\mathbf{f}_n) & = \log p(\mathbf{y}_n \,|\, \mathbf{f}_n) \\ \mathbf{J}_{k,n} & = \nabla_{\mathbf{f}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}) \\ \mathbf{H}_{k,n,n} & = \nabla^2_{\mathbf{f}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}) \end{array}\right\} \quad \begin{array}{l} \text{surrogate target \& gradients} \\ \text{(Newton / Laplace)} \end{array} \tag{8}$$

We then iterate the following updates,

$$\left.\begin{array}{ll} \overline{\boldsymbol{\lambda}}^{(2)}_{k+1} & = (1-\rho)\, \overline{\boldsymbol{\lambda}}^{(2)}_{k} + \rho\, \frac{1}{2} \mathbf{H}_k \\ \overline{\boldsymbol{\lambda}}^{(1)}_{k+1} & = (1-\rho)\, \overline{\boldsymbol{\lambda}}^{(1)}_{k} + \rho\, (\mathbf{J}_k - \mathbf{H}_k\, \mathbf{m}_k) \end{array}\right\} \quad \begin{array}{l} \text{local likelihood} \\ \text{online Newton update} \end{array} \tag{9}$$

$$\left.\begin{array}{ll} \mathbf{C}_{k+1} & = -\frac{1}{2}(\boldsymbol{\lambda}^{(2)}_{\text{prior}} + \overline{\boldsymbol{\lambda}}^{(2)}_{k+1})^{-1} \\ \mathbf{m}_{k+1} & = \mathbf{C}_{k+1}(\boldsymbol{\lambda}^{(1)}_{\text{prior}} + \overline{\boldsymbol{\lambda}}^{(1)}_{k+1}) \end{array}\right\} \quad \begin{array}{l} \text{global posterior} \\ \text{update} \end{array} \tag{10}$$

Since $\overline{\boldsymbol{\lambda}}^{(2)}_{k+1}$ is block-diagonal (the likelihood factorises) Equation (9) is cheap to compute. That being said, the full algorithm has the same computational complexity as global updates because Equation (10) involves inverting a dense matrix. Equation (10) is equivalent to a conjugate regression step with the prior and the approximate likelihood. We notice that the Newton updates are now applied to a surrogate target, the log likelihood $\log p(\mathbf{y} \,|\, \mathbf{f})$, rather than the log posterior. However, we have shown that these updates completely characterise the full Newton method, since the contribution from the prior is static.

Whilst Equations (9) and (10) seem more complicated than the standard updates in Equation (6) (and have the same computational complexity), it turns out that this new form, in which Newton updates are applied to the local likelihoods before performing a conjugate update, will provide us with a unifying perspective that subsumes many approximate Bayesian inference algorithms. We now derive multiple such algorithms, showing how they all result in Newton-like updates of this form.

4. Bayes–Newton: Approximate Bayesian Inference as Probabilistic Variants of Newton's Method

In this section we will present three prominent Bayesian inference methods: variational inference, power expectation propagation, and posterior linearisation. Our presentation will demonstrate how these methods can all be viewed as generalisations of Newton's method, with the parameter updates taking a surprising similar form to those presented in the previous section.

The methods presented here have an important distinction from Newton's method: the target $\overline{\mathcal{L}}(\cdot)$ is a function of not only the posterior mean, $\mathbf{m}$, but also the posterior covariance, $\mathbf{C}$, and involves computing expectations with respect to a probability distribution rather than using a single point estimate at the mean.

Due to the incorporation of the full Bayesian posterior into the updates we name this class of inference algorithms *Bayes–Newton methods*. We will show that they too are completely characterised by a set of local likelihood parameter updates. In later sections we will show that common approximations to Newton's method from the optimisation literature (Gauss–Newton and quasi-Newton methods) are still valid under the Bayes–Newton framework.

4.1 Variational Inference

Variational inference aims to minimise the KL divergence of the approximation $q(\mathbf{f})$ from the true posterior,

$$q(\mathbf{f}) = \arg\min_{q^*(\mathbf{f})} \mathrm{D}_{\mathrm{KL}}\big[q^*(\mathbf{f}) \parallel p(\mathbf{f} \,|\, \mathbf{y})\big]. \tag{11}$$

Doing so is equivalent to minimising the variational free energy (VFE), *i.e.*, the negative evidence lower bound (ELBO),

$$\begin{aligned}\mathrm{VFE}(q(\mathbf{f})) &= -\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] + \mathrm{D}_{\mathrm{KL}}\big[q(\mathbf{f}) \parallel p(\mathbf{f})\big] \\ &= \mathbb{E}_{q(\mathbf{f})}[-\log p(\mathbf{y}, \mathbf{f}) + \log q(\mathbf{f})],\end{aligned} \tag{12}$$

with respect to the parameters of $q(\mathbf{f})$. It is highly desirable to apply *natural gradient* descent (Amari, 1998) to the VFE to obtain the posterior natural parameters, $\boldsymbol{\lambda}$. Following Khan and Lin (2017), we use the property that the gradient with respect to the mean parameters, $\boldsymbol{\omega} = [\mathbf{m}, \mathbf{C} + \mathbf{m}\mathbf{m}^\top]$, is equivalent to the natural gradient. This results in the following natural gradient update step,

$$\begin{aligned}\boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k - \rho\, \nabla_{\boldsymbol{\omega}} \mathrm{VFE}(q(\mathbf{f})) \\ &= (1-\rho)\boldsymbol{\lambda}_k + \rho\, \nabla_{\boldsymbol{\omega}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}, \mathbf{f})].\end{aligned} \tag{13}$$

By application of the chain rule to obtain Equation (13) in terms of gradients with respect to $\mathbf{m}$, the individual posterior parameter updates then become

$$\begin{aligned}\boldsymbol{\lambda}_{k+1}^{(2)} &= (1-\rho)\boldsymbol{\lambda}_k^{(2)} + \rho\, \frac{1}{2} \nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}, \mathbf{f})] \\ &= \boldsymbol{\lambda}_{\text{prior}}^{(2)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(2)} + \rho\, \frac{1}{2} \nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})], \\ \boldsymbol{\lambda}_{k+1}^{(1)} &= (1-\rho)\boldsymbol{\lambda}_k^{(1)} + \rho \left(\nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}, \mathbf{f})] - \nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}, \mathbf{f})]\, \mathbf{m}_k\right) \\ &= \boldsymbol{\lambda}_{\text{prior}}^{(1)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(1)} + \rho \left(\nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] - \nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})]\, \mathbf{m}_k\right).\end{aligned} \tag{14}$$

A more detailed derivation is given in Appendix B. The striking similarity between these updates and Equation (7) leads to the following remark:

Remark 2 *Natural gradient VI is fully characterised by updates to the local approximate likelihoods: only the block-diagonal of the posterior precision is updated iteratively since* $\nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})]$ *is block-diagonal. Furthermore, the updates can be framed in a similar local/global fashion to Newton's method by defining the expected log likelihood,* $\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})]$*, as the target, and calculating its gradients with respect to the posterior mean.*

We conclude that natural gradient VI can be performed by defining:

$$\left.\begin{array}{ll} \overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) & = \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\ \mathbf{J}_{k,n} & = \nabla_{\mathbf{m}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}, \mathbf{C}_{k,n,n}) \\ \mathbf{H}_{k,n,n} & = \nabla^2_{\mathbf{m}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}, \mathbf{C}_{k,n,n}) \end{array}\right\} \begin{array}{l} \text{surrogate target \& gradients} \\ \text{(VI)} \end{array} \tag{15}$$

and then iterating the local damped Newton updates, Equation (9), and the posterior updates given in Equation (10). This analysis shows that the approximate posterior does indeed factorise in the way given by Equation (2), which is perhaps surprising since VI is not usually characterised this way (whereas EP, for example, explicitly uses this parameterisation).

4.1.1 The Variational Free Energy

Whilst the natural gradient updates above have a very convenient form, it is still often useful to be able to compute the variational free energy explicitly, for example when optimising the hyperparameters of a Gaussian process model (Section 8.1), or when monitoring convergence in line search methods. Due to our parametrisation of the approximate posterior, the VFE can be written,

$$\begin{aligned} \text{VFE}(q(\mathbf{f})) &= -\mathbb{E}_{q(\mathbf{f})}\left[\log \frac{p(\mathbf{y} \,|\, \mathbf{f}) p(\mathbf{f})}{q(\mathbf{f})}\right] \\ &= -\mathbb{E}_{q(\mathbf{f})}\left[\log \frac{p(\mathbf{y} \,|\, \mathbf{f}) \int p(\mathbf{f}) t(\mathbf{f}) \,\mathrm{d}\mathbf{f}}{t(\mathbf{f})}\right] \\ &= -\mathbb{E}_{q(\mathbf{f})}\left[\log \frac{p(\mathbf{y} \,|\, \mathbf{f}) \int p(\mathbf{f}) \mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}}) \,\mathrm{d}\mathbf{f}}{\mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}})}\right] \\ &= -\sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)] + \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{f}_n)}[\log \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})] - \log \mathcal{Z}, \end{aligned} \tag{16}$$

where $\mathcal{Z} = \int p(\mathbf{f}) \mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}}) \,\mathrm{d}\mathbf{f} = \mathrm{N}(\overline{\mathbf{m}} \,|\, \boldsymbol{\mu}, \mathbf{K} + \overline{\mathbf{C}})$. The first term in Equation (16) can be computed via quadrature methods, whilst the remaining terms are both Gaussian and can be computed in closed form. Also recall that the approximate likelihood factors are unnormalised, $t(\mathbf{f}_n) = z_n \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})$, but that the constants z_n cancel out in the VFE and therefore can be ignored.

We also propose using an approximation to the VFE as the energy associated with the Laplace/Newton method by replacing the expectations in Equation (16) by point estimates at the posterior mean. We call this the Laplace energy (LE),

$$\text{LE}(q(\mathbf{f})) = -\sum_{n=1}^{N} \log p(\mathbf{y}_n \,|\, \mathbf{m}_n) + \sum_{n=1}^{N} \log \mathrm{N}(\mathbf{m}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n}) - \log \mathcal{Z}. \tag{17}$$

Alternatively, Rasmussen and Williams (2006) propose an approximation to the negative log marginal likelihood based on a first-order Taylor expansion of the exact marginal likelihood evaluated at $\mathbf{m}$, giving,

$$\text{LE}_2(q(\mathbf{f})) = -\sum_{n=1}^{N} \log p(\mathbf{y}_n \,|\, \mathbf{m}_n) + \frac{1}{2} \mathbf{m}^\top \mathbf{K}^{-1} \mathbf{m} + \frac{1}{2} \log |\mathbf{K}| + \frac{1}{2} \log |\mathbf{K}^{-1} + \overline{\mathbf{C}}^{-1}|. \tag{18}$$

4.2 Power Expectation Propagation

In the previous sections we have seen how updates to the global posterior in both Newton's method and VI are completely characterised by iterative updates to local parameters. Whilst this is not the standard presentation for either of these schemes, we now turn our attention to a method which is typically (and necessarily) defined this way.

Power expectation propagation (PEP, Minka, 2004) aims to minimise the α-divergence of the true posterior from its approximation,

$$q(\mathbf{f}) = \arg\min_{q^*(\mathbf{f})} \mathrm{D}_\alpha\big[p(\mathbf{f}\,|\,\mathbf{y}) \parallel q^*(\mathbf{f})\big]. \tag{19}$$

In practice this is intractable, so instead the local approximate likelihood terms, $t(\mathbf{f}_n)$, are updated iteratively. To update a single $t(\mathbf{f}_n)$ the current term is removed from the posterior, and replaced with the true likelihood. This new quantity is termed the *tilted distribution*, and PEP minimises the α-divergence of the approximate posterior from the tilted distribution, which can be done by raising the likelihood terms to a power of α and minimising the forward KL divergence:

$$\begin{aligned} t(\mathbf{f}_n) &= \arg\min_{t_*(\mathbf{f}_n)} \mathrm{D}_{\mathrm{KL}}\left[\frac{1}{Z_n}\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}) \parallel \frac{1}{W_n}\frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f})\right] \\ &= \arg\min_{t_*(\mathbf{f}_n)} \mathrm{D}_{\mathrm{KL}}\left[\frac{1}{Z_n}\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n) \parallel \frac{1}{W_n}\frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\right], \end{aligned} \tag{20}$$

for $Z_n = \int \frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f})\mathrm{d}\mathbf{f} = \int \frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n$ and $W_n = \int \frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f})\mathrm{d}\mathbf{f} = \int \frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n$.

The above divergence can be minimised by choosing $t_*(\mathbf{f}_n)$ such that the first two moments of the tilted distribution and the approximate posterior are matched, *i.e.*, by computing the moments of both sides, setting them to be equal, and then solving for $t_*(\mathbf{f}_n)$. In practice, a useful shortcut involves differentiating $\log Z_n$ with respect to the mean of the cavity, $q_n^{\setminus}(\mathbf{f}_n) = \mathrm{N}(\mathbf{f}_n\,|\,\mathbf{m}_n^{\setminus}, \mathbf{C}_{n,n}^{\setminus}) \propto q(\mathbf{f}_n)/t^\alpha(\mathbf{f}_n)$ (Seeger, 2005; Rasmussen and Williams, 2006). The derivatives of $\log Z_n$ turn out to be a function of the required moments, and so after some rearranging we obtain the following update algorithm for a single factor $t(\mathbf{f}_n)$ (see Appendix C for the derivation),

$$\begin{aligned} \mathbf{R}_n &= \mathbf{C}_{n,n}^{\setminus -1}\left(\nabla^2_{\mathbf{m}_n^{\setminus}} \log \mathbb{E}_{q_{\setminus n}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)] + \mathbf{C}_{n,n}^{\setminus -1}\right)^{-1}, \\ \bar{\boldsymbol{\lambda}}^{(2)}_{k+1,n,n} &= (1-\rho)\,\bar{\boldsymbol{\lambda}}^{(2)}_{k,n,n} + \rho\,\frac{1}{2\alpha}\mathbf{R}_n\,\nabla^2_{\mathbf{m}_n^{\setminus}} \log \mathbb{E}_{q_{\setminus n}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)], \\ \bar{\boldsymbol{\lambda}}^{(1)}_{k+1,n} &= (1-\rho)\,\bar{\boldsymbol{\lambda}}^{(1)}_{k,n} + \rho\,\frac{1}{\alpha}\mathbf{R}_n\Big(\nabla_{\mathbf{m}_n^{\setminus}} \log \mathbb{E}_{q_{\setminus n}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)] - \nabla^2_{\mathbf{m}_n^{\setminus}} \log \mathbb{E}_{q_{\setminus n}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)]\,\mathbf{m}_n^{\setminus}\Big), \end{aligned} \tag{21}$$

where we have *damped* the updates (Jylänki et al., 2011) using learning rate ρ. Examining Equation (21) leads to the following remark:

Remark 3 *The PEP updates for a single factor take the form of damped Bayes–Newton updates to the approximate likelihood natural parameters, with two important distinctions:* (i) *the Jacobian and Hessian are scaled by a factor* $\mathbf{R}_n$ *to account for the fact that the target is a function of the cavity rather than the posterior, and* (ii) *the updates act on the cavity mean rather than the posterior mean.*

We now define

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\mathbf{m}_n^{\setminus}, \mathbf{C}_{n,n}^{\setminus}) &= \tfrac{1}{\alpha} \log \mathbb{E}_{q^{\setminus}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\
\mathbf{R}_{k,n} &= \mathbf{C}_{k,n,n}^{\setminus -1} \left(\alpha \nabla^2_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) + \mathbf{C}_{k,n,n}^{\setminus -1} \right)^{-1} \\
\mathbf{J}_{k,n} &= \mathbf{R}_{k,n} \nabla_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) \\
\mathbf{H}_{k,n,n} &= \mathbf{R}_{k,n} \nabla^2_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus})
\end{aligned}\right\} \begin{array}{l}\text{surrogate target} \\ \text{\& gradients} \\ \text{(PEP)}\end{array} \tag{22}$$

For notational convenience we can collect the marginal cavity means $\mathbf{m}_n^{\setminus}$ together: $\mathbf{m}^{\setminus} = [\mathbf{m}_1^{\setminus}, \ldots, \mathbf{m}_N^{\setminus}]^\top$, which allows us to apply the local updates of Equation (9) by using $\mathbf{m}^{\setminus}$ in place of the posterior mean, $\mathbf{m}$. Equation (10) is then used to update the full posterior.

It is well known that when $\alpha \to 0$ the PEP energy becomes equivalent to the variational free energy (Minka, 2005; Bui et al., 2017) and that in this case, if PEP converges, it converges to the same fixed points as VI. However, our presentation reveals even deeper connections between the two methods. In Appendix D we show that the surrogate target used in PEP has the following property,

$$\lim_{\alpha \to 0} \frac{1}{\alpha} \log \mathbb{E}_{q^{\setminus}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] = \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)], \tag{23}$$

and furthermore we can see that the PEP scaling factor reverts to the identity in the limit, $\lim_{\alpha \to 0} \mathbf{R}_n = \mathbf{I}$. Combined with the fact that $\lim_{\alpha \to 0} q^{\setminus}(\mathbf{f}_n) = q(\mathbf{f}_n)$, this shows that Equation (22) ($\alpha \to 0$) is identical to the VI updates given by Equation (15), which leads to the following result:

Remark 4 *When $\alpha \to 0$, a single step of the power EP algorithm is equivalent to a natural gradient descent step in the variational free energy (i.e., a natural gradient VI step).*

This connection between PEP and natural gradient VI had gone largely unnoticed until Bui et al. (2018) recently derived the same result. However, our presentation arguably makes the connection even clearer.

4.2.1 The Power EP Energy

The PEP algorithm also provides a way to compute an approximation to the negative log marginal likelihood. This approximation is often referred to as the PEP energy (PEPE) and to derive it we must recall that the approximate likelihood factors are defined as *unnormalised* Gaussians, $t(\mathbf{f}_n) = z_n \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})$. The PEPE is then given by

$$\begin{aligned}
\mathrm{PEPE}(q(\mathbf{f})) &= -\log \int p(\mathbf{f}) \prod_{n=1}^{N} t(\mathbf{f}_n) \, \mathrm{d}\mathbf{f} \\
&= -\log \int p(\mathbf{f}) \prod_{n=1}^{N} z_n \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n}) \, \mathrm{d}\mathbf{f} \\
&= -\sum_{n=1}^{N} \log z_n - \log \mathcal{Z},
\end{aligned} \tag{24}$$

where $\mathcal{Z} = \int p(\mathbf{f}) \prod_{n=1}^{N} \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n}) \,\mathrm{d}\mathbf{f}$, which is the same term used in the VFE in Equation (16). Computing the constants, z_n, is less straightforward. To do so, we match the zero-th moment of the tilted distribution in a similar way to the first two moments, which amounts to setting z_n such that

$$z_n^\alpha \, \mathbb{E}_{q\setminus(\mathbf{f}_n)}[\mathrm{N}^\alpha(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})] = \mathbb{E}_{q\setminus(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)]. \tag{25}$$

Taking the logarithm and rearranging gives

$$\log z_n = \frac{1}{\alpha} \left(\log \mathbb{E}_{q\setminus(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] - \log \mathbb{E}_{q\setminus(\mathbf{f}_n)}[\mathrm{N}^\alpha(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})] \right), \tag{26}$$

such that the PEP energy becomes

$$\mathrm{PEPE}(q(\mathbf{f})) = -\frac{1}{\alpha} \sum_{n=1}^{N} \log \mathbb{E}_{q\setminus(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] + \frac{1}{\alpha} \sum_{n=1}^{N} \log \mathbb{E}_{q\setminus(\mathbf{f}_n)}[\mathrm{N}^\alpha(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n})] - \log \mathcal{Z}. \tag{27}$$

This presentation of the energy highlights its connection to the VFE in Equation (16): applying Equation (23) to the first two terms immediately shows that as $\alpha \to 0$, PEPE $\to$ VFE.

4.3 Posterior Linearisation

Posterior linearisation (PL, García-Fernández et al., 2016) is another approximate inference method which can be framed as updates to local likelihood factors. Whilst PL is not a commonly used method in the machine learning community, it provides an important link to the approximate Bayesian inference methods developed in the signal processing literature: it is an extension of classical statistical linearisation (Gelb, 1974), which itself generalises the sigma-point smoothing algorithms such as the unscented Kalman smoother (Särkkä, 2013). PL seeks an approximate posterior via the likelihood approximation $p(\mathbf{y} \,|\, \mathbf{f}) \approx q(\mathbf{y} \,|\, \mathbf{f}) = \mathrm{N}(\mathbf{y} \,|\, \mathbf{A}\mathbf{f} + \mathbf{b}, \boldsymbol{\Omega})$, computed via statistical linear regression (SLR, Särkkä, 2013) of $\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] := \mathbb{E}_{p(\mathbf{y} \,|\, \mathbf{f})}[\mathbf{y}]$ with respect to the approximate posterior $q(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \mathbf{m}, \mathbf{C})$. SLR provides the optimal linearisation of the likelihood model in the mean-square-error (MSE) sense,

$$\mathrm{MSE}(\mathbf{A}, \mathbf{b}) = \mathbb{E}_{q(\mathbf{f})} \left[(\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] - \mathbf{A}\mathbf{f} - \mathbf{b})^\top (\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] - \mathbf{A}\mathbf{f} - \mathbf{b}) \right]. \tag{28}$$

The PL updates can be derived by setting the derivatives of Equation (28) with respect to $\mathbf{A}$, $\mathbf{b}$ to zero, which gives

$$\begin{aligned} \mathbf{A} &= \mathbb{E}_{q(\mathbf{f})} \left[(\mathbf{f} - \mathbf{m}) (\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] - \mathbb{E}_{q(\mathbf{f})}[\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]])^\top \right] \mathbf{C}^{-1}, \\ \mathbf{b} &= \mathbb{E}_{q(\mathbf{f})}[\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]] - \mathbf{A}\mathbf{m}. \end{aligned} \tag{29}$$

The likelihood covariance is then set equal to the mean-square-error *matrix*,

$$\begin{aligned} \boldsymbol{\Omega} &= \mathbb{E}_{q(\mathbf{f}), p(\mathbf{y} \,|\, \mathbf{f})} \left[(\mathbf{y} - \mathbf{A}\mathbf{f} - \mathbf{b})(\mathbf{y} - \mathbf{A}\mathbf{f} - \mathbf{b})^\top \right] \\ &= \mathbb{E}_{q(\mathbf{f})} \left[(\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] - \mathbf{A}\mathbf{f} - \mathbf{b})(\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] - \mathbf{A}\mathbf{f} - \mathbf{b})^\top + \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}] \right]. \end{aligned} \tag{30}$$

This approach has an intuitive interpretation: PL seeks the best affine fit to the conditional expectation $\mathbb{E}[\mathbf{y} \mid \mathbf{f}]$ in the region of the approximate posterior, and sets the likelihood covariance equal to the error induced by this approximation. To enable direct comparison with VI and PEP, the approximate likelihood $q(\mathbf{y} \mid \mathbf{f})$ can also be written as a Gaussian distribution over $\mathbf{f}$: $t(\mathbf{f}) = \mathrm{N}(\mathbf{f} \mid \overline{\mathbf{m}}, \overline{\mathbf{C}})$, with update rule (see Appendix E for the derivation):

$$\left.\begin{aligned} \overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) &= \log \mathrm{N}(\mathbf{y}_n \mid \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]], \boldsymbol{\Omega}_{n,n}) \\ \mathbf{J}_{k,n} &= \nabla_{\mathbf{m}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]]^\top \boldsymbol{\Omega}_{k,n,n}^{-1} (\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]]) \\ \mathbf{H}_{k,n,n} &= -\nabla_{\mathbf{m}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]]^\top \boldsymbol{\Omega}_{k,n,n}^{-1} \nabla_{\mathbf{m}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]] \end{aligned}\right\} \begin{array}{l} \text{surrogate target} \\ \text{\& gradients} \\ \text{(PL)} \end{array} \tag{31}$$

The damped local/global Newton updates of Equation (9) and Equation (10) can now be applied to perform inference. Notice that PL only requires the Jacobian of the objective, whereas both EP and VI also require the Hessian, *i.e.*, they utilise second-order derivative information of the objective. This first-order approximation to the Hessian means PL is a *Gauss–Newton* method. We will discuss Gauss–Newton methods further in Section 5.

4.3.1 Taylor Expansion / the Iterated Extended Kalman Smoother

Another approximate inference algorithm can be obtained by replacing statistical linear regression in PL by analytical linearisation, *i.e.*, a first-order Taylor expansion (and using a simpler covariance approximation). This can be achieved by defining the simpler surrogate target $\overline{\mathcal{L}}(\mathbf{f}) = \log p(\mathbf{y} \mid \mathbf{f})$ and setting the likelihood covariance to $\boldsymbol{\Omega} = \mathrm{Cov}[\mathbf{y} \mid \mathbf{m}] := \mathrm{Cov}[\mathbf{y} \mid \mathbf{f}]_{\mathbf{f}=\mathbf{m}}$, after which the update equations become:

$$\left.\begin{aligned} \overline{\mathcal{L}}(\mathbf{f}_n) &= \log \mathrm{N}(\mathbf{y} \mid \mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n], \mathrm{Cov}[\mathbf{y}_n \mid \mathbf{f}_n]) \\ \mathbf{J}_{k,n} &= \nabla_{\mathbf{f}_n} \mathbb{E}[\mathbf{y}_n \mid \mathbf{m}_{k,n}]^\top \mathrm{Cov}[\mathbf{y}_n \mid \mathbf{m}_{k,n}]^{-1} (\mathbf{y}_n - \mathbb{E}[\mathbf{y}_n \mid \mathbf{m}_{k,n}]) \\ \mathbf{H}_{k,n,n} &= -\nabla_{\mathbf{f}_n} \mathbb{E}[\mathbf{y}_n \mid \mathbf{m}_{k,n}]^\top \mathrm{Cov}[\mathbf{y}_n \mid \mathbf{m}_{k,n}]^{-1} \nabla_{\mathbf{f}_n} \mathbb{E}[\mathbf{y}_n \mid \mathbf{m}_{k,n}] \end{aligned}\right\} \begin{array}{l} \text{surrogate target} \\ \text{\& gradients} \\ \text{(Taylor)} \end{array} \tag{32}$$

where $\mathbb{E}[\mathbf{y}_n \mid \mathbf{m}_{k,n}] = \mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]|_{\mathbf{f}_n = \mathbf{m}_{k,n}}$. Here $\mathbf{H}_{k,n,n}$ is again a first-order approximation to the true Hessian, *i.e.*, a Gauss–Newton approximation (see Section 5 and Appendix G). This approach is equivalent to the updates used in the iterated extended Kalman smoother (EKS, Bell, 1994) when performing inference in nonlinear state space models.

4.3.2 The Posterior Linearisation Energy

Unfortunately, it is not clear how to construct a marginal likelihood approximation using the computations involved in PL. García-Fernández et al. (2019) suggest an approximation for Gaussian process models similar to the PEP energy, but where they discard some terms that do not depend on the model hyperparameters. The best approach may therefore be to use the full PEP energy, however this is sub-optimal since does not re-use the computations performed during inference. Alternatively, the VFE could also be used.

On the other hand, since the Taylor/EKS approach is equivalent to a Gauss–Newton method, the Laplace energy, Equation (17), is a natural candidate to be used as the negative log marginal likelihood approximation in this case.

Table 1: Comparison of optimisation target and derivative approximations used by various Bayesian inference schemes. $\mathbf{R}_n = \mathbf{C}_{n,n}^{\backslash -1}(\alpha \nabla^2_{\mathbf{m}_n^\backslash} \overline{\mathcal{L}}(\mathbf{m}_n^\backslash) + \mathbf{C}_{n,n}^{\backslash -1})^{-1}$ is the PEP scaling factor, $q^\backslash(\mathbf{f}_n) \propto q(\mathbf{f}_n)/t^\alpha(\mathbf{f}_n)$ is the PEP cavity, $\boldsymbol{\Omega}_{n,n}$ is the mean square error matrix of the SLR approximation, Equation (28), and $\boldsymbol{\nu}(\mathbf{f}_n) = \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]$, $\boldsymbol{\Sigma}(\mathbf{f}_n) = \mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]$ are the first two moments of $p(\mathbf{y} \,|\, \mathbf{f})$.

	Surrogate target, $\overline{\mathcal{L}}(\cdot)$	Jacobian, $\mathbf{J}_n$	Hessian, $\mathbf{H}_{n,n}$
Newton	$\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)$	$\nabla_{\mathbf{f}_n} \overline{\mathcal{L}}_n(\mathbf{m}_n)$	$\nabla^2_{\mathbf{f}_n} \overline{\mathcal{L}}_n(\mathbf{m}_n)$
Taylor	$\log \mathrm{N}(\mathbf{y}_n \,|\, \boldsymbol{\nu}(\mathbf{f}_n), \boldsymbol{\Sigma}(\mathbf{f}_n))$	$\nabla_{\mathbf{f}_n} \boldsymbol{\nu}(\mathbf{m}_n)^\top \boldsymbol{\Sigma}(\mathbf{m}_n)^{-1}(\mathbf{y}_n - \boldsymbol{\nu}(\mathbf{m}_n))$	$-\nabla_{\mathbf{f}_n} \boldsymbol{\nu}(\mathbf{m}_n)^\top \boldsymbol{\Sigma}(\mathbf{m}_n)^{-1} \nabla_{\mathbf{f}_n} \boldsymbol{\nu}(\mathbf{m}_n)$
PL	$\log \mathrm{N}(\mathbf{y}_n \,|\, \mathbb{E}_q[\boldsymbol{\nu}(\mathbf{f}_n)], \boldsymbol{\Omega}_{n,n})$	$\nabla_{\mathbf{m}_n} \mathbb{E}_q[\boldsymbol{\nu}(\mathbf{f}_n)]^\top \boldsymbol{\Omega}_{n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_q[\boldsymbol{\nu}(\mathbf{f}_n)])$	$-\nabla_{\mathbf{m}_n} \mathbb{E}_q[\boldsymbol{\nu}(\mathbf{f}_n)]^\top \boldsymbol{\Omega}_{n,n}^{-1} \nabla_{\mathbf{m}_n} \mathbb{E}_q[\boldsymbol{\nu}(\mathbf{f}_n)]$
VI	$\mathbb{E}_q[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)]$	$\nabla_{\mathbf{m}_n} \overline{\mathcal{L}}_n(\mathbf{m}_n, \mathbf{C}_{n,n})$	$\nabla^2_{\mathbf{m}_n} \overline{\mathcal{L}}_n(\mathbf{m}_n, \mathbf{C}_{n,n})$
PEP	$\frac{1}{\alpha} \log \mathbb{E}_{q^\backslash}[p(\mathbf{y}_n \,|\, \mathbf{f}_n)]$	$\mathbf{R}_n \nabla_{\mathbf{m}_n^\backslash} \overline{\mathcal{L}}(\mathbf{m}_n^\backslash, \mathbf{C}_{n,n}^\backslash)$	$\mathbf{R}_n \nabla^2_{\mathbf{m}_n^\backslash} \overline{\mathcal{L}}(\mathbf{m}_n^\backslash, \mathbf{C}_{n,n}^\backslash)$

Table 2: A comparison of the different method and approximation types employed by various Bayesian inference schemes. The Newton and Taylor methods are mode-seeking algorithms whose updates depend only on the posterior mean, whereas the Bayes–Newton methods (PL, PEP, VI) optimise the full posterior mean and covariance. It can be shown that Taylor and PL make Gauss–Newton approximations to the Hessian of the optimisation target.

	Method Type	Approximation Type
Newton/Laplace	Newton	Laplace approximation around posterior mode
Taylor/EKS	Gauss–Newton	Taylor expansion around posterior mode
PL	Bayes–Gauss–Newton	$\arg\min_{q(\mathbf{y} \,|\, \mathbf{f})} \mathbb{E}_{p(\mathbf{f} \,|\, \mathbf{y})}[\mathrm{D}_{\mathrm{KL}}\big[p(\mathbf{y} \,|\, \mathbf{f}) \parallel q(\mathbf{y} \,|\, \mathbf{f})\big]]$ Optimal likelihood linearisation in MSE sense
PEP	Bayes–Newton	$\approx \arg\min_{q(\mathbf{f})} \mathrm{D}_\alpha\big[p(\mathbf{f} \,|\, \mathbf{y}) \parallel q(\mathbf{f})\big]$ Stationary point of PEP Energy
VI	Bayes–Newton	$\arg\min_{q(\mathbf{f})} \mathrm{D}_{\mathrm{KL}}\big[q(\mathbf{f}) \parallel p(\mathbf{f} \,|\, \mathbf{y})\big]$ Minimizes variational free energy (VFE)

4.4 Method Comparison

Table 1 compares the surrogate targets used for updating the approximate likelihood parameters. Laplace (*i.e.*, Newton) and Taylor (*i.e.*, EKS) both compute the derivatives of the target with respect to $\mathbf{f}_n$ and then evaluate them at the mean, $\mathbf{f}_n = \mathbf{m}_n$. The Bayes–Newton methods (PL, PEP, VI) marginalise out $\mathbf{f}_n$ and then compute the gradients with respect to the posterior (or cavity) mean. Table 2 compares the types of approximations employed by the each inference algorithm, and Figure 1 is a flow chart showing the connections between the methods.

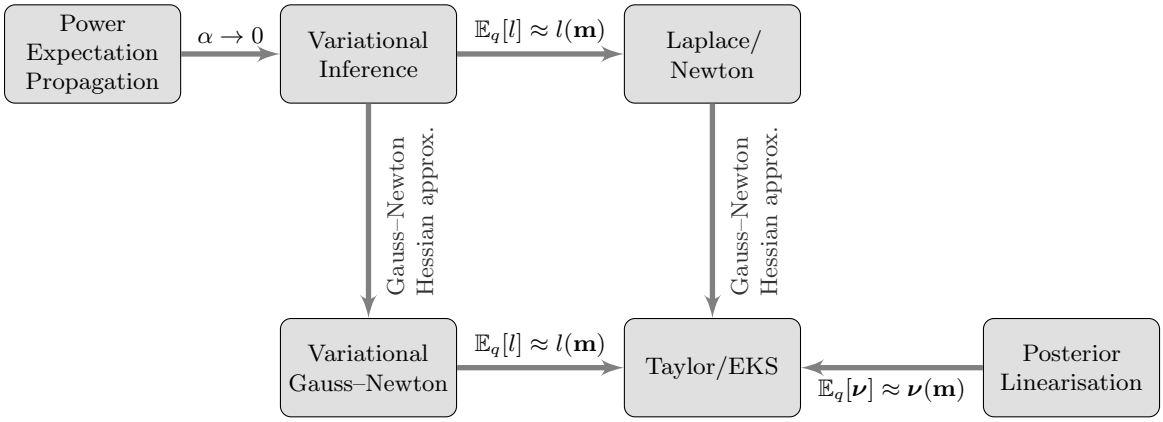

Figure 1: A graphical representation of the links between various inference methods. When the EP power tends to zero, the method becomes identical to natural gradient variational inference. Replacing the expectations of $l = \log p(\mathbf{y} \,|\, \mathbf{f})$ in the VI updates with point estimates leads to the Laplace approximation / Newton's method. Applying a (generalised) Gauss–Newton approximation to the Hessian of the target results in the Taylor method (extended Kalman smoother). Similarly, replacing the expectations of $\boldsymbol{\nu} = \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]$ in PL with point estimates leads to the Taylor method. In Section 5 we show that a Gauss–Newton approximation can also be applied to VI, which guarantees PSD covariance updates.

4.5 Issues with Newton and Bayes–Newton Methods

There are two sources of potential instability in the algorithms described above: *(i)* the algorithms that involve a full Hessian computation may result in the approximate likelihood covariance, $\overline{\mathbf{C}}$, being negative definite, such that the algorithm fails, *(ii)* the algorithms may oscillate or diverge. In the following sections, we present ways to address issue i) by developing Bayesian variants of approximation schemes from the optimisation literature, namely Gauss–Newton and Quasi-Newton methods.

Regarding issue *(ii)*, whilst a practical approach is to choose a relatively small step size, Newton's method with a constant step size is not guaranteed to converge, even for convex objective functions. It is therefore often used in conjunction with globalisation strategies such as line-search or trust region methods (Nocedal and Wright, 2006). We consider these strategies to be beyond the scope of this work. However, we hope that our approach motivates development of these ideas in the future.

4.5.1 Heuristic Methods for Ensuring PSD Updates

One common way to improve the conditioning of a matrix is to add a constant value to its diagonal entries. When applying Newton's method, this is a valid way to improve stability, because adding a infinitely large value to the diagonal of the Hessian results in the steepest descent algorithm. For the Bayes–Newton algorithms, adding a large constant to the diagonal is not valid, since the Hessian is also used as the approximate likelihood precision, so modifying its entries can lead to poor results.

Therefore we propose a simple heuristic approach to ensuring that updates remain PSD, which will serve as a baseline method for our Bayes–Gauss–Newton and Bayes-quasi-Newton methods presented in Section 5 and Section 6. Given a possibly indefinite precision matrix, $\overline{\mathbf{C}}^{-1}$, our heuristic approach sets all the off-diagonal entries to zero, and replaces all the negative diagonal entries with a small positive value, $\varepsilon = 0.01$.

5. Bayes–Gauss–Newton

Here we will first derive a Gauss–Newton approximation to Newton's method for approximate inference. After which, we will show how a similar Bayesian analogy results in an efficient and stable VI method that guarantees the likelihood covariance remain PSD, and show how this presentation provides insight into the PL and EKS methods. Unfortunately, it is not possible to derive a similar Gauss–Newton approximation to PEP, because Bonnet's and Price's theorems (see Section 5.2) do not apply to the PEP surrogate target.

5.1 The (Partial) Gauss–Newton Method for Approximate Bayesian Inference

Consider again Newton's method applied to the log posterior, $\mathcal{L}(\mathbf{f}) = \log p(\mathbf{f} \,|\, \mathbf{y})$, with a Gaussian prior, Equation (5). We now reformulate this algorithm as a *nonlinear least-squares problem*, the form required in order to derive and apply a Gauss–Newton approximation (Björck, 1996). This rewriting of the model into a least-squares form is a crucial step. However, it seems to have been neglected in the approximate inference literature (*e.g.*, Khan and Rue, 2021), where it is common to simply use the approximation $\nabla^2_{\mathbf{f}}\mathcal{L}(\mathbf{f}) \approx \nabla_{\mathbf{f}}\mathcal{L}(\mathbf{f})^\top \nabla_{\mathbf{f}}\mathcal{L}(\mathbf{f})$, which despite guaranteeing PSD updates is not a correct Gauss–Newton method.

We first consider the case where the likelihood is a Gaussian noise model of the form,

$$\begin{aligned}\log p(\mathbf{y} \,|\, \mathbf{f}) &= \log \mathrm{N}(\mathbf{y} \,|\, \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}], \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]) \\ &= \log Z(\mathbf{f}) - \frac{1}{2}(\mathbf{y} - \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}])^\top \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]^{-1}(\mathbf{y} - \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]),\end{aligned} \tag{33}$$

where $Z(\mathbf{f}) = (2\pi)^{-N/2}|\,\mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]|^{-1/2}$ is the normaliser, and where $\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]$ and $\mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]$ are the conditional moments, which may be nonlinear functions of $\mathbf{f}$ (making inference intractable). For example, the heteroscedastic noise likelihood (see Section 8.4.1) takes this form, as do many models from the signal processing literature involving dynamical systems corrupted by noise. For models of this type, we can define the vector $\mathbf{V}(\mathbf{f})$ to be a collection of *residual components* as follows,

$$\mathbf{V}(\mathbf{f}) = \begin{bmatrix} \mathrm{Cov}[\mathbf{y}_1 \,|\, \mathbf{f}_1]^{-\frac{1}{2}}(\mathbf{y}_1 - \mathbb{E}[\mathbf{y}_1 \,|\, \mathbf{f}_1]) \\ \vdots \\ \mathrm{Cov}[\mathbf{y}_N \,|\, \mathbf{f}_N]^{-\frac{1}{2}}(\mathbf{y}_N - \mathbb{E}[\mathbf{y}_N \,|\, \mathbf{f}_N]) \\ \mathbf{K}^{-\frac{1}{2}}(\mathbf{f} - \boldsymbol{\mu}) \end{bmatrix}, \tag{34}$$

such that the log posterior can be written $\log p(\mathbf{f} \,|\, \mathbf{y}) = -\frac{1}{2}\mathbf{V}(\mathbf{f})^\top\mathbf{V}(\mathbf{f}) + \log Z(\mathbf{f}) + c = -\frac{1}{2}\|\mathbf{V}(\mathbf{f})\|_2^2 + \log Z(\mathbf{f}) + c$, where $c = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{K}| - p(\mathbf{y})$ collects the terms that do not depend on $\mathbf{f}$ and so can be ignored. This shows that optimisation of the target can be cast as a *partial nonlinear least-squares problem*. We use the term 'partial' because the log normaliser may depend on $\mathbf{f}$, but cannot be written in the required least-squares form.

The Jacobian of the residual vector is obtained by taking the partial derivatives of each component,

$$\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f}) = \begin{bmatrix} \mathbf{G}_1 \\ \vdots \\ \mathbf{G}_N \\ \mathbf{K}^{-\frac{1}{2}} \end{bmatrix}, \tag{35}$$

where,

$$\begin{aligned} \mathbf{G}_n &= \nabla_{\mathbf{f}^\top} \left(\mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}} (\mathbf{y}_n - \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]) \right) \\ &= \left[\mathbf{0}, \ldots, \nabla_{\mathbf{f}_n} \left(\mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}} (\mathbf{y}_n - \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]) \right), \ldots, \mathbf{0} \right] \in \mathbb{R}^{D_{\mathbf{y}} \times ND}. \end{aligned} \tag{36}$$

The residual Hessian $\nabla^2_{\mathbf{f}} \mathbf{V}(\mathbf{f})$ is defined similarly. The Jacobian and Hessian of the log posterior can then be computed via the chain rule,

$$\begin{aligned} \nabla_{\mathbf{f}} \mathcal{L}(\mathbf{f}) &= -\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + \nabla_{\mathbf{f}} \log Z(\mathbf{f}), \\ \nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{f}) &= -\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f}) - \nabla^2_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + \nabla^2_{\mathbf{f}} \log Z(\mathbf{f}). \end{aligned} \tag{37}$$

A *Gauss–Newton* approximation to the Hessian involves discarding the second-order terms to give

$$\begin{aligned} \nabla^2_{\mathbf{f}} \mathcal{L}(\mathbf{f}) &\approx -\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f}) \\ &= -\sum_{n=1}^{N} \mathbf{G}_n^\top \mathbf{G}_n - \mathbf{K}^{-1}. \end{aligned} \tag{38}$$

Inference based on Equation (38) is guaranteed to result in PSD covariances because $\mathbf{G}_n^\top \mathbf{G}_n$ is PSD. Note that $\sum_{n=1}^{N} \mathbf{G}_n^\top \mathbf{G}_n$ is a block-diagonal matrix with block size D.

For many likelihood models the second-order term, $\nabla^2_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f})$, will be zero or close to zero, meaning that the Gauss–Newton approach is either exact or a good approximation. However, Gauss–Newton is arguably most desirable in scenarios where the second-order term is not close to zero, since this often causes non-PSD updates to occur.

Equation (38) also discards the log normaliser term, $\nabla^2_{\mathbf{f}} \log Z(\mathbf{f})$, which is often positive definite (undesirably so). For some models the normaliser $Z(\mathbf{f})$ does not depend on $\mathbf{f}$, and hence this term will be zero. Even if $Z(\mathbf{f})$ does depend on $\mathbf{f}$, we might still expect the residual term to dominate the Hessian computation.

5.1.1 Generalised Gauss–Newton for General Likelihoods

We now consider the case where the likelihood model cannot be written in the Gaussian form of Equation (33), for example in discrete models such as the Bernoulli or Poisson likelihoods, or continuous models such as the Gamma or Student-t likelihoods. In this case, the model can no longer be cast as a least-squares problem, so we resort to a *generalised Gauss–Newton* approach. The generalised approach, initially developed to address constrained optimisation problems (Golub and Pereyra, 1973), and more recently applied to Bayesian neural networks (Khan et al., 2019), works by defining a transformation of variables such that application of

the chain rule leads to a sum of first- and second-order terms. We propose the transformation $\mathbf{u}(\mathbf{f}) = \mathbb{E}[\mathbf{y} \mid \mathbf{f}]$, giving

$$\begin{aligned}
\nabla_{\mathbf{f}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) &= \nabla_{\mathbf{f}} \mathbf{u}(\mathbf{f})^\top \nabla_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})), \\
\nabla^2_{\mathbf{f}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) &= \nabla^2_{\mathbf{f}} \mathbf{u}(\mathbf{f})^\top \nabla_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) + \nabla_{\mathbf{f}}^\top \mathbf{u}(\mathbf{f}) \nabla^2_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) \nabla_{\mathbf{f}} \mathbf{u}(\mathbf{f}) \\
&\approx \nabla_{\mathbf{f}}^\top \mathbf{u}(\mathbf{f}) \nabla^2_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) \nabla_{\mathbf{f}} \mathbf{u}(\mathbf{f}),
\end{aligned} \tag{39}$$

where the final line amounts to the generalised Gauss–Newton approximation. However, $\nabla^2_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f}))$ is not guaranteed to be negative semi-definite as required. We therefore propose using the Laplace approximation, $\nabla^2_{\mathbf{u}} \log p(\mathbf{y} \mid \mathbf{u}(\mathbf{f})) \approx -\operatorname{Cov}[\mathbf{y} \mid \mathbf{f}]^{-1}$, inspired by the fact that for many models this Hessian term will be well approximated by the negative likelihood precision, which is guaranteed to be negative semi-definite. This leads to the following generalised Gauss–Newton method

$$\begin{aligned}
\nabla^2_{\mathbf{f}} \log p(\mathbf{y} \mid \mathbf{f}) &\approx -\nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \mid \mathbf{f}]^\top \operatorname{Cov}[\mathbf{y} \mid \mathbf{f}]^{-1} \nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \mid \mathbf{f}] \\
&= -\sum_{n=1}^{N} \mathbf{G}_n^\top \mathbf{G}_n,
\end{aligned} \tag{40}$$

where $\mathbf{G}_n = \operatorname{Cov}[\mathbf{y}_n \mid \mathbf{f}_n]^{-\frac{1}{2}} \nabla_{\mathbf{f}^\top} \mathbb{E}[\mathbf{y}_n \mid \mathbf{f}_n]$. This version matches the full Gauss–Newton method when $\nabla_{\mathbf{f}} \operatorname{Cov}[\mathbf{y} \mid \mathbf{f}]$ and $\nabla_{\mathbf{f}} \log Z(\mathbf{f})$ are both zero (or assumed to be zero). Interestingly, Equation (40) also matches the Taylor method / EKS updates in Equation (32) exactly. Next we will show that our presentation has the benefit of allowing for a variational Gauss–Newton extension, which improves on the EKS approach whilst still ensuring PSD updates.

5.2 Variational Gauss–Newton

As discussed in Section 4.1, natural gradient VI can be derived by defining the expected log likelihood, $\overline{\mathcal{L}}(\mathbf{m}, \mathbf{C}) = \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})]$, to be a surrogate optimisation target. Therefore, assuming the continuous Gaussian model in Equation (33) and letting

$$\mathbf{V}(\mathbf{f}) = \begin{bmatrix} \operatorname{Cov}[\mathbf{y}_1 \mid \mathbf{f}_1]^{-\frac{1}{2}} (\mathbf{y}_1 - \mathbb{E}[\mathbf{y}_1 \mid \mathbf{f}_1]) \\ \vdots \\ \operatorname{Cov}[\mathbf{y}_N \mid \mathbf{f}_N]^{-\frac{1}{2}} (\mathbf{y}_N - \mathbb{E}[\mathbf{y}_N \mid \mathbf{f}_N]) \end{bmatrix}, \tag{41}$$

the expected log likelihood can be written,

$$\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})] = \mathbb{E}_{q(\mathbf{f})} \left[-\frac{1}{2} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + \log Z(\mathbf{f}) \right] + c, \tag{42}$$

where c again collects the terms that do not depend on $\mathbf{f}$. By Bonnet's and Price's theorems (see Lin et al., 2019) we can write $\nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})] = \mathbb{E}_{q(\mathbf{f})}[\nabla_{\mathbf{f}} \log p(\mathbf{y} \mid \mathbf{f})]$, and

$\nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] = \mathbb{E}_{q(\mathbf{f})}[\nabla^2_{\mathbf{f}} \log p(\mathbf{y} \,|\, \mathbf{f})]$, such that

$$
\begin{aligned}
\nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] &= \mathbb{E}_{q(\mathbf{f})}[-\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + \nabla_{\mathbf{f}} \log Z(\mathbf{f})], \\
\nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] &= \mathbb{E}_{q(\mathbf{f})}[-\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f}) - \nabla^2_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + \nabla^2_{\mathbf{f}} \log Z(\mathbf{f})] \\
&\approx \mathbb{E}_{q(\mathbf{f})}\left[-\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})^\top \nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f})\right] \\
&= \mathbb{E}_{q(\mathbf{f})}\left[-\sum_{n=1}^{N} \mathbf{G}_n^\top \mathbf{G}_n\right],
\end{aligned}
\tag{43}
$$

where $\mathbf{G}_n = \nabla_{\mathbf{f}^\top}\Big(\mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}}(\mathbf{y}_n - \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n])\Big)$. Here $\mathbf{G}_n^\top \mathbf{G}_n$ is guaranteed to be PSD such that $\mathbb{E}_{q(\mathbf{f}_n)}\left[\mathbf{G}_n^\top \mathbf{G}_n\right]$ is also PSD. We refer to Equation (43) as the *variational Gauss–Newton* method.

As above, for discrete likelihoods we again propose the use of $\mathbf{G}_n = \mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}} \nabla_{\mathbf{f}^\top} \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]$, leading to the following approximation,

$$
\nabla^2_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})] \approx -\mathbb{E}_{q(\mathbf{f})}\left[\nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}] \,\mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]^{-1} \nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]\right], \tag{44}
$$

that we refer to as the *variational generalised Gauss–Newton* method. Equation (44) is equivalent to the Taylor/EKS method where we take an expectation with respect to the full posterior rather than simply evaluating the Hessian at the posterior mean. It also bears an interesting resemblance to the PL updates, Equation (31), making it clear that whilst PL and variational Gauss–Newton are both Bayes–Gauss–Newton methods, PL makes additional limiting assumptions and approximates the likelihood covariance differently.

The variational Gauss–Newton method is guaranteed to produce PSD updates even in the presence of numerical integration error when computing $\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})]$, and in scenarios where the model hyperparameters change across iterations. This gives it a significant advantage relative to the PSD constraints based on Riemannian gradients in Section 7.

5.3 Second-Order Posterior Linearisation

In Appendix F, we show how the PL updates in Equation (31) amount to a Bayes–Gauss–Newton approximation, with the Hessian computation only involving first-order derivatives. Additionally, the covariance, $\mathbf{\Omega}_{n,n}$, is assumed to have zero gradient with respect to $\mathbf{m}_n$. This means PL results in a poor approximation for likelihood models of the form given in Equation (33) where $\mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]$ depends on $\mathbf{f}$. We propose a full Newton-like version of PL which takes into account this dependency and includes the second-order terms. Recall that the PL surrogate target is $\overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) = \log \mathrm{N}(\mathbf{y}_n \,|\, \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]],\, \mathbf{\Omega}_{n,n})$. Letting Z_n be the normaliser of this Gaussian gives

$$
\left.
\begin{aligned}
\mathbf{D}_{k,n} &= \mathbf{\Omega}_{k,n,n}^{-\frac{1}{2}}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]) \\
\overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) &= \log Z_n - \tfrac{1}{2}\mathbf{D}_n^\top \mathbf{D}_n \\
\mathbf{J}_{k,n} &= \nabla_{\mathbf{m}_n} \log Z_n - \nabla_{\mathbf{m}_n} \mathbf{D}_{k,n}^\top \mathbf{D}_{k,n} \\
\mathbf{H}_{k,n,n} &= \nabla^2_{\mathbf{m}_n} \log Z_n - \nabla_{\mathbf{m}_n} \mathbf{D}_{k,n}^\top \nabla_{\mathbf{m}_n} \mathbf{D}_{k,n} - \nabla^2_{\mathbf{m}_n} \mathbf{D}_{k,n}^\top \mathbf{D}_{k,n}
\end{aligned}
\right\} \begin{array}{l} \text{surrogate target} \\ \text{\& gradients} \\ (2^{\text{nd}}\text{-order PL}) \end{array}
\tag{45}
$$

Following the approach above, a Gauss–Newton approximation can then be derived by setting

$$\mathbf{H}_{k,n,n} = -\nabla_{\mathbf{m}_n}\mathbf{D}_{k,n}^\top \nabla_{\mathbf{m}_n}\mathbf{D}_{k,n}, \tag{46}$$

which improves upon standard PL when $\mathrm{Cov}[\mathbf{y} \mid \mathbf{f}]$ depends on $\mathbf{f}$, and still guarantees PSD updates. If the likelihood model is not of the form given in Equation (33), for example if it is discrete, then a generalised Gauss–Newton approximation in which the normaliser Z_n and the gradients of $\mathbf{\Omega}_{k,n,n}$ are ignored results in exactly the standard PL method.

6. Bayes-Quasi-Newton

As discussed in Section 5.1, Gauss–Newton methods are accurate when the Hessian computation is dominated by the first-order term. If this is not the case, a better approximation may be a quasi-Newton method (Broyden, 1967; Nocedal and Wright, 2006). Quasi-Newton methods replace the full Hessian computation with a series of efficient low-rank updates, and whilst these updates are not guaranteed to result in PSD covariances, they do provide a way of checking whether a given update will be PSD, which can be used to determine whether an update should be applied. A *damped* version of the updates can also be applied, which does guarantee that the resulting covariances are PSD.

6.1 The Quasi-Newton Method

Quasi-Newton methods approximate the Hessian of the optimisation target, $\nabla^2\mathcal{L}$, via a matrix, $\mathbf{B}$, in a way that avoids the Hessian computation by utilising changes in first-order derivative information along the optimisation search direction. The application of the quasi-Newton method to the global Newton updates, Equation (6), whilst extremely efficient, is a poor approximation because the full-rank matrix $\mathbf{C}^{-1}$ cannot be well approximated by iterative low-rank updates. Therefore, we apply individual quasi-Newton updates to the local approximate likelihood terms, $\overline{\mathcal{L}}(\mathbf{f}_n) = \log p(\mathbf{y}_n \mid \mathbf{f}_n)$:

$$\nabla^2_{\mathbf{f}_n}\overline{\mathcal{L}}(\mathbf{f}_n) \approx \mathbf{B}_{k,n}. \tag{47}$$

The matrix $\mathbf{B}_{k,n}$ is chosen to ensure it satisfies the *secant equation*: $\mathbf{B}_{k+1,n}\mathbf{s}_{k,n} = \mathbf{g}_{k,n}$, where $\mathbf{s}_{k,n} = \mathbf{m}_{k+1,n} - \mathbf{m}_{k,n}$ and $\mathbf{g}_{k,n} = \nabla_{\mathbf{f}_n}\overline{\mathcal{L}}(\mathbf{m}_{k+1,n}) - \nabla_{\mathbf{f}_n}\overline{\mathcal{L}}(\mathbf{m}_{k,n})$. Various methods exist for doing do, but we focus on the *BFGS* formula (Broyden, 1969; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970), because it has been shown empirically to be the most effective, and because it guarantees that $\mathbf{B}_{k,n}$ remains negative semi-definite whenever the initial value, $\mathbf{B}_{0,n}$, is negative semi-definite and $\mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} < 0$.

The BFGS formula, which amounts to a rank-two update to $\mathbf{B}_{k,n}$, is given by

$$\mathbf{B}_{k+1,n} = \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{g}_{k,n}\mathbf{g}_{k,n}^\top}{\mathbf{s}_{k,n}^\top\mathbf{g}_{k,n}}. \tag{48}$$

The curvature condition, $\mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} < 0$, is not guaranteed for general nonlinear functions, and we observe it to often be violated for non-log-concave likelihood models. A practical approach to ensure stability is to simply reject updates that do no satisfy the condition.

The local quasi-Newton updates are then

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\mathbf{f}_n) &= \log p(\mathbf{y}_n \,|\, \mathbf{f}_n) \\
\mathbf{J}_{k,n} &= \nabla_{\mathbf{f}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}) \\
\mathbf{H}_{k,n,n} &= \mathbf{B}_{k,n} \\
\mathbf{s}_{k,n} &= \mathbf{m}_{k+1,n} - \mathbf{m}_{k,n}\,, \quad \mathbf{g}_{k,n} = \nabla_{\mathbf{f}_n} \overline{\mathcal{L}}(\mathbf{m}_{k+1,n}) - \nabla_{\mathbf{f}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}) \\
\mathbf{B}_{k+1,n} &= \begin{cases} \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{g}_{k,n}\mathbf{g}_{k,n}^\top}{\mathbf{s}_{k,n}^\top\mathbf{g}_{k,n}}, & \text{if } \mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} < 0 \\ \mathbf{B}_{k,n}\,, & \text{if } \mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} \geq 0 \end{cases}
\end{aligned}\right\} \text{ surrogate target \& gradients (Quasi-Newton)} \tag{49}$$

In some scenarios this approach can cause a large number of the updates to be rejected. We present a damped version whose updates are guaranteed to be stable in Section 6.5. Next we will derive two Bayesian variants of the quasi-Newton method corresponding to approximations to VI and PEP.

6.2 Variational Quasi-Newton

The quasi-Newton method cannot be straightforwardly applied to VI by simply changing the target density, because the parameters being optimised in Bayes–Newton methods include the posterior covariance, $\mathbf{C}$, as well as the posterior mean, $\mathbf{m}$, and the changes in covariance must be accounted for in the secant equation. Since the surrogate target, $\overline{\mathcal{L}}(\mathbf{m}, \mathbf{C}) = \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y} \,|\, \mathbf{f})]$, only depends on the marginal variances ($\mathbf{y}_n$ only depends on $\mathbf{f}_n$) it is sufficient to define a new vector,

$$\boldsymbol{\eta}_n = \begin{pmatrix} \mathbf{m}_n \\ \mathrm{vec}(\mathbf{C}_{n,n}) \end{pmatrix} \in \mathbb{R}^{(D+D^2)\times 1}, \tag{50}$$

where $\mathrm{vec}(\mathbf{C}_{n,n}) \in \mathbb{R}^{D^2\times 1}$ represents the covariance of the marginal, $q(\mathbf{f}_n)$, stored as a column vector. We must then compute the Jacobian of the target with respect to $\boldsymbol{\eta}_n$. Since computing the Jacobian with respect to the covariance comes at the same computational cost as computing the Hessian with respect to the mean, Bayes-quasi-Newton methods are not computationally more efficient than standard Bayes–Newton methods and should be viewed instead as a way to ensure PSD updates.

The *variational quasi-Newton* method is therefore characterised by the following updates,

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\boldsymbol{\eta}_n) &= \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\
\mathbf{J}_{k,n} &= \nabla_{\mathbf{m}_{k,n}} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}) \\
\mathbf{H}_{k,n,n} &= \mathbf{B}_{k,n,1:D,1:D} \\
\mathbf{s}_{k,n} &= \boldsymbol{\eta}_{k+1,n} - \boldsymbol{\eta}_{k,n}\,, \quad \mathbf{g}_{k,n} = \nabla_{\boldsymbol{\eta}_n} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k+1,n}) - \nabla_{\boldsymbol{\eta}_n} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}) \\
\mathbf{B}_{k+1,n} &= \begin{cases} \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{g}_{k,n}\mathbf{g}_{k,n}^\top}{\mathbf{s}_{k,n}^\top\mathbf{g}_{k,n}}, & \text{if } \mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} < 0 \\ \mathbf{B}_{k,n}\,, & \text{if } \mathbf{s}_{k,n}^\top\mathbf{g}_{k,n} \geq 0 \end{cases}
\end{aligned}\right\} \text{ surrogate target \& gradients (Variational Quasi-Newton)} \tag{51}$$

where $\mathbf{B}_{k,n,1:D,1:D}$ represents the upper left $D \times D$ block of $\mathbf{B}_{k,n} \in \mathbb{R}^{(D+D^2)\times(D+D^2)}$, which corresponds to the approximate Hessian with respect to the mean, $\mathbf{m}_{k,n}$.

6.3 Power Expectation Propagation Quasi-Newton

We take a similar approach to derive a quasi-Newton approximation to power EP. In this case, we define a new vector which stacks the cavity means and marginal cavity variances,

$$\boldsymbol{\eta}_n^{\setminus} = \begin{pmatrix} \mathbf{m}_n^{\setminus} \\ \mathrm{vec}(\mathbf{C}_{n,n}^{\setminus}) \end{pmatrix} \in \mathbb{R}^{(D+D^2)\times 1}, \tag{52}$$

where $\mathrm{vec}(\mathbf{C}_{n,n}^{\setminus})$ is a vectorised version of $\mathbf{C}_{n,n}^{\setminus}$. The updates are then

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\boldsymbol{\eta}_n^{\setminus}) &= \tfrac{1}{\alpha}\log \mathbb{E}_{q^{\setminus}(\mathbf{f}_n)}[p^{\alpha}(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\
\mathbf{R}_{k,n} &= \mathbf{C}_{k,n,n}^{\setminus -1}\left(\alpha \mathbf{B}_{k,n,1:D,1:D} + \mathbf{C}_{k,n,n}^{\setminus -1}\right)^{-1} \\
\mathbf{J}_{k,n} &= \mathbf{R}_{k,n}\nabla_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}^{\setminus}) \\
\mathbf{H}_{k,n,n} &= \mathbf{R}_{k,n}\mathbf{B}_{k,n,1:D,1:D} \\
\mathbf{s}_{k,n} &= \boldsymbol{\eta}_{k+1,n}^{\setminus} - \boldsymbol{\eta}_{k,n}^{\setminus}\,, \quad \mathbf{g}_{k,n} = \nabla_{\boldsymbol{\eta}_n^{\setminus}} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k+1,n}^{\setminus}) - \nabla_{\boldsymbol{\eta}_n^{\setminus}} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}^{\setminus}) \\
\mathbf{B}_{k+1,n} &= \begin{cases} \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{g}_{k,n}\mathbf{g}_{k,n}^{\top}}{\mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n}}, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} < 0 \\ \mathbf{B}_{k,n}\,, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} \geq 0 \end{cases}
\end{aligned}\right\} \begin{array}{l}\text{surrogate target} \\ \text{\& gradients} \\ \text{(PEP Quasi-Newton)}\end{array} \tag{53}$$

6.4 Posterior Linearisation Quasi-Newton

A quasi-Newton approximation can also be applied to PL in a very similar way to the VI case:

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\boldsymbol{\eta}_n) &= \log \mathrm{N}(\mathbf{y}_n \,|\, \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]],\, \boldsymbol{\Omega}_{n,n}) \\
\mathbf{J}_{k,n} &= \nabla_{\mathbf{m}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]^{\top} \boldsymbol{\Omega}_{k,n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]) \\
\mathbf{H}_{k,n,n} &= \mathbf{B}_{k,n,1:D,1:D} \\
\mathbf{s}_{k,n} &= \boldsymbol{\eta}_{k+1,n} - \boldsymbol{\eta}_{k,n}\,, \quad \mathbf{g}_{k,n} = \nabla_{\boldsymbol{\eta}_n} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k+1,n}) - \nabla_{\boldsymbol{\eta}_n} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}) \\
\mathbf{B}_{k+1,n} &= \begin{cases} \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{g}_{k,n}\mathbf{g}_{k,n}^{\top}}{\mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n}}, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} < 0 \\ \mathbf{B}_{k,n}\,, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} \geq 0 \end{cases}
\end{aligned}\right\} \begin{array}{l}\text{surrogate target} \\ \text{\& gradients} \\ \text{(PL Quasi-Newton)}\end{array} \tag{54}$$

where $\nabla_{\boldsymbol{\eta}_n} \overline{\mathcal{L}}(\boldsymbol{\eta}_{k,n}) = \nabla_{\boldsymbol{\eta}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]^{\top} \boldsymbol{\Omega}_{k,n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]])$, *i.e.*, the gradient of $\boldsymbol{\Omega}_{n,n}$ is assumed to be zero. This approach can also be adapted to the improved version of PL derived in Section 5.3 by relaxing this assumption (and including the normalisation constant if appropriate).

6.5 Damped Quasi-Newton Updates

For complicated likelihood models, the BFGS updates described above may result in many updates being rejected, which can result in poor performance. The following *damped* BFGS formula can be used as a slightly less accurate approach which guarantees PSD updates. The adaptive damping term, $\boldsymbol{\psi}$, is introduced as follows:

$$\boldsymbol{\psi}_{k,n} = \begin{cases} 1\,, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} \leq (1-\xi)\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n} \\ \xi \frac{\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n}}{\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n} - \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n}}\,, & \text{if } \mathbf{s}_{k,n}^{\top}\mathbf{g}_{k,n} > (1-\xi)\mathbf{s}_{k,n}^{\top}\mathbf{B}_{k,n}\mathbf{s}_{k,n} \end{cases} \tag{55}$$

where ξ is the damping factor (often set to $\xi = 0.8$). Then letting

$$\mathbf{r}_{k,n} = \boldsymbol{\psi}\mathbf{g}_{k,n} + (1 - \boldsymbol{\psi})\mathbf{B}_{k,n}\mathbf{s}_{k,n}, \tag{56}$$

the BFGS update is modified to use $\mathbf{r}_{k,n}$ in place of $\mathbf{g}_{k,n}$:

$$\mathbf{B}_{k+1,n} = \mathbf{B}_{k,n} - \frac{\mathbf{B}_{k,n}\mathbf{s}_{k,n}\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}}{\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}\mathbf{s}_{k,n}} + \frac{\mathbf{r}_{k,n}\mathbf{r}_{k,n}^\top}{\mathbf{s}_{k,n}^\top\mathbf{r}_{k,n}}, \tag{57}$$

which is guaranteed to be negative semi-definite because $\mathbf{s}_{k,n}^\top\mathbf{r}_{k,n} = -(1-\xi)\mathbf{s}_{k,n}^\top\mathbf{B}_{k,n}\mathbf{s}_{k,n} < 0$. When $\boldsymbol{\psi}_{k,n} = 1$, the update is equivalent to the standard BFGS formula. When $\boldsymbol{\psi}_{k,n} \in (0, 1)$, the update results in a negative semi-definite $\mathbf{B}_{k+1,n}$ that interpolates between the previous estimate and the standard update. The damped approach is also applicable to the VI and PEP quasi-Newton variants.

Perhaps a more common approach to ensuring PSD updates during quasi-Newton is to the use the Wolfe line-search (Nocedal and Wright, 2006). However, since our approach is to apply the BFGS updates to the individual likelihood terms separately, whereas the objective (the model energy) is global, we found that a very small step size is often required to ensure that *all* terms remain PSD. In contrast, the damping approach effectively allows a different step size for each term, which is much more effective.

7. PSD Constraints via Riemannian Gradients

Lin et al. (2020) used Riemannian gradient methods to deal with the PSD constraint in VI by adding an additional term to the precision update, in a similar vein to the 'retraction map' presented in Tran et al. (2019). Since the prior precision is guaranteed to be PSD by construction, we derive the corresponding method applied to the approximate likelihood updates by subtracting the prior component from the method of Lin et al. (2020) to give

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) &= \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\
\mathbf{J}_{k,n} &= \nabla_{\mathbf{m}_n}\overline{\mathcal{L}}(\mathbf{m}_{k,n}, \mathbf{C}_{k,n,n}) \\
\mathbf{G}_{k,n} &= \overline{\mathbf{C}}_{k,n,n}^{-1} + \nabla^2_{\mathbf{m}_n}\overline{\mathcal{L}}(\mathbf{m}_{k,n}, \mathbf{C}_{k,n,n}) \\
\mathbf{H}_{k,n,n} &= \nabla^2_{\mathbf{m}_n}\overline{\mathcal{L}}(\mathbf{m}_{k,n}, \mathbf{C}_{k,n,n}) - \tfrac{\rho}{2}\mathbf{G}_{k,n}\overline{\mathbf{C}}_{k,n,n}\mathbf{G}_{k,n}
\end{aligned}\right\} \begin{array}{l}\text{surrogate target} \\ \text{\& gradients} \\ \text{(VI Riemann)}\end{array} \tag{58}$$

This approach provides excellent performance in many cases, although the degree to which the constraint may affect the quality of the approximation is unclear. Unfortunately, despite having theoretical guarantees that Equation (58) results in PSD updates, in practice we find that numerical integration error when computing $\overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n})$ can cause the algorithm to fail. The constraint is also not guaranteed if the model hyperparameters change across optimisation iterations.

Adding a similar retraction map term to the improved PL method, Equation (45), is also valid, and by approximating the target expectation with a point estimate, $\overline{\mathcal{L}}(\mathbf{m}_n) = \log p(\mathbf{y}_n \,|\, \mathbf{m}_n)$, it is straightforward to derive a Newton / Laplace version. Our unifying

presentation also allows us to formulate a similar PSD constraint for PEP as follows:

$$\left.\begin{aligned}
\overline{\mathcal{L}}(\mathbf{m}_n^{\setminus}, \mathbf{C}_{n,n}^{\setminus}) &= \tfrac{1}{\alpha} \log \mathbb{E}_{q^{\setminus}(\mathbf{f}_n)}[p^{\alpha}(\mathbf{y}_n \,|\, \mathbf{f}_n)] \\
\mathbf{R}_{k,n} &= \mathbf{C}_{k,n,n}^{\setminus -1} \left(\alpha \nabla^2_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) + \mathbf{C}_{k,n,n}^{\setminus -1} \right)^{-1} \\
\mathbf{G}_{k,n} &= \overline{\mathbf{C}}_{k,n,n}^{-1} + \mathbf{R}_{k,n} \nabla^2_{\mathbf{m}_n} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) \\
\mathbf{J}_{k,n} &= \mathbf{R}_{k,n} \nabla_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) \\
\mathbf{H}_{k,n,n} &= \mathbf{R}_{k,n} \nabla^2_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) - \tfrac{\rho}{2} \mathbf{G}_{k,n} \overline{\mathbf{C}}_{k,n,n} \mathbf{G}_{k,n}
\end{aligned}\right\} \begin{array}{l}\text{surrogate target} \\ \text{\& gradients} \\ \text{(PEP Riemann)}\end{array} \tag{59}$$

These constraints for the PEP updates only hold if $\left(\alpha \nabla^2_{\mathbf{m}_n^{\setminus}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\setminus}, \mathbf{C}_{k,n,n}^{\setminus}) + \mathbf{C}_{k,n,n}^{\setminus -1}\right)$ is PSD, which is not guaranteed to be the case. However, in practice, we find the method greatly reduces the chance of the update resulting in a negative-definite covariance. We find that most of the practical issues involved with these PSD constraints for both VI and PEP can be alleviated by setting the learning rate ρ to a small value. Reducing the power, α, in the PEP case also improves stability.

8. Examples, Experiments, and Connections to Related Work

As we have shown, all approximate inference schemes can be cast as updates to local approximate likelihoods, Equation (9), combined with conjugate global parameter updates, Equation (10). For this reason, the choice of approximate inference scheme is completely independent of the choice of model, and of the approach used to compute the global updates. To illustrate this point, we now demonstrate how to apply the inference schemes when the latent variable $\mathbf{f}$ is characterised by a Gaussian process, a sparse Gaussian process, or a state space model. The full algorithms are described in Appendix H.

8.1 Gaussian Processes

A Gaussian process (GP, Rasmussen and Williams, 2006) is a distribution over functions, which states that realisations of a function, $f(x)$, at any finite collection of inputs, $\mathbf{X} \in \mathbb{R}^{N \times D_{\mathbf{x}}}$, is jointly Gaussian distributed, $\mathbf{f} = f(\mathbf{X}) \sim \mathrm{N}(f(\mathbf{X}) \,|\, \boldsymbol{\mu} = \mu(\mathbf{X}), \mathbf{K} = \kappa(\mathbf{X}, \mathbf{X}))$. A GP prior, written $f(x) \sim \mathrm{GP}(\mu(x), \kappa(x, x'))$, is characterised by a mean function, $\mu(x)$, and a covariance function, $\kappa(x, x')$, and sophisticated domain knowledge can be incorporated into μ and κ, which may have some hyperparameters, $\boldsymbol{\theta}$, associated with them.

If a GP prior is combined with a Gaussian likelihood, $p(\mathbf{y} \,|\, \mathbf{f}) = \mathrm{N}(\mathbf{y} \,|\, \mathbf{f}, \overline{\mathbf{C}})$, then the application of Bayes' rule allows us to obtain the posterior distribution, $p(\mathbf{f} \,|\, \mathbf{y}) \propto p(\mathbf{f})\, p(\mathbf{y} \,|\, \mathbf{f})$, whose mean and covariance are given by (assuming the prior has zero mean, $\boldsymbol{\mu} = \mathbf{0}$),

$$\begin{aligned}
\mathbf{m} &= \mathbf{K}(\mathbf{K} + \overline{\mathbf{C}})^{-1} \mathbf{y}, \\
\mathbf{C} &= \mathbf{K} - \mathbf{K}(\mathbf{K} + \overline{\mathbf{C}})^{-1} \mathbf{K}.
\end{aligned} \tag{60}$$

When $p(\mathbf{y} \,|\, \mathbf{f})$ is non-Gaussian, the approximate inference schemes presented above can be used by replacing the true likelihood with an *approximate likelihood*, $t(\mathbf{f}) = \mathrm{N}(\mathbf{f} \,|\, \overline{\mathbf{m}}, \overline{\mathbf{C}})$. In this case, by setting $\mathbf{y} = \overline{\mathbf{m}}$, the above equations are equivalent to, but slightly more stable than, the updates given in Equation (10). Additionally, GPs allow for predictions to be

made at 'test' locations, $\mathbf{X}^*$, by exploiting the marginalisation and conditional properties of Gaussian densities. In Section 8.4 we also consider the case where f is vector-valued with D elements: $f(x) : \mathbb{R}^{D_\mathbf{x}} \to \mathbb{R}^D$.

Inference in non-conjugate GP models therefore involves the iterative application of a local update scheme for $t(\mathbf{f})$, followed by Equation (9), and Equation (60) with $\mathbf{y} = \overline{\mathbf{m}}$. Algorithm 1 gives the full algorithm. These iterations can be alternated with updates to the hyperparameters, $\boldsymbol{\theta}$, via gradient-based optimisation of the model energy (see Section 4). If the VI updates of Equation (15) are used, we recover a natural gradient version of the *variational GP* (Opper and Archambeau, 2009). The PEP updates of Equation (22) recover the approach of Minka (2001). Newton's method, Equation (8), results in the Laplace approach described in Rasmussen and Williams (2006). PL, Equation (31), recovers García-Fernández et al. (2019).

8.2 Sparse Gaussian Processes

The global update in Equation (60) has $\mathcal{O}(N^3D^3)$ computational scaling, which can be prohibitive. A common approach to reducing this scaling is the *sparse GP* (Csató and Opper, 2002). Sparse GPs introduce a set of *inducing inputs*, $\mathbf{Z} \in \mathbb{R}^{M \times D_\mathbf{x}}$, where $M < N$. The *inducing variables*, $\mathbf{u} = f(\mathbf{Z})$, have prior $p(\mathbf{u}) = \mathrm{N}(\mathbf{u} \,|\, \boldsymbol{\mu}_\mathbf{u} = \mu(\mathbf{Z}), \mathbf{K}_{\mathbf{uu}} = \kappa(\mathbf{Z}, \mathbf{Z}))$.

The posterior over the inducing variables, $q(\mathbf{u}) = \mathrm{N}(\mathbf{u} \,|\, \mathbf{m}_\mathbf{u}, \mathbf{C}_\mathbf{u})$, is stored in memory, from which the approximate posterior marginals can be obtained by conditioning: $q(\mathbf{f}_n) = \int p(\mathbf{f}_n \,|\, \mathbf{u}) q(\mathbf{u}) \, \mathrm{d}\mathbf{u}$, where the conditional is Gaussian: $p(\mathbf{f}_n \,|\, \mathbf{u}) = \mathrm{N}(\mathbf{f}_n \,|\, \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{u}, \mathbf{K}_{n,n} - \mathbf{K}_{\mathbf{f}_n\mathbf{u}}\mathbf{W}_{\mathbf{f}_n\mathbf{u}}^\top)$, for $\mathbf{W}_{\mathbf{f}_n\mathbf{u}} = \mathbf{K}_{\mathbf{f}_n\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}$, and $\mathbf{K}_{\mathbf{f}_n\mathbf{u}} = \kappa(\mathbf{X}_n, \mathbf{Z})$. This leads to computational savings because $\mathbf{f}_i$ and $\mathbf{f}_j$ are now conditionally independent given $\mathbf{u}$. The approximate likelihoods are also redefined to be functions of $\mathbf{u}$,

$$t(\mathbf{u}) = \prod_{n=1}^{N} t_n(\mathbf{u}). \tag{61}$$

In order to update the parameters of $t_n(\mathbf{u})$ we must compute the gradients of the surrogate target (see Table 1), but where the posterior marginal, $q(\mathbf{f}_n)$, is computed via the conditional. For example, the sparse VI surrogate target becomes $\mathbb{E}_{q(\mathbf{u})}[\mathbb{E}_{p(\mathbf{f}_n \,|\, \mathbf{u})}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)]]$ and we compute its gradients with respect to the mean of $t_n(\mathbf{u})$. Fortunately, *for all* inference schemes this quantity amounts to applying the standard update rules to $t(\mathbf{f}_n)$ followed by a deterministic mapping: recalling that $\overline{\boldsymbol{\lambda}}^{(1)}$, $\overline{\boldsymbol{\lambda}}^{(2)}$, are the natural parameters of $t(\mathbf{f})$, the corresponding natural parameters of $t(\mathbf{u}) = z_\mathbf{u}\mathrm{N}(\mathbf{u} \,|\, \overline{\mathbf{m}}_\mathbf{u}, \overline{\mathbf{C}}_\mathbf{u})$ are

$$\begin{aligned} \overline{\boldsymbol{\lambda}}_\mathbf{u}^{(1)} &= \mathbf{W}_{\mathbf{fu}}^\top \overline{\boldsymbol{\lambda}}^{(1)}, \\ \overline{\boldsymbol{\lambda}}_\mathbf{u}^{(2)} &= \mathbf{W}_{\mathbf{fu}}^\top \overline{\boldsymbol{\lambda}}^{(2)} \mathbf{W}_{\mathbf{fu}} \,. \end{aligned} \tag{62}$$

Note that $\overline{\boldsymbol{\lambda}}_\mathbf{u}^{(2)} \in \mathbb{R}^{MD \times MD}$ is a dense matrix, whereas $\overline{\boldsymbol{\lambda}}^{(2)} \in \mathbb{R}^{ND \times ND}$ is block-diagonal. To update the inducing posterior we now apply a modified version of Equation (60),

$$\begin{aligned} \mathbf{m}_\mathbf{u} &= \mathbf{K}_{\mathbf{uu}}(\mathbf{K}_{\mathbf{uu}} + \overline{\mathbf{C}}_\mathbf{u})^{-1}\overline{\mathbf{m}}_\mathbf{u}, \\ \mathbf{C}_\mathbf{u} &= \mathbf{K}_{\mathbf{uu}} - \mathbf{K}_{\mathbf{uu}}(\mathbf{K}_{\mathbf{uu}} + \overline{\mathbf{C}}_\mathbf{u})^{-1}\mathbf{K}_{\mathbf{uu}}. \end{aligned} \tag{63}$$

Updates to the local factor, $t(\mathbf{f}_n)$, require the posterior marginal $q(\mathbf{f}_n) = \mathrm{N}(\mathbf{f}_n \,|\, \mathbf{m}_n, \mathbf{C}_{n,n})$, which is given by

$$\begin{aligned} \mathbf{m}_n &= \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{m}_\mathbf{u}, \\ \mathbf{C}_{n,n} &= \mathbf{K}_{n,n} - \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{K}_{\mathbf{f}_n\mathbf{u}}^\top + \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{C}_\mathbf{u}\mathbf{W}_{\mathbf{f}_n\mathbf{u}}^\top. \end{aligned} \tag{64}$$

This algorithm has $\mathcal{O}(NM^2D^3)$ dominant computational scaling, and can be combined with any of the inference methods presented above by choosing the appropriate update rule for $t(\mathbf{f}_n)$. The full algorithm is given in Algorithm 2. Use of the VI updates of Equation (15) leads to the method of Adam et al. (2021), which they show to be an improved version of natural gradient inference for the *sparse variational GP* (Hensman et al., 2015; Salimbeni et al., 2018). The PEP updates of Equation (22) lead to sparse PEP (Bui et al., 2017).

If the local factors, $t(\mathbf{f}_n)$, are stored in memory, then the overall memory requirement scales as $\mathcal{O}(ND + M^2D^2)$. However, for all algorithms apart from PEP (where the local factors must be kept in order to compute the cavities), it's possible to instead store only $t(\mathbf{u})$, leading to memory scaling of $\mathcal{O}(M^2D^2)$. The inducing inputs, $\mathbf{Z}$, can be treated as hyperparameters and optimised based on the model energy. Crucially, a *stochastic* version of the sparse algorithm can be obtained by updating only a subset of the local factors on each iteration, which is particularly efficient since the marginals of $q(\mathbf{f})$ can be computed independently given $q(\mathbf{u})$. In this case, the parameters for each factor are 'tied': the likelihood contribution for each data point is simply given by $t_n(\mathbf{u}) = t^{N^{-1}/N}(\mathbf{u})$. This approach is used with VI in Adam et al. (2021), and when used with PEP it leads to *stochastic PEP* (Li et al., 2015) which also has memory scaling $\mathcal{O}(M^2D^2)$.

8.2.1 The Sparse GP Energy

The GP energy can also be efficiently approximated via the sparse model. The sparse variational free energy (see Section 4.1.1) is given by

$$\mathrm{VFE}(q(\mathbf{u})) = -\sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{u})}[\mathbb{E}_{p(\mathbf{f}_n \,|\, \mathbf{u})}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)]] + \mathbb{E}_{q(\mathbf{u})}[\log \mathrm{N}(\mathbf{u} \,|\, \overline{\mathbf{m}}_\mathbf{u}, \overline{\mathbf{C}}_\mathbf{u})] - \log \mathcal{Z}_\mathbf{u}, \tag{65}$$

with $\log \mathcal{Z}_\mathbf{u} = \int p(\mathbf{u}) \mathrm{N}(\mathbf{u} \,|\, \overline{\mathbf{m}}_\mathbf{u}, \overline{\mathbf{C}}_\mathbf{u}) \,\mathrm{d}\mathbf{u}$. Replacing the expectations with point estimates in a similar way to Equation (17) leads to a sparse version of the Laplace energy.

The sparse PEP energy is

$$\begin{aligned} \mathrm{PEPE}(q(\mathbf{u})) = -\frac{1}{\alpha}\sum_{n=1}^{N} \log \mathbb{E}_{q_n^{\setminus}(\mathbf{u})}[\mathbb{E}_{p(\mathbf{f}_n \,|\, \mathbf{u})}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)]] \\ + \frac{1}{\alpha}\log \mathbb{E}_{q^{\setminus}(\mathbf{u})}[\mathrm{N}^\alpha(\mathbf{u} \,|\, \overline{\mathbf{m}}_\mathbf{u}, \overline{\mathbf{C}}_\mathbf{u})] - \log \mathcal{Z}_\mathbf{u}, \end{aligned} \tag{66}$$

where $q^{\setminus}(\mathbf{u}) = q(\mathbf{u})/t^\alpha(\mathbf{u})$ is the 'global' cavity, and $q_n^{\setminus}(\mathbf{u}) = q(\mathbf{u})/t_n^\alpha(\mathbf{u})$ is the cavity associated with data point $\mathbf{y}_n$. This result is arrived at using a similar approach to Section 4.2.1: we set $z_\mathbf{u}$ such that the zero-th moment (*i.e.*, the log normaliser) of the approximate posterior matches the (product of) zero-th moments of the tilted distributions,

$$z_\mathbf{u}^\alpha \, \mathbb{E}_{q^{\setminus}(\mathbf{u})}[\mathrm{N}^\alpha(\mathbf{u} \,|\, \overline{\mathbf{m}}_\mathbf{u}, \overline{\mathbf{C}}_\mathbf{u})] = \prod_{n=1}^{N} \mathbb{E}_{q_n^{\setminus}(\mathbf{u})}[\mathbb{E}_{p(\mathbf{f}_n \,|\, \mathbf{u})}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)]], \tag{67}$$

and then rearrange in a similar fashion to Equation (26).

Again using the property in Equation (23), we can see that the PEP energy is equal to the VFE in the limit as $\alpha \to 0$. Since the sparse algorithm presented above is identical for both methods, we can conclude the following:

Remark 5 *The connection between PEP and natural gradient VI holds in the sparse GP case: sparse PEP ($\alpha \to 0$) is equivalent to natural gradient sparse VI.*

8.3 State Space Models and Markovian Gaussian Processes

All of the listed inference schemes can be used to perform inference in the discrete-time state space model with linear Gaussian dynamics of the following form,

$$\begin{aligned} \bar{\mathbf{f}}_{n+1} &= \mathbf{A}_n \bar{\mathbf{f}}_n + \mathbf{q}_n \,, \qquad \mathbf{q}_n \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_n), \\ \mathbf{y}_n \,|\, \bar{\mathbf{f}}_n &\sim p(\mathbf{y}_n \,|\, \bar{\mathbf{H}} \bar{\mathbf{f}}_n), \end{aligned} \tag{68}$$

where $\bar{\mathbf{f}}_n \in \mathbb{R}^{S \times 1}$ is the Gaussian distributed state vector, $\mathbf{A}_n \in \mathbb{R}^{S \times S}$ is the transition matrix, and $\mathbf{Q}_n \in \mathbb{R}^{S \times S}$ is the process noise. $\bar{\mathbf{H}} \in \mathbb{R}^{D \times S}$ is the measurement matrix such that $\mathbf{f}_n = \bar{\mathbf{H}} \bar{\mathbf{f}}_n$. In essence, the Gaussian prior in Equation (1) has been replaced by the linear Gaussian state space model on the first line of Equation (68). Therefore inference again simply involves replacing the true likelihood, $p(\mathbf{y}_n \,|\, \bar{\mathbf{H}} \bar{\mathbf{f}}_n)$, with the approximate likelihood, $t(\mathbf{f}_n)$, after which the approximate posterior over the state, $q(\bar{\mathbf{f}}) \approx p(\bar{\mathbf{f}} \,|\, \mathbf{y})$, is given by application of the linear Kalman filter followed by the Rauch–Tung–Striebel smoother (Särkkä, 2013). That is, the global posterior update, Equation (10), is replaced with linear filtering and smoothing. The full algorithm is given in Algorithm 3.

8.3.1 Markovian Gaussian Processes

Consider again the GP model in Section 8.1. If the inputs are vector valued and ordered, $\mathbf{X} \in \mathbb{R}^{N \times 1} = [x_1, \ldots, x_N]^\top$, for example if they represent sequential time steps, then for many common covariance functions the GP can be rewritten as a linear time-invariant stochastic differential equation (Hartikainen, 2013),

$$\mathrm{d}\bar{\mathbf{f}}(x) = \mathbf{F}\, \bar{\mathbf{f}}(x)\, \mathrm{d}x + \mathbf{L}\, \mathrm{d}\boldsymbol{\beta}(x), \tag{69}$$

where $\mathbf{F}$ is the feedback matrix and $\mathrm{d}\boldsymbol{\beta}(x)$ has spectral density $\mathbf{Q}_c$. $\mathbf{F}$ and $\mathbf{Q}_c$ are determined by the GP covariance function such that the model exhibits similar covariance properties to the standard GP (see Särkkä and Solin, 2019, for details). There is a linear relationship between the state, $\bar{\mathbf{f}}$, and the function, f, characterised by the measurement matrix, $f(x_n) = \bar{\mathbf{H}} \bar{\mathbf{f}}(x_n)$. The discrete-time solution to this SDE is given by Equation (68) where

$$\mathbf{A}_n = \boldsymbol{\Phi}(\mathbf{F} \Delta_n) \quad \text{and} \quad \mathbf{Q}_n = \int_0^{\Delta_n} \boldsymbol{\Phi}(\Delta_n - \tau)\, \mathbf{L}\, \mathbf{Q}_c\, \mathbf{L}^\top\, \boldsymbol{\Phi}(\Delta_n - \tau)^\top \mathrm{d}\tau, \tag{70}$$

for step size $\Delta_n = x_{n+1} - x_n$, where $\boldsymbol{\Phi}(\cdot)$ is the matrix exponential. The initial state is $\bar{\mathbf{f}}_0 \sim \mathrm{N}(\mathbf{0}, \mathbf{P}_0)$, where $\mathbf{P}_0$ is the solution to the Lyapunov equation, $\mathbf{F}\mathbf{P}_0 + \mathbf{P}_0\mathbf{F}^\top + \mathbf{L}\mathbf{Q}_c\mathbf{L}^\top = \mathbf{0}$. For many covariance functions, $\mathbf{A}_n$, $\mathbf{Q}_n$, and $\mathbf{P}_0$ can be computed in closed form. Once constructed, inference can again proceed by application of linear filtering and smoothing,

which will return the exact same posterior as that given by the approach in Section 8.1. This algorithm scales linearly in the number of data points, $\mathcal{O}(S^3N)$.

The use of the VI updates in conjunction with filtering and smoothing gives the approach of Chang et al. (2020), whilst the use of PEP was explored in Wilkinson et al. (2020). The Newton/Laplace approach was first applied to Markovian GPs in Nickisch et al. (2018). PL was initially derived as an approach for inference in state space models, and generalises the iterated nonlinear Kalman smoothers such as the extended and unscented smoothers. The Markovian approach can be extended to spatio-temporal data with more than one input dimension via the use of infinite-dimensional filtering methods (Särkkä et al., 2013; Hamelijnck et al., 2021; Tebbutt et al., 2021). The sparse and Markovian approaches can also be combined to further reduce the computational scaling (Wilkinson et al., 2021).

8.3.2 The Markovian GP Energy

The GP model energy can also be computed efficiently when using the Markovian approach. Consider the energy functions for VI, Laplace, and PEP, given by Equation (16), Equation (17) and Equation (27) respectively. In each case, the first two terms can be computed as usual, since they only require the marginals, $q(\mathbf{f}_n)$, which are given by filtering and smoothing. The final term, $\log \mathcal{Z} = \log \int p(\mathbf{f}) \prod_{n=1}^{N} \mathrm{N}(\mathbf{f}_n \,|\, \overline{\mathbf{m}}_n, \overline{\mathbf{C}}_{n,n}) \,\mathrm{d}\mathbf{f}$, can be computed sequentially during the forward filter as,

$$\log \mathcal{Z} = \sum_{n=1}^{N} \log \int p(\bar{\mathbf{f}}(x_n) \,|\, \overline{\mathbf{m}}_{1:n-1}) \,\mathrm{N}(\overline{\mathbf{m}}_n \,|\, \bar{\mathbf{H}}\bar{\mathbf{f}}(x_n), \overline{\mathbf{C}}_{n,n}) \,\mathrm{d}\bar{\mathbf{f}}(x_n), \tag{71}$$

where $p(\bar{\mathbf{f}}(x_n) \,|\, \overline{\mathbf{m}}_{1:n-1})$ is the predictive filtering distribution. Therefore the energy for all inference schemes can also be computed in $\mathcal{O}(S^3N)$.

8.4 Experiments

To evaluate our proposed methods, we consider three case studies. These include likelihood models that are a nonlinear function of multiple latent Gaussian processes, and which consistently result in non-PSD covariances when inference is applied naïvely. These models can be seen as instances of *chained GPs* (Saul et al., 2016), with the distinction that Saul et al. (2016) assume independence between the latent processes to make the model tractable and stable, which amounts to a similar approach to the heuristic method discussed in Section 4.5.1. We do not assume independence between the posterior processes, and we demonstrate that this can improve the inference result. Such an approach can become computationally prohibitive when the number of latent processes is large, although the sparse algorithm given in Section 8.2 is applicable in all cases, and the Markovian approach of Section 8.3 is applicable whenever the inputs are one-dimensional. We compute the negative log predictive density (NLPD) of the test data as the main performance metric. We set the EP power to $\alpha = 0.5$. In almost all cases, we find that the methods based on Riemannian gradients result in non-PSD covariances when the method nears its optima. Therefore in the reported results we take the last stable step before the algorithm fails as the final result.

We have also added a first-order variational inference method as a baseline to each experiment. This is achieved by training the approximate likelihood mean and covariance

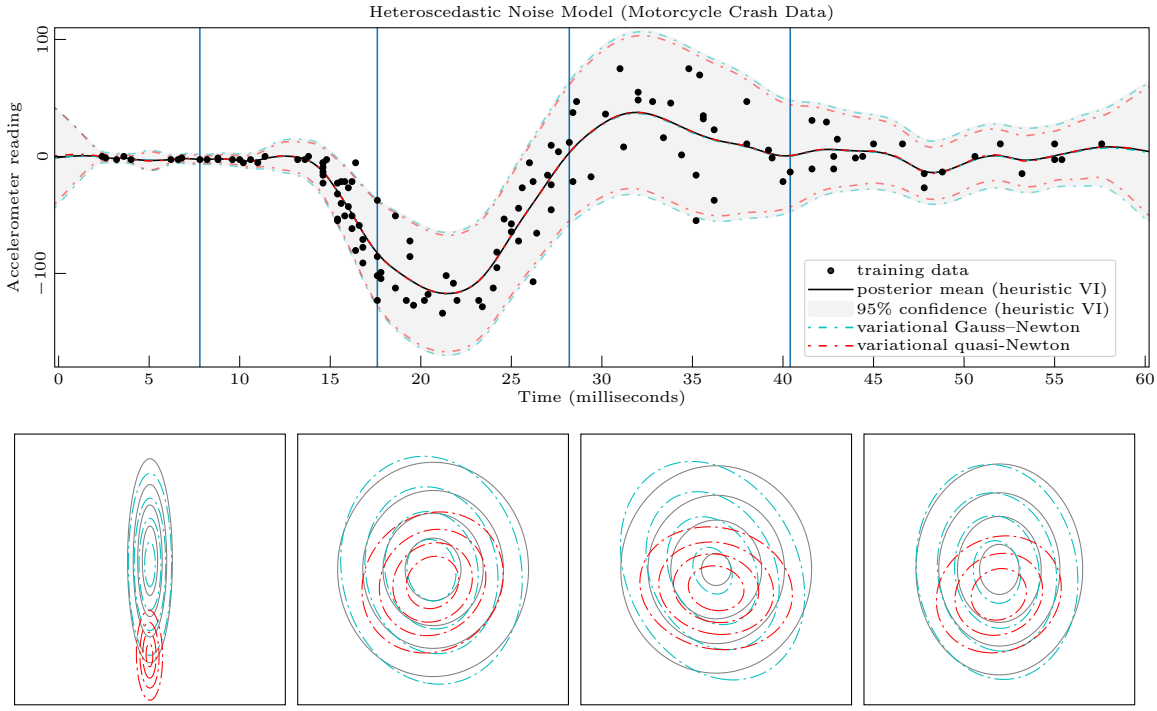

Figure 2: Example heteroscedastic noise model results. The top figure compares the posterior obtained for the motorcycle crash data set when using heuristic VI, variational Gauss–Newton and variational quasi-Newton. All methods obtain similar mean values, but different covariances. The contour plots show the marginal posterior covariance between f_1 (x-axis) and f_2 (y-axis, the noise standard deviation GP) at the time points marked by vertical lines in the top figure. The heuristic method (grey) assumes posterior independence between the two latents, variational Gauss–Newton (cyan) captures small amounts of cross-covariance, whilst variational quasi-Newton (red) captures more significant cross-covariance.

via gradient-base optimisation of the VFE using the Adam optimiser with a learning rate of 0.1 (we empirically found this to give the best performance and convergence). This is equivalent to the variational-GP method of Opper and Archambeau (2009). As expected, the first-order method converges more slowly than the other methods.

8.4.1 Gaussian Process Regression with Heteroscedastic Noise

First we consider the model of Goldberg et al. (1997), which augments a standard regression model with an additional GP prior on the observation noise standard deviation,

$$\begin{aligned} &f_1(\cdot) \sim \mathrm{GP}(0, \kappa_1(\cdot,\cdot)), \quad f_2(\cdot) \sim \mathrm{GP}(0, \kappa_2(\cdot,\cdot)), \\ &\mathbf{y}_n \,|\, f_1(\mathbf{X}_n), f_2(\mathbf{X}_n) \sim \mathrm{N}(\mathbf{y}_n \,|\, f_1(\mathbf{X}_n),\, \phi(f_2(\mathbf{X}_n))^2), \end{aligned} \tag{72}$$

where $\phi(\cdot) = \log(1 + \exp(\cdot))$ is the softplus function which ensures the standard deviation is positive. There are two latent processes, $D = 2$, and the observations are scalars, $D_{\mathbf{y}} = 1$. This model has been a focus of much research due to its relevance to real-world applications

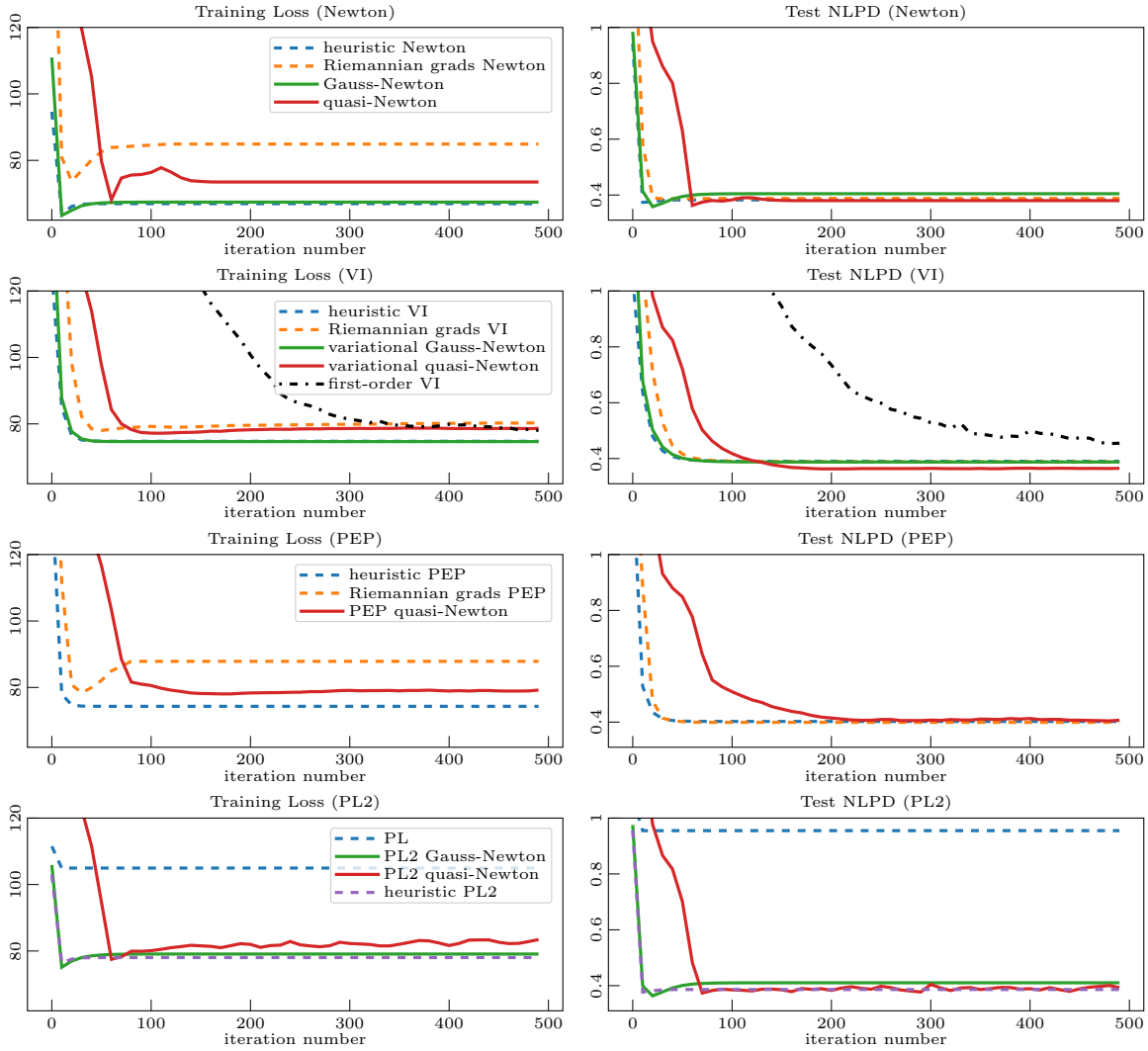

Figure 3: Heteroscedastic noise results. Mean of 4-fold cross validation shown. Variational quasi-Newton is capable of obtaining the best predictive performance, however all quasi-Newton methods converge slowly due to the need for damping. Gauss–Newton variants obtain similar performance to the heuristic method, but do not require computation of second-order derivatives. PL2 is the second-order PL method, and it significantly outperforms PL because it takes into account the gradients of the likelihood covariance.

(Tolvanen et al., 2014; Lázaro-Gredilla and Titsias, 2011). The model is applied to data simulating $N = 133$ accelerometer readings from a motorcycle crash (Silverman, 1985). κ_1, κ_2 are Matérn-$3/2$ kernels, and we use a learning rate of $\rho = 0.3$ and a quasi-Newton damping rate of $\xi = 0.5$. The data inputs and outputs are scaled to have zero mean and unit variance, and the kernel hyperparameters (lengthscales and variances) are all fixed at the value 1. We use Gauss–Hermite integration with $20^2 = 400$ points to solve the intractable integrals required for the VI-, EP- and PL-based methods. An example inference result is shown in Figure 2, and the test performance using 4-fold cross validation is shown in Figure 3.

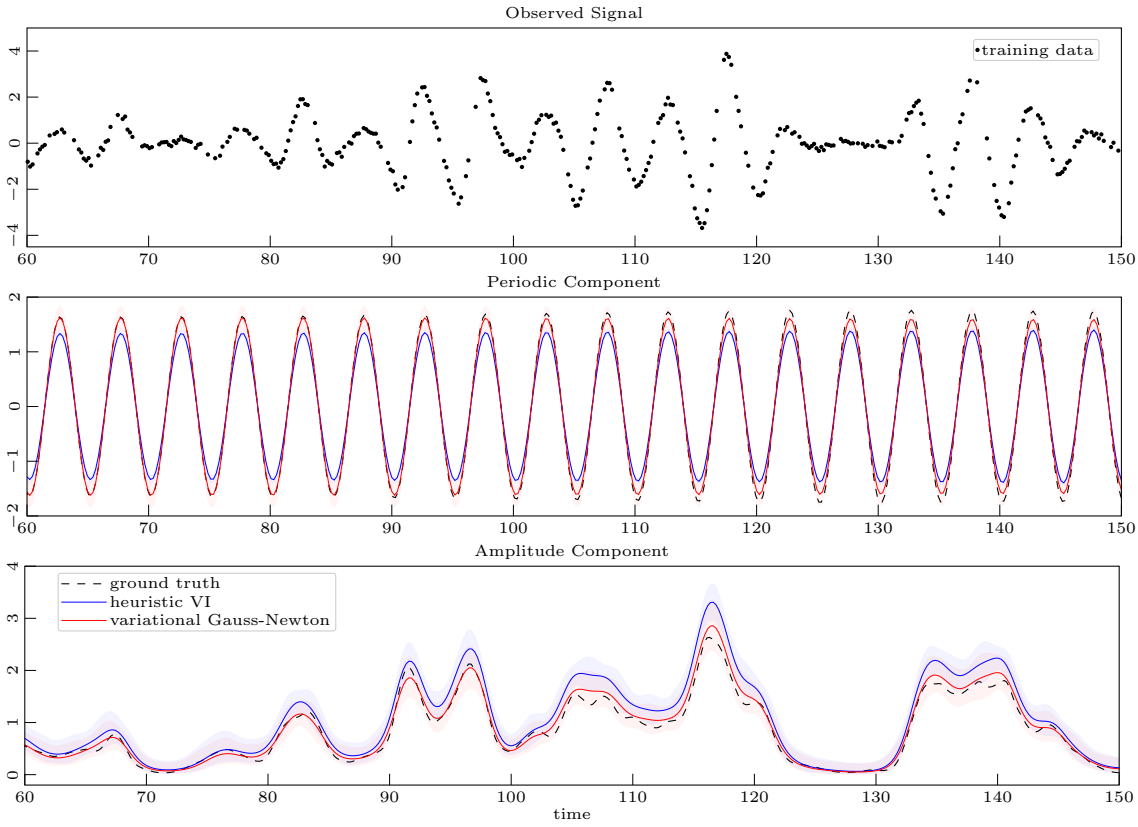

Figure 4: A short segment of the signal used for the amplitude demodulation experiment. The observed signal is produced via the product of a periodic component and a positive amplitude envelope. By taking into account cross-covariance between components, variational Gauss–Newton is better able to recover the ground truth than the heuristic approach.

Standard PL fails badly on this task since it does not take into account the gradients of the likelihood covariance (see Section 5.3 for discussion). From Figure 2 we can see that variational Gauss–Newton does not capture a significant amount of cross-covariance between the latent components and provides very similar performance to the heuristic method, but has the benefit of not requiring computation of the full Hessian. The variational quasi-Newton method provides the best test performance, but all quasi-Newton methods converge slowly due to the use of damped BFGS updates (see Section 6.5).

8.4.2 Bayesian Amplitude Demodulation: the Product Likelihood

Next we apply our methods to the model of Turner and Sahani (2011), which assumes an observed signal is produced by the product of a periodic component and a positive amplitude envelope. The task is to uncover these latent components from the signal alone. We use the following generative model,

$$\begin{aligned} &f_1(\cdot) \sim \mathrm{GP}(0, \kappa_1(\cdot,\cdot))\,, \quad f_2(\cdot) \sim \mathrm{GP}(0, \kappa_2(\cdot,\cdot))\,, \\ &\mathbf{y}_n \,|\, f_1(\mathbf{X}_n), f_2(\mathbf{X}_n) \sim \mathrm{N}(\mathbf{y}_n \,|\, f_1(\mathbf{X}_n)\phi(f_2(\mathbf{X}_n)), \sigma^2)\,, \end{aligned} \tag{73}$$

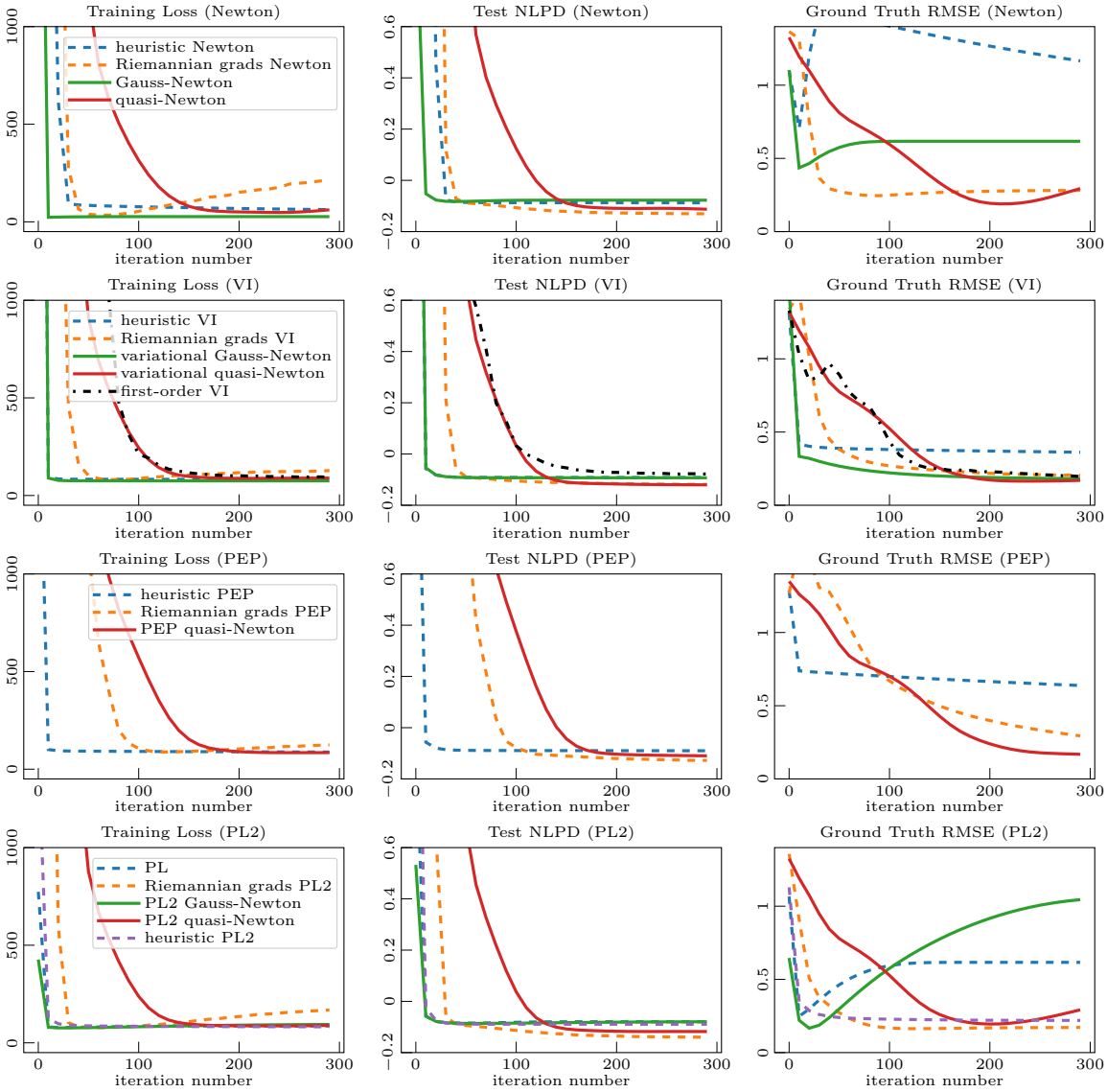

Figure 5: Bayesian amplitude demodulation results. Mean of 4-fold cross validation shown. Whilst the test NLPD is similar for most methods, those capable of capturing the cross-covariance between latent components obtain better RMSE relative to the ground truth. PL quasi-Newton results not plotted since it is identical to PL2 quasi-Newton for this model.

where $\phi(\cdot)$ is again the softplus function to ensure the amplitude is positive. Again we have $D = 2$, and $D_{\mathbf{y}} = 1$. κ_1 is an oscillator kernel made up of the product of the cosine kernel and the Matérn-3/2: $\kappa_1(x, x') = \cos(\omega(x - x'))\kappa_{\text{Mat-3/2}}(x, x')$ with $\omega = \frac{2}{5}\pi$, unit variance and a lengthscale of 500. We consider $N = 1000$ evenly spaced inputs in the range $[0, 200]$, therefore this amounts to an almost perfectly sinusoidal prior. κ_2 is a Matérn-5/2 kernel with a lengthscale of 3 and a variance of 2. The likelihood noise variance is $\sigma^2 = 0.1$.

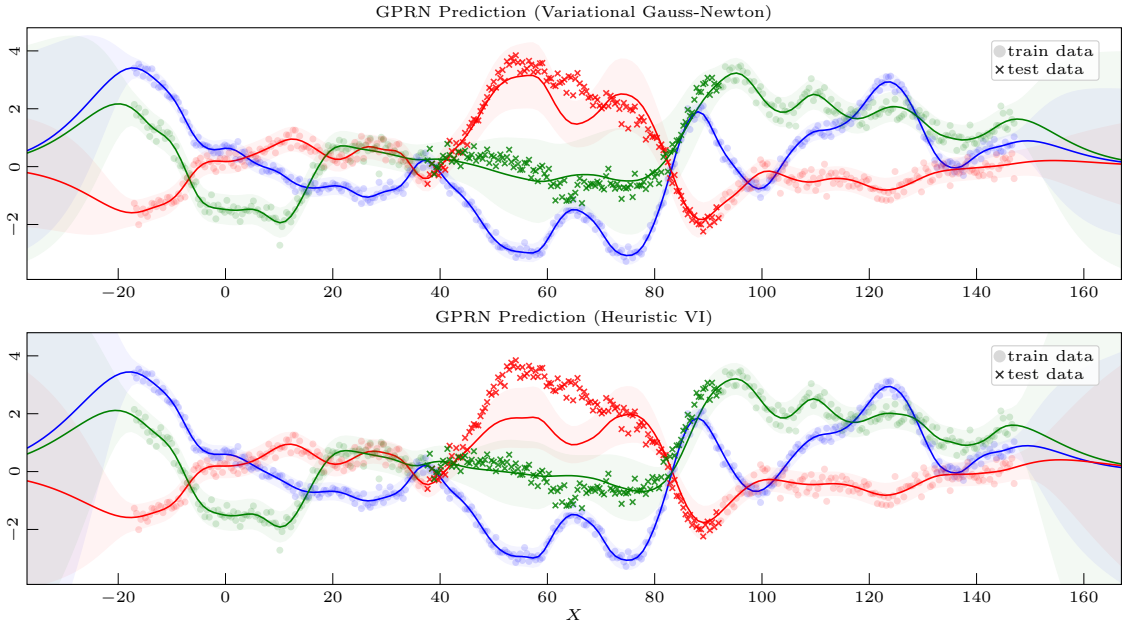

Figure 6: Example GPRN prediction results. The variational Gauss–Newton method is capable of learning meaningful dependencies between the output streams, and hence is able to interpolate accurately. The heuristic VI approach, which assumes independence between the posterior latent processes, makes less accurate predictions.

We use a learning rate of $\rho = 0.1$ and a quasi-Newton damping rate of $\xi = 0.5$. We again use Gauss–Hermite integration to solve the intractable integrals. We draw a sample from the model to be used as the training and test data, and the hyperparameters are fixed to their true values. The aim of Bayesian amplitude demodulation is to uncover the ground truth latent functions whilst characterising the model uncertainty. Figure 4 shows an example inference result, and we plot the 4-fold cross validation results in Figure 5 where we measure both the test NLPD and the RMSE of the posterior mean relative to the ground truth components. Whilst the test NLPD results show similar behaviour to the heteroscedastic noise task, we can see that the methods which take into account the cross-covariance between the latent components are able to better recover the ground truth oscillator and amplitude envelope.

8.4.3 The Gaussian Process Regression Network

The Gaussian process regression network (GPRN, Wilson et al., 2012) is a multi-output GP model capable of capturing complex time-varying dependencies between observation streams. We use the following generative model,

$$\begin{aligned} &f_i(\cdot) \sim \mathrm{GP}(0, \kappa_f(\cdot,\cdot)), \quad i = 1, 2, \\ &W_{j,i}(\cdot) \sim \mathrm{GP}(0, \kappa_W(\cdot,\cdot)), \quad j = 1, \ldots, 3, \ i = 1, 2, \\ &\mathbf{y}_n \,|\, f(\mathbf{X}_n), W(\mathbf{X}_n) \sim \mathrm{N}(\mathbf{y}_n \,|\, W(\mathbf{X}_n) f(\mathbf{X}_n), \, \Sigma), \end{aligned} \tag{74}$$

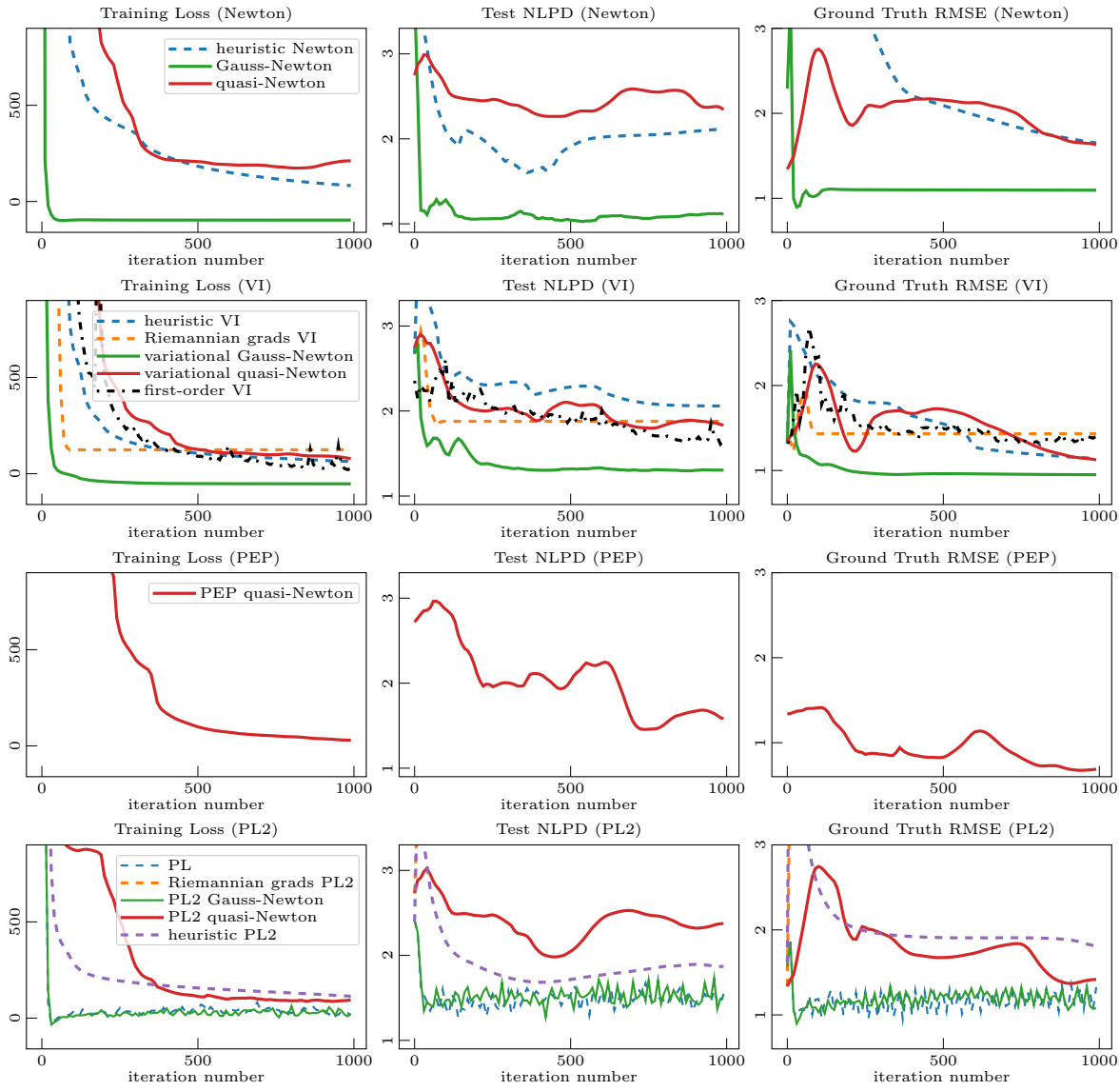

Figure 7: Gaussian process regression network (GPRN) results. Mean value across 4 synthetic data sets shown. Some methods not plotted due to divergent behaviour during training. The Gauss–Newton methods (including PL) significantly outperform all other approaches in terms of convergence rate, test NLPD and their ability to recover the ground truth.

where $\mathbf{y}_n \in \mathbb{R}^{3\times 1}$. Inference in Equation (74) is notoriously difficult due to the nonlinear interaction of many components (there are 8 latent Gaussian processes, $D = 8$, $D_{\mathbf{y}} = 3$). κ_f and κ_W are Matérn-$5/2$ kernels with unit variance. κ_f and κ_W have lengthscales of 10 and 70 respectively. We assume correlated observation noise,

$$\Sigma = \begin{pmatrix} 0.02 & -0.015 & -0.005 \\ -0.015 & 0.04 & 0.01 \\ -0.005 & 0.01 & 0.06 \end{pmatrix},$$

and draw four samples from the model to be used as separate data sets, with 400 time steps (each with 3 outputs, giving 1200 total data points) evenly spaced in the range $[-17, 147]$. When the output streams are partially observed, the GPRN can be a powerful model for interpolation of missing data, so we remove the middle third of data for two of the three output streams and then compute the NLPD of the removed data as well as the RMSE of the posterior mean relative to the ground truth. We use a learning rate of $\rho = 0.3$ and a quasi-Newton damping rate of $\xi = 0.3$. The hyperparameters are fixed to their true values. Since the intractable integrals required for the VI-, EP- and PL-based methods are now 8-dimensional, we use the 5^{th}-order unscented transform (McNamee and Stenger, 1967) to approximate them instead of Gauss–Hermite.

Figure 6 plots an example prediction result on one of the synthetic data sets, and Figure 7 plots the mean prediction performance across the four data sets. These results show that the independence assumption between latent processes made in the original GPRN paper is highly detrimental to performance and convergence rate. By including the cross-covariance terms, the model improves in terms of prediction quality. However, the original approach proposed by Wilson et al. (2012) is much more efficient than our methods when the number of latent functions is large.

The Gauss–Newton methods significantly outperform all others with regards to convergence rate, test NLPD and ground truth RMSE. PL and PL2 Gauss–Newton both perform well, but exhibit oscillatory behaviour suggesting they may require a smaller learning rate than the Gauss–Newton and variational Gauss–Newton approaches. The quasi-Newton methods perform poorly on this task, since the rank-two BFGS updates are less accurate than on the previous tasks due to the higher number of latent functions.

9. Conclusions and Discussion

Through careful analysis of variational inference, expectation propagation and posterior linearisation, we have shown that many approaches to approximate Bayesian inference can be viewed as update rules to local approximate likelihood terms. Furthermore these update rules involve computing the Jacobian and (approximate) Hessian of a surrogate target, and can be cast under the framework of numerical optimisation.

Our work aims to draw connections between such ideas from optimisation, machine learning and signal processing, and in particular the development of the variational Gauss–Newton method provides a key link between VI, the Gauss–Newton method, and linearisation-based methods (which also turn out to make Gauss–Newton approximations). Additionally, it serves as an approximation to VI that guarantees PSD updates and which exhibits excellent performance in some very difficult inference tasks such as the GPRN multi-output model.

The quasi-Newton methods showed less promise experimentally, especially in terms of convergence rates, but for some tasks they are able to capture shared information between latent components that other methods are not. This may motivate further work in this area, since it is clear that the heuristic and Gauss–Newton methods are not optimal in terms of inference quality. Our PEP quasi-Newton approach also contributes to the search for accurate and stable EP algorithms.

The explicit connections to the optimisation literature presented here motivate avenues for further research. Perhaps the most interesting of these would be analysis of the convergence

properties of the Bayes–Newton methods, and development of line-search methods for improving the stability and convergence of all schemes. Whilst convergence analysis was out of scope for this paper, we hope that the connections to Newton's method and its variants is a useful perspective in this regard. Python code for the methods and experiments is provided at `https://github.com/AaltoML/BayesNewton` (see Appendix I for instructions on how to reproduce the results).

Acknowledgments

We acknowledge funding from the Academy of Finland (project numbers 324345 and 339730) and the computational resources provided by the Aalto Science-IT project. We thank the anonymous reviewers and Paul Chang for comments on the manuscript.

References

Vincent Adam, Paul E. Chang, Mohammad Emtiyaz Khan, and Arno Solin. Dual parameterization of sparse variational Gaussian processes. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 11474–11486. Curran Associates, Inc., 2021.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2): 251–276, 1998.

Bradley M Bell. The iterated Kalman smoother as a Gauss–Newton method. *SIAM Journal on Optimization*, 4(3):626–636, 1994.

Åke Björck. *Numerical Methods for Least Squares Problems*. SIAM, 1996.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Charles G Broyden. Quasi-Newton methods and their application to function minimisation. *Mathematics of Computation*, 21(99):368–381, 1967.

Charles G Broyden. A new double-rank minimisation algorithm. *Notices of the American Mathematical Society*, 16(4):670, 1969.

Thang D Bui, Josiah Yan, and Richard E Turner. A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *Journal of Machine Learning Research (JMLR)*, 18(1):3649–3720, 2017.

Thang D Bui, Cuong V Nguyen, Siddharth Swaroop, and Richard E Turner. Partitioned variational inference: A unified framework encompassing federated and continual learning. *arXiv preprint arXiv:1811.11206*, 2018.

Edward Challis and David Barber. Gaussian Kullback-Leibler approximate inference. *Journal of Machine Learning Research*, 14(8):2239–2286, 2013.

Paul E. Chang, William J. Wilkinson, Mohammed Emtiyaz Khan, and Arno Solin. Fast variational learning in state-space Gaussian process models. In *International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2020.

Lehel Csató and Manfred Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

Guillaume Dehaene and Simon Barthelmé. Expectation propagation in the large data limit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(1):199–217, 2018.

Roger Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13 (3):317–322, 1970.

Ángel F García-Fernández, Lennart Svensson, and Simo Särkkä. Iterated posterior linearization smoother. *IEEE Transactions on Automatic Control*, 62(4):2056–2063, 2016.

Ángel F García-Fernández, Filip Tronarp, and Simo Särkkä. Gaussian process classification using posterior linearization. *IEEE Signal Processing Letters*, 26(5):735–739, 2019.

Arthur Gelb. *Applied Optimal Estimation.* MIT Press, 1974.

Paul W Goldberg, Christopher KI Williams, and Christopher M Bishop. Regression with input-dependent noise: A Gaussian process treatment. *Advances in Neural Information Processing Systems 10 (NIPS)*, pages 493–499, 1997.

Donald Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of Computation*, 24(109):23–26, 1970.

Gene H Golub and Victor Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on Numerical Analysis*, 10(2):413–432, 1973.

Oliver Hamelijnck, William J. Wilkinson, Niki Loppi, Arno Solin, and Theodoros Damoulas. Spatio-temporal variational Gaussian processes. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 23621–23633. Curran Associates, Inc., 2021.

Jouni Hartikainen. *Sequential Inference for Latent Temporal Gaussian Process Models.* Doctoral dissertation, Aalto University, Finland, 2013.

Philipp Hennig and Martin Kiefel. Quasi-Newton methods: A new direction. *The Journal of Machine Learning Research*, 14(1):843–865, 2013.

Philipp Hennig, Michael A Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2179):20150142, 2015.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 38 of *Proceedings of Machine Learning Research*, pages 351–360. PMLR, 2015.

Pasi Jylänki, Jarno Vanhatalo, and Aki Vehtari. Robust Gaussian process regression with a Student-t likelihood. *Journal of Machine Learning Research*, 12(99):3227–3257, 2011.

Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 878–887. PMLR, 2017.

Mohammad Emtiyaz Khan and Håvard Rue. The Bayesian Learning Rule. *arXiv preprint arXiv:2107.04562*, 2021.

Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 2611–2620. PMLR, 2018.

Mohammad Emtiyaz Khan, Alexander Immer, Ehsan Abedi, and Maciej Korzepa. Approximate inference turns deep networks into Gaussian processes. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 3094–3104. Curran Associates, Inc., 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Miguel Lázaro-Gredilla and Michalis K Titsias. Variational heteroscedastic Gaussian process regression. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. Omnipress, 2011.

William E Leithead and Yunong Zhang. O(N^2)-operation approximation of covariance matrix inverse in Gaussian process regression based on quasi-Newton BFGS method. *Communications in Statistics—Simulation and Computation*, 36(2):367–380, 2007.

Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 2323–2331. Curran Associates, Inc., 2015.

Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Stein's lemma for the reparameterization trick with exponential family mixtures. In *ICML Workshop on Stein's Method in Machine Learning and Statistics*, 2019.

Wu Lin, Mark Schmidt, and Mohammad Emtiyaz Khan. Handling the positive-definite constraint in the Bayesian Learning Rule. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020.

John McNamee and Frank Stenger. Construction of fully symmetric numerical integration formulas. *Numerische Mathematik*, 10(4):327–344, 1967.

Thomas P. Minka. Power EP. Technical report, Microsoft Research, 2004. MSR-TR-2005-173.

Thomas Peter Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

Tom Minka. Divergence measures and message passing. Technical report, Microsoft Research, 2005.

Hannes Nickisch and Carl Edward Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

Hannes Nickisch, Arno Solin, and Alexander Grigorievskiy. State space Gaussian processes with non-Gaussian likelihood. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 3789–3798. PMLR, 2018.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering, 2006.

Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Manfred Opper and Ole Winther. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6(Dec):2177–2204, 2005.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, USA, 2006.

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in Gaussian process models. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84 of *Proceedings of Machine Learning Research*, pages 689–697. PMLR, 2018.

Simo Särkkä. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Cambridge University Press, 2019.

Simo Särkkä, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing. *IEEE Signal Processing Magazine*, 30(4): 51–61, 2013.

Masa-Aki Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681, 2001.

Alan D. Saul, James Hensman, Aki Vehtari, and Neil D. Lawrence. Chained Gaussian processes. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 51 of *Proceedings of Machine Learning Research*, pages 1431–1440. PMLR, 2016.

Matthias Seeger. Expectation propagation for exponential families. Technical report, University of California at Berkeley, 2005.

Matthias Seeger and Hannes Nickisch. Fast convergent algorithms for expectation propagation approximate Bayesian inference. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 15 of *Proceedings of Machine Learning Research*, pages 652–660. PMLR, 2011.

David F Shanno. Conditioning of quasi-Newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.

Bernhard W Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society: Series B (Methodological)*, 47(1):1–21, 1985.

Will Tebbutt, Arno Solin, and Richard E. Turner. Combining pseudo-point and state space approximations for sum-separable Gaussian processes. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence (UAI)*, Proceedings of Machine Learning Research. PMLR, 2021.

Luke Tierney and Joseph B Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

Ville Tolvanen, Pasi Jylänki, and Aki Vehtari. Expectation propagation for nonstationary heteroscedastic Gaussian process regression. In *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2014.

Minh-Ngoc Tran, Dang H Nguyen, and Duy Nguyen. Variational Bayes on manifolds. *arXiv preprint arXiv:1908.03097*, 2019.

Richard E Turner and Maneesh Sahani. Demodulation as probabilistic inference. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2398–2411, 2011.

William Wilkinson, Arno Solin, and Vincent Adam. Sparse algorithms for Markovian Gaussian processes. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130 of *Proceedings of Machine Learning Research*, pages 1747–1755. PMLR, 2021.

William J. Wilkinson, Paul E. Chang, Michael Riis Andersen, and Arno Solin. State space expectation propagation: Efficient inference schemes for temporal Gaussian processes. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 10270–10281. PMLR, 2020.

Andrew G. Wilson, David A. Knowles, and Zoubin Ghahramani. Gaussian processes regression networks. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*. Omnipress, 2012.

Appendix A. Derivation of the Online Newton Updates

Newton's method corresponds to approximating

$$\mathcal{L}(\mathbf{f}) \approx \mathcal{L}(\mathbf{m}_k) + \nabla\mathcal{L}(\mathbf{m}_k)^\top (\mathbf{f} - \mathbf{m}_k) + \frac{1}{2}(\mathbf{f} - \mathbf{m}_k)^\top \nabla^2\mathcal{L}(\mathbf{m}_k)\,(\mathbf{f} - \mathbf{m}_k), \tag{75}$$

whose minimum is given by setting the derivative of the right hand side to zero:

$$\nabla\mathcal{L}(\mathbf{m}_k) + \nabla^2\mathcal{L}(\mathbf{m}_k)\,(\mathbf{f} - \mathbf{m}_k) = 0, \tag{76}$$

which then gives the next iterate as

$$\mathbf{m}_{k+1} = \mathbf{m}_k - (\nabla^2\mathcal{L}(\mathbf{m}_k))^{-1}\,\nabla\mathcal{L}(\mathbf{m}_k). \tag{77}$$

We can now set $\mathbf{C}_{k+1} = -(\nabla^2\mathcal{L}(\mathbf{m}_k))^{-1}$ which allows us to write

$$\begin{aligned} \mathbf{C}_{k+1}^{-1} &= -(\nabla^2\mathcal{L}(\mathbf{m}_k))^{-1}, \\ \mathbf{m}_{k+1} &= \mathbf{m}_k + \mathbf{C}_{k+1}\,\nabla\mathcal{L}(\mathbf{m}_k). \end{aligned} \tag{78}$$

However, instead of taking full step of Newton's we can also take a partial step (as is often done via line-search in the optimization literature) which replaces the update by

$$\begin{aligned} \mathbf{C}_{k+1}^{-1} &= (1-\rho)\,\mathbf{C}_k - (\nabla^2\mathcal{L}(\mathbf{m}_k))^{-1}, \\ \mathbf{m}_{k+1} &= \mathbf{m}_k + \rho\,\mathbf{C}_{k+1}\,\nabla\mathcal{L}(\mathbf{m}_k), \end{aligned} \tag{79}$$

as given in Equation (6). The iterates $\mathbf{C}_k$ indeed converge to $\mathbf{C} = -(\nabla^2\mathcal{L}(\mathbf{m}^*))^{-1}$ which is the Laplace approximation to the posterior covariance.

The following is a more detailed derivation of the result given in Equation (7),

$$\begin{aligned} \boldsymbol{\lambda}_{k+1}^{(2)} := -\frac{1}{2}\mathbf{C}_{k+1}^{-1} &= -(1-\rho)\frac{1}{2}\mathbf{C}_k^{-1} - \rho\,\frac{1}{2}\left(\mathbf{K}^{-1} - \nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\right) \\ &= -(1-\rho)\frac{1}{2}(\mathbf{K}^{-1} + \overline{\mathbf{C}}_k^{-1}) - \rho\,\frac{1}{2}\left(\mathbf{K}^{-1} - \nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\right) \\ &= \boldsymbol{\lambda}_{\text{prior}}^{(2)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(2)} + \rho\,\frac{1}{2}\nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\,, \\ \boldsymbol{\lambda}_{k+1}^{(1)} := \mathbf{C}_{k+1}^{-1}\mathbf{m}_{k+1} &= \mathbf{C}_{k+1}^{-1}\mathbf{m}_k + \rho\,\nabla_{\mathbf{f}} \log p(\mathbf{y}\,|\,\mathbf{m}_k) - \rho\,\mathbf{K}^{-1}(\mathbf{m}_k - \boldsymbol{\mu}) \\ &= (1-\rho)\,\mathbf{C}_k^{-1}\mathbf{m}_k + \rho\,\mathbf{K}^{-1}\boldsymbol{\mu} + \rho\,\left(\nabla_{\mathbf{f}} \log p(\mathbf{y}\,|\,\mathbf{m}_k) - \nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\,\mathbf{m}_k\right) \\ &= \boldsymbol{\lambda}_{\text{prior}}^{(2)}\mathbf{m}_k - \rho\,\mathbf{K}^{-1}\mathbf{m}_k + \rho\,\mathbf{K}^{-1}\boldsymbol{\mu} + \rho\,\left(\nabla_{\mathbf{f}} \log p(\mathbf{y}\,|\,\mathbf{m}_k) - \nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\,\mathbf{m}_k\right) \\ &= \boldsymbol{\lambda}_{\text{prior}}^{(1)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(1)} + \rho\left(\nabla_{\mathbf{f}} \log p(\mathbf{y}\,|\,\mathbf{m}_k) - \nabla_{\mathbf{f}}^2 \log p(\mathbf{y}\,|\,\mathbf{m}_k)\,\mathbf{m}_k\right). \end{aligned} \tag{80}$$

Appendix B. Derivation of the VI Updates

The following is a more detailed derivation of the result given in Equation (14),

$$\begin{aligned} \boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k - \rho\,\nabla_{\boldsymbol{\omega}}\text{VFE}(q(\mathbf{f})) \\ &= \boldsymbol{\lambda}_k - \rho\,(-\nabla_{\boldsymbol{\omega}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})] + \nabla_{\boldsymbol{\omega}}\mathbb{E}_{q(\mathbf{f})}[\log q(\mathbf{f})]) \\ &= \boldsymbol{\lambda}_k - \rho\,(-\nabla_{\boldsymbol{\omega}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})] + \boldsymbol{\lambda}_k) \\ &= (1-\rho)\boldsymbol{\lambda}_k + \rho\,\nabla_{\boldsymbol{\omega}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})]. \end{aligned} \tag{81}$$

By application of the chain rule (see Khan and Rue, 2021, for a detailed explanation of this step), the individual posterior parameter updates then become

$$\begin{aligned}
\boldsymbol{\lambda}_{k+1}^{(2)} &= (1-\rho)\boldsymbol{\lambda}_k^{(2)} + \rho\,\frac{1}{2}\nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})] \\
&= (1-\rho)\boldsymbol{\lambda}_k^{(2)} + \rho\left(\frac{1}{2}\nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})] + \boldsymbol{\lambda}_{\text{prior}}^{(2)}\right) \\
&= \boldsymbol{\lambda}_{\text{prior}}^{(2)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(2)} + \rho\,\frac{1}{2}\nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})], \\
\boldsymbol{\lambda}_{k+1}^{(1)} &= (1-\rho)\boldsymbol{\lambda}_k^{(1)} + \rho\left(\nabla_{\mathbf{m}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})] - \nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y},\mathbf{f})]\,\mathbf{m}_k\right) \\
&= (1-\rho)\boldsymbol{\lambda}_k^{(1)} + \rho\left(\nabla_{\mathbf{m}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})] - \mathbf{K}^{-1}(\mathbf{m}_k-\boldsymbol{\mu}) - (\nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})] - \mathbf{K}^{-1})\,\mathbf{m}_k\right) \\
&= \boldsymbol{\lambda}_{\text{prior}}^{(1)} + (1-\rho)\bar{\boldsymbol{\lambda}}_k^{(1)} + \rho\left(\nabla_{\mathbf{m}}\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})] - \nabla_{\mathbf{m}}^2\mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}\,|\,\mathbf{f})]\,\mathbf{m}_k\right).
\end{aligned} \tag{82}$$

Appendix C. Derivation of the PEP Updates

It is tempting to view PEP as minimising an alternative free energy approximation, and to apply natural gradient descent to the approximate energy as in the VI case. However, this approach would be flawed since this energy is not a bound for the true energy, and stationary points obtained via moment matching may not even be local minima (Opper and Winther, 2005).

The PEP algorithm proceeds by minimising local KL divergences,

$$t(\mathbf{f}_n) \leftarrow \arg\min_{t_*(\mathbf{f}_n)} \mathrm{D}_{\mathrm{KL}}\left[\frac{1}{Z_n}\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n) \;\|\; \frac{1}{W_n}\frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\right], \tag{83}$$

where $Z_n = \int \frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n$ and $W_n = \int \frac{t_*^\alpha(\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n$. Since the right hand side is Gaussian, this minimisation amounts to moment matching. Hence we must compute the first two moments of $\frac{1}{Z_n}\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)$. To derive the moment matching equations we take the derivatives of Z_n w.r.t. the cavity mean (we let $\mathrm{N}(\mathbf{f}_n\,|\,\mathbf{m}_n^{\setminus}, \mathbf{C}_{n,n}^{\setminus})$ be the marginal cavity mean),

$$\begin{aligned}
\frac{\partial Z_n}{\partial \mathbf{m}_n^{\setminus}} &= \mathbf{C}_{n,n}^{\setminus -1}\int(\mathbf{f}_n - \mathbf{m}_n^{\setminus})\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n \\
&= \mathbf{C}_{n,n}^{\setminus -1}\int \mathbf{f}_n\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n - \mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus}\int\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n \\
&= \mathbf{C}_{n,n}^{\setminus -1}Z_n\int \mathbf{f}_n Z_n^{-1}\frac{p^\alpha(\mathbf{y}_n\,|\,\mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)}q(\mathbf{f}_n)\mathrm{d}\mathbf{f}_n - \mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus}Z_n \\
&= \mathbf{C}_{n,n}^{\setminus -1}Z_n\mathbb{E}_{\tilde{q}}[\mathbf{f}_n] - \mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus}Z_n,
\end{aligned} \tag{84}$$

and rearranging the terms gives

$$\begin{aligned}
\mathbb{E}_{\tilde{q}}[\mathbf{f}_n] &= \mathbf{m}_n^{\setminus} + \mathbf{C}_{n,n}^{\setminus}\frac{\partial Z_n}{\partial \mathbf{m}_n^{\setminus}}Z_n^{-1} \\
&= \mathbf{m}_n^{\setminus} + \mathbf{C}_{n,n}^{\setminus}\frac{\partial \log Z_n}{\partial \mathbf{m}_n^{\setminus}}.
\end{aligned} \tag{85}$$

Differentiating again we get,

$$
\begin{aligned}
\frac{\partial^2 Z_n}{\partial \mathbf{m}_n^\backslash \partial \mathbf{m}_n^{\backslash\top}} &= \mathbf{C}_{n,n}^{\backslash -1} \int (\mathbf{f}_n - \mathbf{m}_n^\backslash)(\mathbf{f}_n - \mathbf{m}_n^\backslash)^\top \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} \\
&\quad - \mathbf{C}_{n,n}^{\backslash -1} \int \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \\
&= \mathbf{C}_{n,n}^{\backslash -1} \int \mathbf{f}_n \mathbf{f}_n^\top \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} - 2\mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash \int \mathbf{f}_n \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} \\
&\quad + \mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} \int \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} - \mathbf{C}_{n,n}^{\backslash -1} \int \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \\
&= \mathbf{C}_{n,n}^{\backslash -1} Z_n \int \mathbf{f}_n \mathbf{f}_n^\top Z_n^{-1} \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} \\
&\quad - 2\mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash Z_n \int \mathbf{f}_n Z_n^{-1} \frac{p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)}{t^\alpha(\mathbf{f}_n)} q(\mathbf{f}_n) \mathrm{d}\mathbf{f}_n \mathbf{C}_{n,n}^{\backslash -1} \\
&\quad + \mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} Z_n \mathbf{C}_{n,n}^{\backslash -1} - \mathbf{C}_{n,n}^{\backslash -1} Z_n \\
&= \mathbf{C}_{n,n}^{\backslash -1} Z_n \mathbb{E}_{\tilde{q}}[\mathbf{f}_n \mathbf{f}_n^\top] \mathbf{C}_{n,n}^{\backslash -1} - 2\mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash Z_n \mathbb{E}_{\tilde{q}}[\mathbf{f}_n] \mathbf{C}_{n,n}^{\backslash -1} \\
&\quad + \mathbf{C}_{n,n}^{\backslash -1} \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} Z_n \mathbf{C}_{n,n}^{\backslash -1} - \mathbf{C}_{n,n}^{\backslash -1} Z_n
\end{aligned} \tag{86}
$$

which gives

$$
\mathbb{E}_{\tilde{q}}[\mathbf{f}_n \mathbf{f}_n^\top] = 2\mathbf{m}_n^\backslash \mathbb{E}_{\tilde{q}}[\mathbf{f}_n] - \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} + \mathbf{C}_{n,n}^\backslash + \mathbf{C}_{n,n}^\backslash \frac{\partial^2 Z_n}{\partial \mathbf{m}_n^\backslash \partial \mathbf{m}_n^{\backslash\top}} Z_n^{-1} \mathbf{C}_{n,n}^\backslash, \tag{87}
$$

$$
\begin{aligned}
\mathrm{Cov}_{\tilde{q}}[\mathbf{f}_n] &= \mathbb{E}_{\tilde{q}}[\mathbf{f}_n \mathbf{f}_n^\top] - \mathbb{E}_{\tilde{q}}[\mathbf{f}_n] \mathbb{E}_{\tilde{q}}[\mathbf{f}_n]^\top \\
&= 2\mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} + 2\mathbf{m}_n^\backslash \mathbf{C}_{n,n}^\backslash \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash} Z_n^{-1} - \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} + \mathbf{C}_{n,n}^\backslash + \mathbf{C}_{n,n}^\backslash \frac{\partial^2 Z_n}{\partial \mathbf{m}_n^\backslash \partial \mathbf{m}_n^{\backslash\top}} Z_n^{-1} \mathbf{C}_{n,n}^\backslash \\
&\quad - \mathbf{m}_n^\backslash \mathbf{m}_n^{\backslash\top} - 2\mathbf{m}_n^\backslash \mathbf{C}_{n,n}^\backslash \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash} Z_n^{-1} - \mathbf{C}_{n,n}^\backslash \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash} \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash}^\top Z_n^{-2} \mathbf{C}_{n,n}^\backslash \\
&= \mathbf{C}_{n,n}^\backslash + \mathbf{C}_{n,n}^\backslash \left(\frac{\partial^2 Z_n}{\partial \mathbf{m}_n^\backslash \partial \mathbf{m}_n^{\backslash\top}} Z_n^{-1} - \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash} \frac{\partial Z_n}{\partial \mathbf{m}_n^\backslash}^\top Z_n^{-2} \right) \mathbf{C}_{n,n}^\backslash \\
&= \mathbf{C}_{n,n}^\backslash + \mathbf{C}_{n,n}^\backslash \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^\backslash \partial \mathbf{m}_n^{\backslash\top}} \mathbf{C}_{n,n}^\backslash.
\end{aligned} \tag{88}
$$

Now removing the cavity contribution from these posterior moments and scaling by the inverse power (since substituting the cavity results in a fraction α of the full likelihood)

gives the new approximate likelihood parameter updates,

$$\begin{aligned}
\overline{\mathbf{C}}_{n,n}^{-1} &= \frac{1}{\alpha}\left(\mathbf{C}_{n,n}^{\setminus} + \mathbf{C}_{n,n}^{\setminus}\frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1} - \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1} \\
&= \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\mathbf{C}_{n,n}^{\setminus -1} - \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1} \\
&= \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\mathbf{C}_{n,n}^{\setminus -1} - \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)\mathbf{C}_{n,n}^{\setminus -1} \\
&= \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\left(-\frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\right) \\
&= \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1}\left(\mathbf{C}_{n,n}^{\setminus -1} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\right)^{-1}\left(-\frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\right), \qquad (89)
\end{aligned}$$

where the re-arrangement on the last line is to ensure we only invert a symmetric matrix (note that whilst the quantity being inverted is guaranteed to be symmetric, it is not guaranteed to be PSD).

$$\begin{aligned}
\overline{\boldsymbol{\lambda}}_n^{(1)} &= \frac{1}{\alpha}\left(\mathbf{C}_{n,n}^{\setminus} + \mathbf{C}_{n,n}^{\setminus}\frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\left(\mathbf{m}_n^{\setminus} + \mathbf{C}_{n,n}^{\setminus}\frac{\partial \log Z_n}{\partial \mathbf{m}_n^{\setminus}}\right) - \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus} \\
&= \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus}\right)^{-1}\left(\mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus} + \frac{\partial \log Z_n}{\partial \mathbf{m}_n^{\setminus}}\right) - \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1}\mathbf{m}_n^{\setminus} \\
&= \frac{1}{\alpha}\left(\mathbf{I} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{C}_{n,n}^{\setminus -1}\right)^{-1}\left(\frac{\partial \log Z_n}{\partial \mathbf{m}_n^{\setminus}} - \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{m}_n^{\setminus}\right) \\
&= \frac{1}{\alpha}\mathbf{C}_{n,n}^{\setminus -1}\left(\mathbf{C}_{n,n}^{\setminus -1} + \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\right)^{-1}\left(\frac{\partial \log Z_n}{\partial \mathbf{m}_n^{\setminus}} - \frac{\partial^2 \log Z_n}{\partial \mathbf{m}_n^{\setminus} \partial \mathbf{m}_n^{\setminus\top}}\mathbf{m}_n^{\setminus}\right). \qquad (90)
\end{aligned}$$

Appendix D. Equivalence of PEP $\alpha \to 0$ and Natural Gradient VI

Following Bui et al. (2017), we utilise the Maclaurin series, $\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$, to write

$$\begin{aligned}
p^{\alpha}(\mathbf{y}_n \,|\, \mathbf{f}_n) &= \exp(\alpha \log p(\mathbf{y}_n \,|\, \mathbf{f}_n)) \\
&= 1 + \alpha \log p(\mathbf{y}_n \,|\, \mathbf{f}_n) + \underbrace{\frac{1}{2!}(\alpha \log p(\mathbf{y}_n \,|\, \mathbf{f}_n))^2 + \frac{1}{3!}(\alpha \log p(\mathbf{y}_n \,|\, \mathbf{f}_n))^3 + \ldots}_{\alpha^2 \psi(\cdot)}, \qquad (91)
\end{aligned}$$

which leads to (using the series $\log(1+x) = x - \frac{x^2}{2!} + \frac{x^3}{3!} - \dots$),

$$\begin{aligned}\frac{1}{\alpha}\log \mathbb{E}_{q^{\backslash}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] &= \frac{1}{\alpha}\log \int q^{\backslash}(\mathbf{f}_n)[1 + \alpha \log p(\mathbf{y}_n \,|\, \mathbf{f}_n) + \alpha^2 \psi(\cdot)]\mathrm{d}\mathbf{f}_n \\ &= \frac{1}{\alpha}\log \left[1 + \alpha \int q^{\backslash}(\mathbf{f}_n) \log p(\mathbf{y}_n \,|\, \mathbf{f}_n)\mathrm{d}\mathbf{f}_n + \alpha^2 \psi(\cdot)\right] \\ &= \int q^{\backslash}(\mathbf{f}_n) \log p(\mathbf{y}_n \,|\, \mathbf{f}_n)\mathrm{d}\mathbf{f}_n + \alpha\psi(\cdot), \end{aligned} \tag{92}$$

therefore,

$$\lim_{\alpha \to 0} \frac{1}{\alpha}\log \mathbb{E}_{q^{\backslash}(\mathbf{f}_n)}[p^\alpha(\mathbf{y}_n \,|\, \mathbf{f}_n)] = \mathbb{E}_{q(\mathbf{f}_n)}[\log p(\mathbf{y}_n \,|\, \mathbf{f}_n)]. \tag{93}$$

It is also the case that the PEP scaling factor reverts to the identity in the limit,

$$\begin{aligned} \mathbf{R}_n &= {\mathbf{C}_n^{\backslash}}^{-1} \left(\alpha \nabla^2_{\mathbf{m}_n^{\backslash}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\backslash}) + {\mathbf{C}_n^{\backslash}}^{-1}\right)^{-1} \\ &= \left(\alpha \nabla^2_{\mathbf{m}_n^{\backslash}} \overline{\mathcal{L}}(\mathbf{m}_{k,n}^{\backslash})\mathbf{C}_n^{\backslash} + \mathbf{I}\right)^{-1} \\ \implies \lim_{\alpha \to 0} \mathbf{R}_n &= \mathbf{I} \end{aligned} \tag{94}$$

which shows that the PEP updates given by Equation (22) when $\alpha \to 0$ are equivalent to the natural gradient VI updates given by Equation (15).

Appendix E. Derivation of the PL Updates

After performing SLR on a single likelihood term, we obtain the approximation

$$p(\mathbf{y}_n \,|\, \mathbf{f}_n) \approx \mathrm{N}(\mathbf{y}_n \,|\, \mathbf{A}_n \mathbf{f}_n + \mathbf{b}_n, \boldsymbol{\Omega}_{n,n}), \tag{95}$$

where $\mathbf{A}_n = \mathbf{Q}_n^\top \mathbf{C}_{n,n}^{-1}$, $\mathbf{b}_n = \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]] - \mathbf{Q}_n^\top \mathbf{C}_{n,n}^{-1} \mathbf{m}_n$, and $\boldsymbol{\Omega}_{n,n} = \mathbf{S}_n - \mathbf{Q}_n^\top \mathbf{C}_{n,n}^{-1} \mathbf{Q}_n$ for,

$$\begin{aligned} \mathbf{S}_n =& \mathbb{E}_{q(\mathbf{f}_n)}\left[(\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n] - \mathbb{E}_{q(\mathbf{f}_n)}\left[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]\right])(\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n] - \mathbb{E}_{q(\mathbf{f}_n)}\left[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]\right])^\top + \mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]\right], \\ \mathbf{Q}_n =& \mathbb{E}_{q(\mathbf{f}_n)}\left[(\mathbf{f}_n - \mathbf{m}_n)(\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n] - \mathbb{E}_{q(\mathbf{f}_n)}\left[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]\right])^\top\right]. \end{aligned} \tag{96}$$

Given this Gaussian term, the question remains what its corresponding factor $t(\mathbf{f}_n)$ is, such that PL can be presented in the same light as the other approximate inference methods. If $\mathbf{A}_n$ is square and invertible, then finding $t(\mathbf{f}_n)$ is trivial, but we are interested in the more general case where the dimension of $\mathbf{f}_n$ and $\mathbf{y}_n$ may not be the same.

Fortunately, the EP and VI updates provide a means by which to compute a factor $t(\mathbf{f}_n)$ given a general likelihood model, and furthermore we know that these methods are exact in the Gaussian case (which PL provides after linearisation). Since the VI updates are slightly simpler than the EP ones, we proceed by treating $\mathrm{N}(\mathbf{y}_n \,|\, \mathbf{A}_n \mathbf{f}_n + \mathbf{b}_n, \boldsymbol{\Omega}_{n,n})$ as the target

distribution in Equation (15), which gives

$$\begin{aligned}
\mathcal{L}_n &= \mathbb{E}_{q(\mathbf{f}_n)}[\log \mathrm{N}(\mathbf{y}_n \,|\, \mathbf{A}_n \mathbf{f}_n + \mathbf{b}_n, \boldsymbol{\Omega}_{n,n})] \\
&= -\frac{1}{2}\log 2\pi - \frac{1}{2}\log \boldsymbol{\Omega}_{n,n} - \frac{1}{2}((\mathbf{y}_n - \mathbf{A}_n \mathbf{m}_n - \mathbf{b}_n)\boldsymbol{\Omega}_{n,n}^{-1}(\mathbf{y}_n - \mathbf{A}_n \mathbf{m}_n - \mathbf{b}_n)^\top + \mathbf{C}_{n,n}), \\
&= -\frac{1}{2}\log 2\pi - \frac{1}{2}\log \boldsymbol{\Omega}_{n,n} - \frac{1}{2}((\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]])\boldsymbol{\Omega}_{n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]])^\top + \mathbf{C}_{n,n}), \\
&= c - \frac{1}{2}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]])\boldsymbol{\Omega}_{n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]])^\top, \qquad (97)
\end{aligned}$$

where c is constant and contains the terms that do not depend on $\mathbf{m}_n$ (and hence do not effect the updates). Differentiating with respect to $\mathbf{m}_n$ gives

$$\begin{aligned}
\frac{\partial \mathcal{L}_n}{\partial \mathbf{m}_n^\top} &= \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]}{\partial \mathbf{m}_n}^\top \boldsymbol{\Omega}_{n,n}^{-1}(\mathbf{y}_n - \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]), \\
\frac{\partial^2 \mathcal{L}_n}{\partial \mathbf{m}_n^\top \partial \mathbf{m}_n} &= -\frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]}{\partial \mathbf{m}_n}^\top \boldsymbol{\Omega}_{n,n}^{-1} \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]]}{\partial \mathbf{m}_n}, \qquad (98)
\end{aligned}$$

which gives the desired updates in Equation (31).

Appendix F. Posterior Linearisation as a Bayes–Gauss–Newton Method

Posterior linearisation can be characterised as a Gauss–Newton algorithm. Whilst it is less clear what optimisation target PL minimises, the updates can be re-derived as the result of a least-squares problem by defining the surrogate target $\overline{\mathcal{L}}(\mathbf{m}_n, \mathbf{C}_{n,n}) = \log \mathrm{N}(\mathbf{y}_n \,|\, \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]],\, \boldsymbol{\Omega}_{n,n})$, and the least-squares residuals

$$\mathbf{V}(\mathbf{m}, \mathbf{C}) = \begin{bmatrix} \boldsymbol{\Omega}_{1,1}^{-\frac{1}{2}}(\mathbf{y}_1 - \mathbb{E}_{q(\mathbf{f}_1)}[\mathbb{E}[\mathbf{y}_1 \,|\, \mathbf{f}_1]]) \\ \vdots \\ \boldsymbol{\Omega}_{N,N}^{-\frac{1}{2}}(\mathbf{y}_N - \mathbb{E}_{q(\mathbf{f}_N)}[\mathbb{E}[\mathbf{y}_N \,|\, \mathbf{f}_N]]) \end{bmatrix}, \qquad (99)$$

such that $\overline{\mathcal{L}}(\mathbf{m}, \mathbf{C}) = \mathbf{V}(\mathbf{m}, \mathbf{C})^\top \mathbf{V}(\mathbf{m}, \mathbf{C})$. The residual Jacobian is then defined as

$$\nabla_{\mathbf{m}} \mathbf{V}(\mathbf{m}, \mathbf{C}) = \begin{bmatrix} -\boldsymbol{\Omega}_{1,1}^{-\frac{1}{2}} \nabla_{\mathbf{m}^\top} \mathbb{E}_{q(\mathbf{f}_1)}[\mathbb{E}[\mathbf{y}_1 \,|\, \mathbf{f}_1]] \\ \vdots \\ -\boldsymbol{\Omega}_{N,N}^{-\frac{1}{2}} \nabla_{\mathbf{m}^\top} \mathbb{E}_{q(\mathbf{f}_N)}[\mathbb{E}[\mathbf{y}_N \,|\, \mathbf{f}_N]] \end{bmatrix}, \qquad (100)$$

where

$$-\boldsymbol{\Omega}_{n,n}^{-\frac{1}{2}} \nabla_{\mathbf{m}^\top} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]] = \Big[\mathbf{0}, \ldots, -\boldsymbol{\Omega}_{n,n}^{-\frac{1}{2}} \nabla_{\mathbf{m}_n} \mathbb{E}_{q(\mathbf{f}_n)}[\mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]], \ldots, \mathbf{0}\Big]. \qquad (101)$$

Use of the Gauss–Newton approximation to the Hessian results in

$$\begin{aligned}
\nabla_{\mathbf{m}}^2 \overline{\mathcal{L}}(\mathbf{m}, \mathbf{C}) &\approx -\nabla_{\mathbf{m}} \mathbf{V}(\mathbf{m}, \mathbf{C})^\top \nabla_{\mathbf{m}} \mathbf{V}(\mathbf{m}, \mathbf{C}) \\
&= -\nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]]^\top \boldsymbol{\Omega}^{-1} \nabla_{\mathbf{m}} \mathbb{E}_{q(\mathbf{f})}[\mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]], \qquad (102)
\end{aligned}$$

which matches the PL updates. Note that it has been assumed that the gradient of $\boldsymbol{\Omega}_k$ is zero (see Section 5.3 for discussion). Since these terms take into account the full Bayesian posterior via the expectation with respect to $q(\mathbf{f})$, rather than a point estimate, we refer to PL as a *Bayes–Gauss–Newton* method.

Appendix G. Taylor Expansion / Extended Kalman Smoother as a Gauss–Newton Method

It is well known that the Taylor/EKS approach given in Section 4.3.1 is equivalent to a Gauss–Newton method (Bell, 1994). This can be seen by modifying the PL approach in Appendix F by setting $\boldsymbol{\Omega} = \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]$ and replacing the expectations with respect to $q(\mathbf{f})$ with point estimates at the mean. This gives

$$\mathbf{V}(\mathbf{f}) = \begin{bmatrix} \mathrm{Cov}[\mathbf{y}_1 \,|\, \mathbf{f}_1]^{-\frac{1}{2}} (\mathbf{y}_1 - \mathbb{E}[\mathbf{y}_1 \,|\, \mathbf{f}_1]) \\ \vdots \\ \mathrm{Cov}[\mathbf{y}_N \,|\, \mathbf{f}_N]^{-\frac{1}{2}} (\mathbf{y}_N - \mathbb{E}[\mathbf{y}_N \,|\, \mathbf{f}_N]) \end{bmatrix}, \tag{103}$$

such that $\overline{\mathcal{L}}(\mathbf{f}) = \log \mathrm{N}(\mathbf{y} \,|\, \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}], \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]) = \mathbf{V}(\mathbf{f})^\top \mathbf{V}(\mathbf{f}) + c$. The residual Jacobian is then defined as

$$\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{f}) = \begin{bmatrix} -\mathrm{Cov}[\mathbf{y}_1 \,|\, \mathbf{f}_1]^{-\frac{1}{2}} \nabla_{\mathbf{f}^\top} \mathbb{E}[\mathbf{y}_1 \,|\, \mathbf{f}_1] \\ \vdots \\ -\mathrm{Cov}[\mathbf{y}_N \,|\, \mathbf{f}_N]^{-\frac{1}{2}} \nabla_{\mathbf{f}^\top} \mathbb{E}[\mathbf{y}_N \,|\, \mathbf{f}_N] \end{bmatrix}, \tag{104}$$

where

$$-\mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}} \nabla_{\mathbf{f}^\top} \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n] = \Big[\mathbf{0}, \ldots, -\mathrm{Cov}[\mathbf{y}_n \,|\, \mathbf{f}_n]^{-\frac{1}{2}} \nabla_{\mathbf{f}_n} \mathbb{E}[\mathbf{y}_n \,|\, \mathbf{f}_n]], \ldots, \mathbf{0}\Big]. \tag{105}$$

Use of the Gauss–Newton approximation to the Hessian results in,

$$\begin{aligned} \nabla^2_{\mathbf{f}} \overline{\mathcal{L}}(\mathbf{m}) &\approx -\nabla_{\mathbf{f}} \mathbf{V}(\mathbf{m})^\top \nabla_{\mathbf{f}} \mathbf{V}(\mathbf{m}) \\ &= -\nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{m}]^\top \, \mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]^{-1} \nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{m}], \end{aligned} \tag{106}$$

where $\nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{m}] := \nabla_{\mathbf{f}} \mathbb{E}[\mathbf{y} \,|\, \mathbf{f}]_{\mathbf{f}=\mathbf{m}}$, which matches the EKS updates. Again note that it has been assumed that the gradient of $\mathrm{Cov}[\mathbf{y} \,|\, \mathbf{f}]$ is zero.

Appendix H. Full Algorithm Descriptions

Here we outline the exact algorithms used to perform inference in GPs, sparse GPs and Markovian GPs. The 'update rule' $\Lambda(\cdot)$ can be chosen to be any of the local likelihood updates listed in the main paper. Newton: Equation (8), variational inference: Equation (15), power expectation propagation: Equation (22) (update rule acts on cavity marginals rather than posterior marginals), posterior linearisation: Equation (31), extended Kalman smoother: Equation (32), second-order posterior linearisation: Equation (45), quasi-Newton: Equation (49), variational quasi-Newton: Equation (51), power expectation propagation quasi-Newton: Equation (53), posterior linearisation quasi-Newton: Equation (54). A Gauss–Newton approximation to the Hessian can be obtained by applying Equation (38), **??**, Equation (40), Equation (43), or Equation (44). PSD constraints via Riemannian gradients can be applied using Equation (58) or Equation (59).

Algorithm 1 Gaussian process inference

Input: data: $\{\mathbf{X}, \mathbf{y}\}$, kernel: $\kappa(\cdot)$, update rule: $\Lambda(\cdot)$, initial approx. likelihood: $\{\overline{\boldsymbol{\lambda}}^{(1)}, \overline{\boldsymbol{\lambda}}^{(2)}\}$, learning rate: ρ
$\mathbf{K} = \kappa(\mathbf{X}, \mathbf{X})$
while energy not converged **do**
  Convert approximate likelihood to mean/cov:
  $\overline{\mathbf{C}} = -\frac{1}{2}(\overline{\boldsymbol{\lambda}}^{(2)})^{-1}$
  $\overline{\mathbf{m}} = \overline{\mathbf{C}}\overline{\boldsymbol{\lambda}}^{(1)}$
  Compute the approximate posterior:
  $\mathbf{m} = \mathbf{K}(\mathbf{K} + \overline{\mathbf{C}})^{-1}\overline{\mathbf{m}}$
  $\mathbf{C} = \mathbf{K} - \mathbf{K}(\mathbf{K} + \overline{\mathbf{C}})^{-1}\mathbf{K}$
  Update the approximate likelihood:
  $\mathbf{J}_n, \mathbf{H}_{n,n} = \Lambda(\mathbf{m}_n, \mathbf{C}_{n,n}, \mathbf{y}_n) \quad \forall n$
  $\overline{\boldsymbol{\lambda}}^{(2)} = (1-\rho)\,\overline{\boldsymbol{\lambda}}^{(2)} + \rho\,\frac{1}{2}\mathbf{H}$
  $\overline{\boldsymbol{\lambda}}^{(1)} = (1-\rho)\,\overline{\boldsymbol{\lambda}}^{(1)} + \rho\,(\mathbf{J} - \mathbf{H}\,\mathbf{m})$
end while

Algorithm 2 Stochastic sparse GP inference

Input: data: $\{\mathbf{X}, \mathbf{y}\}$, inducing inputs: $\mathbf{Z}$, kernel: $\kappa(\cdot)$, update rule: $\Lambda(\cdot)$, initial approx. likelihood: $\{\overline{\boldsymbol{\lambda}}^{(1)}, \overline{\boldsymbol{\lambda}}^{(2)}\}$, learning rate: ρ, batch size: N_b
$\mathbf{K}_{\mathbf{uu}} = \kappa(\mathbf{Z}, \mathbf{Z})$
$\mathbf{K}_{\mathbf{f}_n\mathbf{u}} = \kappa(\mathbf{X}_n, \mathbf{Z}) \quad \forall n$
$\mathbf{W}_{\mathbf{f}_n\mathbf{u}} = \mathbf{K}_{\mathbf{f}_n\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1} \quad \forall n$
while energy not converged **do**
  Convert approximate likelihood to mean/cov:
  $\overline{\mathbf{C}}_{\mathbf{u}} = -\frac{1}{2}(\mathbf{W}_{\mathbf{fu}}^\top \overline{\boldsymbol{\lambda}}^{(2)} \mathbf{W}_{\mathbf{fu}})^{-1}$
  $\overline{\mathbf{m}}_{\mathbf{u}} = \overline{\mathbf{C}}_{\mathbf{u}} \mathbf{W}_{\mathbf{fu}}^\top \overline{\boldsymbol{\lambda}}^{(1)}$
  Compute the approximate posterior:
  $\mathbf{m}_{\mathbf{u}} = \mathbf{K}_{\mathbf{uu}}(\mathbf{K}_{\mathbf{uu}} + \overline{\mathbf{C}}_{\mathbf{u}})^{-1}\overline{\mathbf{m}}_{\mathbf{u}}$
  $\mathbf{C}_{\mathbf{u}} = \mathbf{K}_{\mathbf{uu}} - \mathbf{K}_{\mathbf{uu}}(\mathbf{K}_{\mathbf{uu}} + \overline{\mathbf{C}}_{\mathbf{u}})^{-1}\mathbf{K}_{\mathbf{uu}}$
  Compute the posterior marginals $\forall n$ in batch:
  $\mathbf{m}_n = \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{m}_{\mathbf{u}}$
  $\mathbf{C}_{n,n} = \kappa(\mathbf{X}_n, \mathbf{X}_n) - \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{K}_{\mathbf{f}_n\mathbf{u}}^\top + \mathbf{W}_{\mathbf{f}_n\mathbf{u}}\mathbf{C}_{\mathbf{u}}\mathbf{W}_{\mathbf{f}_n\mathbf{u}}^\top$
  Update the approximate likelihood:
  $\mathbf{J}_n, \mathbf{H}_{n,n} = \Lambda(\mathbf{m}_n, \mathbf{C}_{n,n}, \mathbf{y}_n) \quad \forall n$ in batch
  $\overline{\boldsymbol{\lambda}}^{(2)} = (1-\rho)\,\overline{\boldsymbol{\lambda}}^{(2)} + \rho\,\frac{1}{2}\mathbf{H}$
  $\overline{\boldsymbol{\lambda}}^{(1)} = (1-\rho)\,\overline{\boldsymbol{\lambda}}^{(1)} + \rho\,(\mathbf{J} - \mathbf{H}\,\mathbf{m})$
end while

Algorithm 3 State space model / Markovian GP inference

Input: data: $\{\mathbf{X}, \mathbf{y}\}$, update rule: $\Lambda(\cdot)$, model matrices: $\{\mathbf{A}_n, \mathbf{Q}_n, \bar{\mathbf{H}}\}_{n=1}^N$, initial state: $\{\bar{\mathbf{m}}_0, \bar{\mathbf{P}}_0\}$, initial approx. likelihood: $\{\bar{\boldsymbol{\lambda}}^{(1)}, \bar{\boldsymbol{\lambda}}^{(2)}\}$, learning rate: ρ
while energy not converged **do**
  Convert approximate likelihood to mean/cov:
  $\overline{\mathbf{C}} = -\frac{1}{2}(\bar{\boldsymbol{\lambda}}^{(2)})^{-1}$
  $\overline{\mathbf{m}} = \overline{\mathbf{C}}\bar{\boldsymbol{\lambda}}^{(1)}$
  Compute the approximate posterior:
  for $n = 1 : N$ **do**
    $\bar{\mathbf{m}}_n = \mathbf{A}_n\bar{\mathbf{m}}_{n-1}, \quad \bar{\mathbf{P}}_n = \mathbf{A}_n\bar{\mathbf{P}}_{n-1}\mathbf{A}_n^\top + \mathbf{Q}_n$
    $\mathbf{V}_n = \bar{\mathbf{H}}\bar{\mathbf{P}}_n\bar{\mathbf{H}}^\top + \overline{\mathbf{C}}_{n,n}, \quad \mathbf{W}_n = \bar{\mathbf{P}}_n\bar{\mathbf{H}}^\top\mathbf{V}_n^{-1}$
    $\bar{\mathbf{m}}_n = \bar{\mathbf{m}}_n + \mathbf{W}_n(\overline{\mathbf{m}}_n - \bar{\mathbf{H}}\bar{\mathbf{m}}_n)$
    $\bar{\mathbf{P}}_n = \bar{\mathbf{P}}_n - \mathbf{W}_n\mathbf{V}_n\mathbf{W}_n^\top$
  end for
  for $n = N - 1 : 1$ **do**
    $\mathbf{G}_n = \bar{\mathbf{P}}_n\mathbf{A}_{n+1}\bar{\mathbf{P}}_n^{-1}$
    $\mathbf{R}_{n+1} = \mathbf{A}_{n+1}\bar{\mathbf{P}}_n\mathbf{A}_{n+1}^\top + \mathbf{Q}_{n+1}$
    $\bar{\mathbf{m}}_n = \bar{\mathbf{m}}_n + \mathbf{G}_n(\bar{\mathbf{m}}_{n+1} - \mathbf{A}_{n+1}\bar{\mathbf{m}}_n)$
    $\bar{\mathbf{P}}_n = \bar{\mathbf{P}}_n + \mathbf{G}_n(\bar{\mathbf{P}}_{n+1} - \mathbf{R}_{n+1})\mathbf{G}_n^\top$
    $\mathbf{m}_n = \bar{\mathbf{H}}\bar{\mathbf{m}}_n$
    $\mathbf{C}_{n,n} = \bar{\mathbf{H}}\bar{\mathbf{P}}_n\bar{\mathbf{H}}^\top$
  end for
  Update the approximate likelihood:
  $\mathbf{J}_n, \mathbf{H}_{n,n} = \Lambda(\mathbf{m}_n, \mathbf{C}_{n,n}, \mathbf{y}_n) \quad \forall n$
  $\bar{\boldsymbol{\lambda}}^{(2)} = (1-\rho)\,\bar{\boldsymbol{\lambda}}^{(2)} + \rho\,\frac{1}{2}\mathbf{H}$
  $\bar{\boldsymbol{\lambda}}^{(1)} = (1-\rho)\,\bar{\boldsymbol{\lambda}}^{(1)} + \rho\,(\mathbf{J} - \mathbf{H}\,\mathbf{m})$
end while

Appendix I. Reproducibility

Scripts to reproduce the results in this paper can be found in the experiments folder in the code repository: `https://github.com/AaltoML/BayesNewton/tree/main/experiments`. The individual experiments can be found in the `motorcycle`, `product` and `gprn` folders respectively. Each folder contains a main Python script, plus bash scripts to produce the results for each inference method class: `bn-newton.sh`, `bn-vi.sh`, `bn-ep.sh`, `bn-pl.sh`. After these have finished running, the `results_bn.py` script can then be run to produce the plots.