NeuroImage 279 (2023) 120310

Contents lists available at ScienceDirect

NeuroImage

journal homepage: www.elsevier.com/locate/ynimg

A primer on Variational Laplace (VL)

Peter Zeidman *, Karl Friston, Thomas Parr

Wellcome Centre for Human Neuroimaging, UCL, 12 Queen Square, London WC1N 3AR, United Kingdom

ARTICLE INFO

Keywords:
Variational Laplace
Bayes
DCM
Modelling

ABSTRACT

This article details a scheme for approximate Bayesian inference, which has underpinned thousands of neuroimaging studies since its introduction 15 years ago. Variational Laplace (VL) provides a generic approach to fitting linear or non-linear models, which may be static or dynamic, returning a posterior probability density over the model parameters and an approximation of log model evidence, which enables Bayesian model comparison. VL applies variational Bayesian inference in conjunction with quadratic or Laplace approximations of the evidence lower bound (free energy). Importantly, update equations do not need to be derived for each model under consideration, providing a general method for fitting a broad class of models. This primer is intended for experimenters and modellers who may wish to fit models to data using variational Bayesian methods, without assuming previous experience of variational Bayes or machine learning. Accompanying code demonstrates how to fit different kinds of model using the reference implementation of the VL scheme in the open-source Statistical Parametric Mapping (SPM) software package. In addition, we provide a standalone software function that does not require SPM, in order to ease translation to other fields, together with detailed pseudocode. Finally, the supplementary materials provide worked derivations of the key equations.

1. Introduction

Scientific enquiry typically involves inferring quantities that cannot be directly observed. For example, a neuroscientist may wish to investigate the activity of neural populations from electrical activity that can be measured on the scalp. Similarly, a seismologist may wish to make inferences about geophysical events that occur beneath the surface of the earth, from seismograph measurements taken at the surface. Both of these are examples of *ill-posed problems*, where multiple configurations of the underlying system of interest could lead to similar measurements. Consequently, any inferences that are made about the underlying mechanisms that generate the data will typically involve uncertainty. This uncertainty needs to be quantified when drawing conclusions, which is why probabilistic, or *Bayesian* inference methods are typically used.

This article explains the mathematics behind a scheme for Bayesian modelling called Variational Laplace (VL), which is widely used in neuroimaging. It is used to test the statistical evidence for competing models, and in tandem, to rapidly estimates models' parameters, without the need for computationally demanding sampling procedures found in other Bayesian approaches. VL is the cornerstone of Dynamic Causal Modelling (DCM) for fMRI and M/EEG - a framework for inferring neural connectivity from non-invasive neuroimaging data (Friston et al., 2003). A special case of VL, referred to as variational Restricted Maximum Likelihood (REML), is applied behind the scenes in every Statistical Parametric Mapping (SPM) analysis for estimating temporal auto-correlation, as well as for Bayesian source localisation with M/EEG data (Friston et al., 2008a). The original implementation of VL was in the SPM software package (Friston et al., 2007), and variants of the scheme are now provided in other packages for analysing neural and psychological data, such as the VBA Toolbox (Daunizeau et al., 2014) and TAPAS (Frässle et al., 2021). It is also increasingly being applied to fields beyond neuroimaging, including theoretical neurobiology (Smith et al., 2022), robotics (Lanillos et al., 2021; Lanillos and van Gerven, 2021) and epidemiology (Friston et al., 2020).

Despite its widespread use, understanding VL can be challenging for the uninitiated; the original description of VL assumed some familiarity with variational methods, statistical physics and Bayesian inference. Here, our aim is didactic. We explain the methodology from first principles, which we anticipate will be particularly useful for people interested in developing new models of neuroimaging data, or new software toolboxes for neuroimaging analysis. Additionally, we hope this article will be useful for experimenters, who wish to gain a deeper understanding of how the analyses they routinely conduct are performed

* Corresponding author.
E-mail address: peter.zeidman@ucl.ac.uk (P. Zeidman).

https://doi.org/10.1016/j.neuroimage.2023.120310
Received 30 September 2022; Received in revised form 13 July 2023; Accepted 4 August 2023
Available online 5 August 2023

under the hood. Finally, VL may have relevance to other fields, including machine learning and other physical / biological sciences. We have therefore provided generic pseudocode and MATLAB code with this paper, which could ease translation for new applications. This article compliments previous tutorial introductions to variational Bayes in other contexts (Chappell et al., 2016; Ostwald et al., 2014), and technical reports which have set out the mathematics of VL in detail (Daunizeau, 2017; Stephan et al., 2005; Friston et al., 2007; Starke and Ostwald, 2017).

1.1. Example problem: modelling neural connectivity

To motivate the use of VL, we consider the following classic fMRI study, which has been used to develop and illustrate many new analysis methods in the decades since it was published. Büchel et al. (1998) investigated area V5 of the visual cortex, which was known to be sensitive to visual motion. While undergoing fMRI, participants viewed white dots on a computer screen, which were either in motion or stationary. On a subset of the trials with motion, participants were instructed to pay attention to the *speed* of the dots' motion. The authors found that the neural response to visual motion in V5 was enhanced when people paid attention to the speed of the moving dots. Attention also enhanced the neural response in brain regions lower in the visual hierarchy (primary visual cortex) and higher in the hierarchy (superior parietal cortex).

Here, for illustrative purposes, we will use the fMRI data from a single participant from Büchel et al. (1998) to address the following question: which neural connections explain why V5 was sensitive to visual motion? We consider two candidate hypotheses[1]:

- H1: Attention modulated *bottom-up* connectivity from primary visual cortex (V1) to V5
- H2: Attention modulated *top-down* connectivity from superior parietal cortex (SPC) to V5

Fig. 1A shows the data – fMRI timeseries that were extracted from brain regions V1, V5 and SPC. To formalize the two hypotheses, the next step is to specify models that can generate or simulate the fMRI data that would be expected under each hypothesis, before using the VL scheme to evaluate their evidence.

1.1.1. From hypothesis to models

Models commonly used for fMRI analysis, referred to as Dynamic Causal Models, are specified as follows. Defining a vector $z(t) = (z_1(t), z_2(t), z_3(t))$ to be the overall level of neural activity in V1, V5 and SPC respectively at time t, and vector $\boldsymbol{y}(t)$ to be the (concatenated) fMRI timeseries from all three regions, either hypothesis H1 or H2 can be expressed using the following pair of equations (a state-space model):

$$\begin{aligned} \dot{z}(t) &= \boldsymbol{J}(t)z(t) + \boldsymbol{C}u_1(t) \\ \boldsymbol{y} &= g(z, \boldsymbol{\theta}_h) + \boldsymbol{\epsilon}_y \end{aligned} \tag{1}$$

The first line, which we will call the neural model, describes how the rate of change in the three brain regions, $\dot{z}$, depends on a time-varying connectivity matrix $\boldsymbol{J}(t)$, which is of dimension $[3 \times 3]$, and vector of external driving inputs $\boldsymbol{C}u_1(t)$. The second line of Eq. (1) includes an *observation model*, g, which translates from neural activity z to fMRI data $\boldsymbol{y}$. This part of the model is governed by a vector of parameters $\boldsymbol{\theta}_h$, which, for MRI, includes the rate of blood flow through the venous compartment. This model, together with the Bayesian methods set out here, are together referred to as DCM for fMRI (Friston et al., 2003).

The parameterisation of $\boldsymbol{J}(t)$ determines which connections are switched on (informed by the data) and which are switched off (fixed at zero). We can therefore formalize our hypotheses by specifying two variants of the model, m_1 and m_2, that differ in which connections can be modulated by attention, as illustrated in Fig. 1B. The corresponding parametrisation of $\boldsymbol{J}(t)$ for the two models is provided in Appendix 1, although this is not required for understanding the rest of this article.

Given these two hypotheses, H1 and H2, which we have now formally stated as mathematical models m_1 and m_2, we have two overall aims:

1 To estimate the models' parameters, which includes the strength of neural connections and the effects of each experimental condition on each connection.
2 To estimate the probability for each model given the data, $P(m_1|\boldsymbol{y})$ and $P(m_2|\boldsymbol{y})$, enabling us to select the best model or models (referred to as Bayesian model selection). This requires the intermediate step of estimating the *model evidence*, also called the *marginal likelihood*, which is the probability of having seen the data under each model, $P(\boldsymbol{y}|m_1)$ and $P(\boldsymbol{y}|m_2)$.

Both objectives are fulfilled by the VL scheme, which we will introduce next.

1.2. VL in a nutshell

To explain the problem that VL solves from a statistical perspective, we begin with Bayes rule. For a model with a vector of parameters $\boldsymbol{\theta}$, we start by defining a *prior* probability density $P(\boldsymbol{\theta})$. This function defines our belief about any particular value of the parameters, or range of values, before seeing the data. The priors serve to regularize or constrain the estimation of the model, making ill-posed problems tractable – in the sense there is a unique solution. We also define a likelihood $P(\boldsymbol{y}|\boldsymbol{\theta})$, which is a function of the parameters $\boldsymbol{\theta}$ and returns the probability of observing the data $\boldsymbol{y}$ given those parameters. Together, the priors and likelihood form a *generative model* of the data. The goal of Bayesian inference, as introduced above, is two-fold. First, to obtain an updated probability density over the parameters after seeing the data – the posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$. Second, to obtain the *marginal likelihood* or *model evidence* $P(\boldsymbol{y})$, which scores the quality of the model and enables models to be compared. If the data 'look like' the kind of data that the model would have predicted, then $P(\boldsymbol{y})$ will be large, whereas if the data are inconsistent with the model, then $P(\boldsymbol{y})$ will be smaller, and thus models can be compared on the basis of their evidence. This is called *Bayesian model comparison* and we will return later to why the model evidence is well-suited to comparing models. The posterior and evidence are related by Bayes rule:

$$P(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{P(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{P(\boldsymbol{y})} = \frac{P(\boldsymbol{y}, \boldsymbol{\theta})}{P(\boldsymbol{y})} \tag{2}$$

Where the model evidence is the integral or sum over possible settings of the parameters:

$$P(\boldsymbol{y}) = \int P(\boldsymbol{y}, \boldsymbol{\theta}) d\boldsymbol{\theta} \tag{3}$$

Thus, calculating the model evidence involves marginalising (summing or integrating) over the parameters. Unfortunately, this integral typically lacks an analytic solution and cannot be calculated directly. This is problematic for the calculating model evidence, but also the posterior probability (which requires the evidence).

It is the intractability of the integral in Eq. (3) that necessities approximate Bayesian inference. Traditional sampling methods are a common approach for approximating the posterior, which eschew the need to tackle this integral, however they can be very computationally intensive and do not provide a straight-forward way to approximate the evidence, thereby precluding Bayesian model comparison. Instead, the

[1] These models are used as teaching examples, which we supply with the SPM software package. The dataset and analysis scripts, can be freely downloaded from https://www.fil.ion.ucl.ac.uk/spm/data/attention/ .

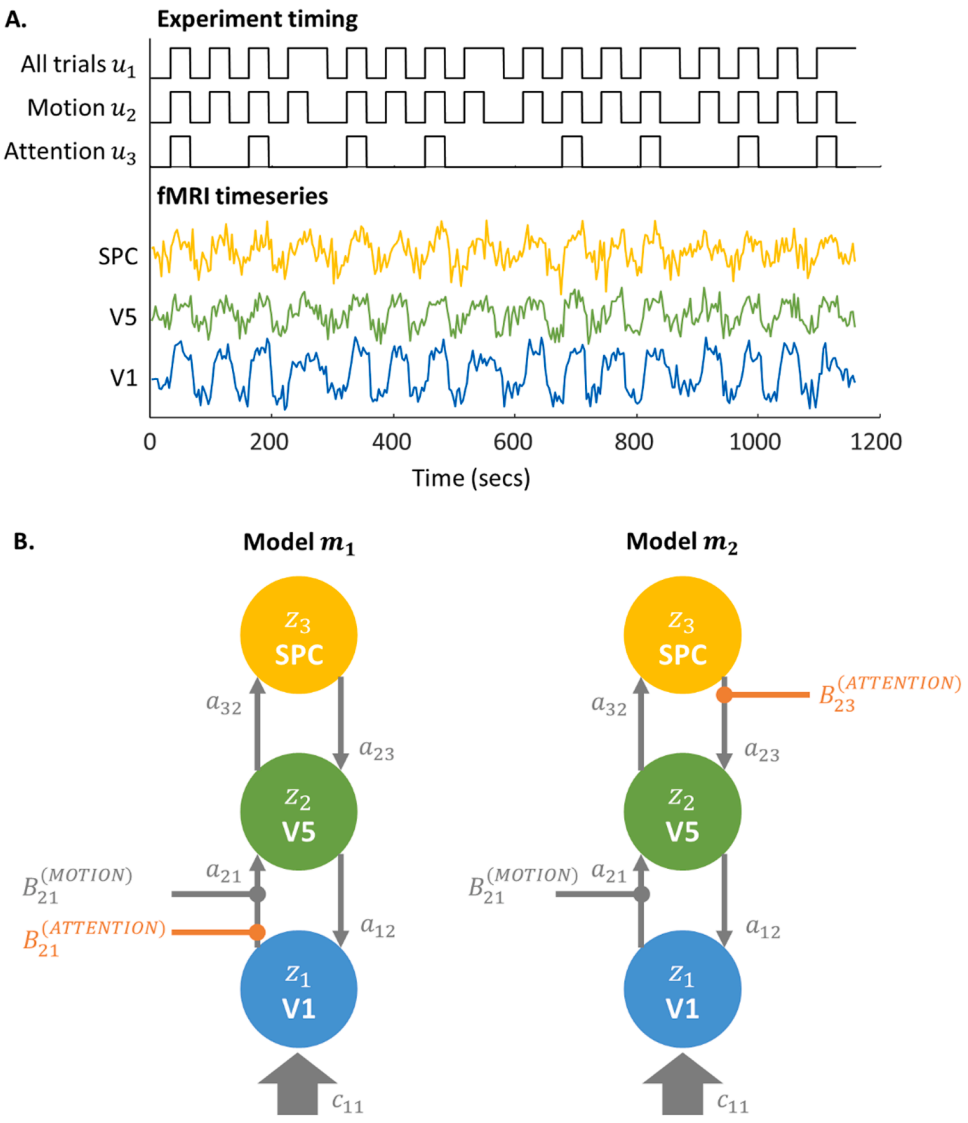


Fig. 1. Exampling modelling problem. A. Timing of three experimental conditions (top) and representative timeseries fMRI from three brain regions that will be the focus of the analysis (bottom). **B.** Structure of two candidate Dynamic Causal Models (DCMs) used to explain these fMRI data. The models differ only in where Attention has an effect. Each large coloured circle is a brain region, with latent level of neural activity z_1, z_2, z_3 respectively. Arrows between the circles are neural connections encoded in matrix A of the model (see Appendix 1). Arrows with rounded ends encode modulatory effects of Motion and Attention.

approach described here is to construct a quantity that lower bounds the log of the evidence (i.e., is always smaller than or equal to it). This is often referred to as an evidence lower bound (ELBO) or a free energy functional.[2] An algorithm is then derived to maximise this bound – making it as close as possible to the log evidence. Methods that involve this optimisation of a bound – and therefore ensure that the ELBO or free energy approximates the log evidence – are collectively referred to as 'Variational Bayes' (VB). These methods were first introduced by Richard Feynman in statistical physics (Feynman, 1972) and terms such as 'free energy' were retained when it was subsequently applied in different fields. Variational methods were introduced into machine learning though ensemble learning (Hinton and Van Camp, 1993; MacKay, 1995b; MacKay, 1995a). Later, schemes like expectation maximisation (EM) were considered in the light of VB (Bishop, 1998; Neal and Hinton, 1998; Beal, 2003), which proved particularly useful for fitting graphical models (Jordan et al., 1999).

A key practical challenge for the routine use of VB is deriving the necessary model-fitting algorithm for a given model. This is time-consuming and requires a certain degree of skill with variational calculus. Approaches have therefore been developed over the years for implementing VB for a sufficiently broad class of models that new models can be introduced and fitted to data using 'plug-and-play' software routines. VL is one such approach (Friston et al., 2007), which has been employed in a large body of work in neuroscience. The reference implementation of VL is a MATLAB function, spm_nlsi_gn,[3] implemented in the Statistical Parametric Mapping (SPM) software package (https://www.fil.ion.ucl.ac.uk/spm/).

We proceed by deriving a score for the quality of a model: the free energy bound on the log evidence. Then, we derive the algorithm that optimises this bound, providing estimates of the log evidence and posterior over parameters. We illustrate this with worked examples, including the attention to visual motion example above, code for which is provided with this paper. Finally, in the discussion, we consider some advantages and disadvantages of this scheme and its relation to other

[2] A *functional* is a function of a function. More specifically - at least for the purposes of this paper - it is a mapping whose input is a function and whose output is a number.

[3] This routine is particularly useful for fitting continuous data when the likelihood has a Gaussian form (whose expectation and covariance may be non-linear functions of the parameters). An alternative MATLAB function, spm_nlsi_Newton, allows the specification of generic likelihood functions (for example, it could be used for fitting binary data using likelihood densities compatible with logistic regression models).

variational inference software tools currently in use. An outline of mathematical notation appears in the footnote.[4]

2. The generative model

The VL scheme works with models that can be written generically in the following form. We have a vector of observed data $\boldsymbol{y}$ of length D and a model $g(\boldsymbol{\beta})$, where $\boldsymbol{\beta}$ are the model parameters:

$$\boldsymbol{y} = g(\boldsymbol{\beta}) + \boldsymbol{\epsilon}_y \tag{4}$$

In the example above, $\boldsymbol{y}$ are fMRI timeseries data and g is a model of how the data were generated, however VL is generic and any data or model could be used. The vector of errors or residuals ϵ_y has a multivariate normal density, with mean zero, and precision matrix $\boldsymbol{\Pi}_y$ of dimension D (which is the inverse of the covariance matrix):

$$\boldsymbol{\epsilon}_y \sim N\left(\boldsymbol{0}, \boldsymbol{\Pi}_y^{-1}\right) \tag{5}$$

Elements on the leading diagonal of matrix $\boldsymbol{\Pi}_y$ are the precision (inverse variance) of each measurement, and any non-zero off-diagonal elements encode the conditional dependencies amongst measurements, which determines their correlation. To parameterize $\boldsymbol{\Pi}_y$ – such that it can be estimated from the data – we decompose it into a weighted mixture of K matrices called *precision components*, $\boldsymbol{Q}_{k=1,\ldots,K} \in \mathbb{R}_{D\times D}$, each of which has a corresponding scalar hyperparameter $\lambda = (\lambda_1, \ldots \lambda_k)$:

$$\boldsymbol{\Pi}_y(\boldsymbol{\lambda}) = \exp(\lambda_1)\boldsymbol{Q}_1 + \ldots + \exp(\lambda_k)\boldsymbol{Q}_k \tag{6}$$

The exponential of each hyperparameter is taken to enforce positivity (because precisions and variances cannot be negative). This approach provides a convenient method for allowing different mixtures of observations to share variance. At its simplest, a single precision component can be used, set to the identity matrix, $\boldsymbol{Q}_1 = \boldsymbol{I}_D$, in which case the corresponding hyperparameter λ_1 encodes the log precision of the observation noise.

To make this into a statistical model, we define a likelihood function $P(\boldsymbol{y}|\boldsymbol{\theta})$ and a prior probability density $P(\boldsymbol{\theta})$, where the parameters and hyperparameters are $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\lambda})$. As set out in the previous section, the likelihood $P(\boldsymbol{y}|\boldsymbol{\theta})$ returns the probability of observing the data $\boldsymbol{y}$ given a particular setting of the parameters $\boldsymbol{\theta}$. This has a multivariate normal density:

$$P(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\lambda}) = N\left(g(\boldsymbol{\beta}), \boldsymbol{\Pi}_y(\boldsymbol{\lambda})^{-1}\right) \tag{7}$$

Eq. (7) states that the observations are expected to be centred on the prediction of the model $g(\boldsymbol{\beta})$, with the level of observation noise given by the precision matrix $\boldsymbol{\Pi}_y(\boldsymbol{\lambda})$. Next, the prior density $P(\boldsymbol{\theta})$ quantifies our belief about any given value of the parameters before performing the analysis. Here, we define multivariate normal densities as priors over the parameters and hyperparameters:

$$\begin{aligned} P(\boldsymbol{\beta}) &= N\left(\boldsymbol{\eta}_{\boldsymbol{\beta}}, \boldsymbol{\Pi}_{\boldsymbol{\beta}}^{-1}\right) \\ P(\boldsymbol{\lambda}) &= N\left(\boldsymbol{\eta}_{\boldsymbol{\lambda}}, \boldsymbol{\Pi}_{\boldsymbol{\lambda}}^{-1}\right) \end{aligned} \tag{8}$$

Having defined the model, we next set out methods for approximating the log of the model evidence $\ln P(\boldsymbol{y})$ and the posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$.

[4] The following mathematical notation is used. **Variables**: Lower case italic text (x) is used for scalar variables, bold uppercase letters ($\boldsymbol{Y}$) are used for matrices and bold lower-case letters are used for vectors ($\boldsymbol{y}$). The expected value or average of variable x under the density Q is written $E_Q[x]$. **Calculus**: The partial derivative of a function $h(x)$ with respect to x is written $\partial_x\, h(x)$, therefore $\partial_x = \frac{\partial}{\partial_x}$. The second derivative of $h(x)$ is written $\partial_{xx}\, h(x)$. **Variational calculus:** The *variation* of a functional δ may be thought of as the derivative of the functional with respect to the function. For instance, for a functional of the form $A = \int S(h(x))dx$, the variation of A with respect to h is: $\delta_{h(x)}S$.

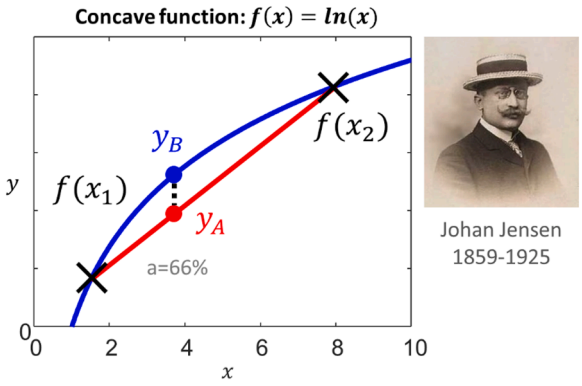


Fig. 2. Jensen's inequality. The curved blue line could be any concave function $y = f(x)$. Here as an example, the function $f(x) = \ln(x)$ is shown, which has been evaluated in the range $x = [0,\ 10]$. The straight red *secant line* joints two arbitrary points: $f(x_1)$ and $f(x_2)$. Any point on the secant line, such as y_A, is a weighted average of $f(x_1)$ and $f(x_2)$: $y_A = af(x_1) + (1 - a)f(x_2)$. Jensen's inequality says that this average-of-functions will always be less than or equal to the corresponding function-of-the-average: $y_B = f(ax_1 + (1 - a)x_2)$. Thus, using the notation of expected values, $E[f(x)] \leq f(E[X])$. Portrait in the public domain via *Wikipedia*. https://en.wikipedia.org/wiki/Johan_Jensen_(mathematician).

3. Variational Bayes

We will convert the difficult problem of calculating the integral in Eq. (3) into a simpler optimization or search problem. This begins by defining a functional called the free energy, which is a *lower bound* on the log of the model evidence $\ln P(\boldsymbol{y})$. This means that by construction, it can only return values that are less than or equal to the log evidence. Then we'll search for a setting of the parameters of the free energy (*variational parameters*) that maximize it, making it as close as possible to the unknown log evidence. Helpfully, the parameters that maximize the bound will turn out to approximate the posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$.

3.1. Constructing a lower bound on the log evidence

The log evidence is defined as:

$$\ln P(\boldsymbol{y}) = \ln \int P(\boldsymbol{y}, \boldsymbol{\theta}) d\boldsymbol{\theta} \tag{9}$$

To construct a lower bound on this quantity, we will make use of Jensen's inequality, which says that the *average of a log* is always less than or equal to the *log of an average* (Fig. 2). That means that if we can express $\ln P(\boldsymbol{y})$ as the average of a log, then to construct a lower bound we simply need to rearrange terms to get the log of an average.

To express the log evidence as the average of a log, we first define a probability density $Q(\boldsymbol{\theta})$ that will form our approximation of the posterior over parameters. We then introduce this density into the log evidence (from Eq. (9)), multiplying and dividing so that it causes no overall change to the evidence:

$$\begin{aligned} \ln\, P(\boldsymbol{y}) &= \ln \int \frac{Q(\boldsymbol{\theta})}{Q(\boldsymbol{\theta})} P(\boldsymbol{y}, \boldsymbol{\theta})\, d\boldsymbol{\theta} \\ &= \ln\, E_{Q(\theta)}\left[\frac{P(\boldsymbol{y}, \boldsymbol{\theta})}{Q(\boldsymbol{\theta})}\right] \end{aligned} \tag{10}$$

Where $E[\cdot]$ is the expected value or average.[5] This is the log of an average, so (by Jensen's inequality) a lower bound on this will be the average of the log, which is called the free energy F:

$$F[Q(\boldsymbol{\theta})] = E_{Q(\boldsymbol{\theta})}\left[\ln \frac{P(\boldsymbol{y},\boldsymbol{\theta})}{Q(\boldsymbol{\theta})}\right] \tag{11}$$

Here, we have written the free energy as a functional (a function of a function), with $Q(\boldsymbol{\theta})$ as its input. By construction, for any choice of probability density $Q(\boldsymbol{\theta})$, it holds that $F[Q(\boldsymbol{\theta})] \leq \ln P(\boldsymbol{y})$. Note that the free energy is sometimes defined to be the negative of this quantity. We have used the terminology typically found in the neuroimaging literature, where the free energy is a lower bound on the log evidence, also known as the ELBO in statistical and machine learning. The next step will be to maximize this lower bound, i.e., find the probability density $Q(\boldsymbol{\theta})$ that maximizes the free energy, bringing it close to the log evidence, so $F \approx \ln P(\boldsymbol{y})$.

3.2. Useful properties of the free energy

The free energy can be rewritten in different ways to illustrate why it serves as a useful score for the quality of a model. First, we can rewrite it as the log evidence minus the Kullback-Leibler (KL) divergence between the true and approximate posterior:

$$\begin{aligned} F[Q(\boldsymbol{\theta})] &= E_{Q(\boldsymbol{\theta})}\left[\ln \frac{P(\boldsymbol{\theta}|\boldsymbol{y})P(\boldsymbol{y})}{Q(\boldsymbol{\theta})}\right] \\ &= E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{\theta}|\boldsymbol{y}) + \ln P(\boldsymbol{y}) - \ln Q(\boldsymbol{\theta})] \\ &= \ln P(\boldsymbol{y}) + E_{Q(\boldsymbol{\theta})}\left[\ln \frac{P(\boldsymbol{\theta}|\boldsymbol{y})}{Q(\boldsymbol{\theta})}\right] \\ &= \underbrace{\ln P(\boldsymbol{y})}_{\text{Log evidence}} \underbrace{- D_{KL}[Q(\boldsymbol{\theta}) \| P(\boldsymbol{\theta}|\boldsymbol{y})]}_{\text{Approximation error}} \end{aligned} \tag{12}$$

Where $D_{KL}[Q \| P]$ is the KL divergence from density P to Q, which is a (non-negative) measure of difference between the two densities. The log evidence is a fixed (but unknown) quantity, so if we can find a density $Q(\boldsymbol{\theta})$ that maximizes the free energy, then we minimize the divergence between the true posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$ and the approximation $Q(\boldsymbol{\theta})$.

We can also re-write the free energy from Eq. (11) as the difference between the model's accuracy and its complexity:

$$\begin{aligned} F[Q(\theta)] &= E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}|\boldsymbol{\theta}) + \ln P(\boldsymbol{\theta}) - \ln Q(\boldsymbol{\theta})] \\ &= E_{Q(\boldsymbol{\theta})}\left[\ln P(\boldsymbol{y}|\boldsymbol{\theta}) + \ln \frac{P(\boldsymbol{\theta})}{Q(\boldsymbol{\theta})}\right] \\ &= \underbrace{E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}|\boldsymbol{\theta})]}_{\text{Accuracy}} - \underbrace{D_{KL}[Q(\boldsymbol{\theta}) \| P(\boldsymbol{\theta})]}_{\text{Complexity}} \end{aligned} \tag{13}$$

Here, the accuracy term is the expected log likelihood, and the complexity is how far the parameters have diverged from the prior to the approximate posterior. Thus, if we select the model with the most positive free energy out of several candidate models, then we inherently select the model that offers the best trade-off between accuracy and complexity (c.f., Occam's razor). Importantly, this definition of complexity takes into account the covariance amongst parameters, and thus enables the free energy to serve as a better approximation of the log evidence than statistics which discard the covariance – in particular the BIC and AIC (Penny, 2012b).

Finally, we can re-arrange the free energy in Eq. (11) as follows:

$$\begin{aligned} F[Q(\boldsymbol{\theta})] &= E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y},\boldsymbol{\theta})] - E_{Q(\boldsymbol{\theta})}[\ln Q(\boldsymbol{\theta})] \\ &= \underbrace{E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y},\boldsymbol{\theta})]}_{\text{Expected (negative) energy}} + \underbrace{H[Q(\boldsymbol{\theta})]}_{\text{Entropy}} \end{aligned} \tag{14}$$

By analogy with its applications in statistical physics, the first term is referred to as an *energy*, and the latter is the Shannon entropy H of the posterior density $Q(\boldsymbol{\theta})$. A probability density $Q(\boldsymbol{\theta})$ with high entropy will be smooth (or in the limit, flat) over the possible values of $\boldsymbol{\theta}$, whereas a probability density with low entropy will have peaks around particular values. This means that if we have two candidate $Q(\boldsymbol{\theta})$ densities, both equally likely under the priors, to maximize the free energy we would select the one that is smoother or, equivalently, with the higher entropy. This is referred to as Jaynes' Principle of Maximum Entropy (Jaynes, 1957), and means that we select the simplest explanation for the data where possible. (A further consequence of Jaynes' Principle is that as we come closer to maximizing the free energy, the free energy forms a smoother landscape, aiding the performance of optimization algorithms. We will return to the notion of a free energy landscape in Section 5).

4. Free energy under the Laplace approximation

We now build up to the definition of the free energy that is used in the VL scheme, by first defining the probability densities that appear in the definition of the free energy (Eq. (11)) and then by introducing a *mean-field partition* over parameters and hyperparameters.

4.1. Quadratic approximations

In what follows, we will approximate the various unknown probability densities that appear in Eq. (11) using multivariate normal densities, by way of *quadratic approximations* and *Laplace's method*, which we will first reprise. A quadratic approximation, also called a second order Taylor approximation, approximates any (twice differentiable) function $g(\boldsymbol{x})$ that has vector input $\boldsymbol{x}$, close to its peak or mode $\boldsymbol{x} = \boldsymbol{x}_0$, with the linear function:

$$T(\boldsymbol{x}) = \underbrace{g(\boldsymbol{x}_0)}_{\text{Constant}} + \underbrace{\nabla g(\boldsymbol{x}_0)\cdot(\boldsymbol{x}-\boldsymbol{x}_0)}_{\text{Linear term=0}} + \underbrace{\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T H_g(\boldsymbol{x}_0)(\boldsymbol{x}-\boldsymbol{x}_0)}_{\text{Quadratic term}} \tag{15}$$

$$= \underbrace{g(\boldsymbol{x}_0)}_{\text{Constant}} + \underbrace{\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T H_g(\boldsymbol{x}_0)(\boldsymbol{x}-\boldsymbol{x}_0)}_{\text{Quadratic term}} \tag{16}$$

where $\nabla g(\boldsymbol{x}_0)$ is the gradient of function g evaluated at $\boldsymbol{x}_0$ and $H_g(\boldsymbol{x}_0)$ is the Hessian matrix—a matrix of second derivatives—of g evaluated at $\boldsymbol{x}_0$, i.e., $[H_g(\boldsymbol{x}_0)]_{i,j} = \partial_{x_i x_j} g(\boldsymbol{x}_0)$. At the mode of the density, $\boldsymbol{x} = \boldsymbol{x}_0$, the linear term equals zero and thus disappears in Eq. (15). This has a very similar form to the log of a normal density, which is utilized in *Laplace's method* for function approximation.

4.2. Laplace's method

Laplace's method leverages the quadratic approximation in order to approximate a function g near its mode as a (scaled) normal density, as detailed in Fig. 3. If $g(x)$ is a function of scalar variable x, the Laplace approximation $L(x)$ is:

$$\begin{aligned} g(x) &\approx L(x) \propto N(x; \mu, \pi^{-1}) \\ \mu &= x_0 \\ \pi &= -\partial_{x_0 x_0} \ln g(x_0) \end{aligned} \tag{17}$$

where μ is the mean and π is the precision. When $g(\boldsymbol{x})$ is a function of a vector of variables $\boldsymbol{x}$, the Laplace approximation $L(\boldsymbol{x})$ returns a (scaled) multivariate normal density:

[5] For a probability density function $P(x)$, the expected value $E[P(x)]$ is a weighted average over the possible values of x, where each value is weighted by its probability, i.e., $E[P(x)] = \int xP(x)dx$. The expected value can also be taken with respect to another probability density function $Q(x)$, which we write as $E_{Q(x)}[P(x)] = \int Q(x)P(x)dx$. This is a weighted average of $P(x)$, where the weighting comes from $Q(x)$.

$$\begin{aligned} g(\boldsymbol{x}) &\approx L(\boldsymbol{x}) \propto N(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Pi}^{-1}) \\ \boldsymbol{\mu} &= \boldsymbol{x}_0 \\ \boldsymbol{\Pi} &= -\partial_{x_0 x_0} \ln g(\boldsymbol{x}_0) \end{aligned} \tag{18}$$

Where $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\Pi}$ is the precision matrix, i.e., the inverse of the variance-covariance matrix. We will apply this method several times to approximate the different quantities that comprise the free energy.

4.3. Approximating the free energy

Recall Eq. (11), the definition of the free energy: $F[Q(\boldsymbol{\theta})] = E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}, \boldsymbol{\theta}) - \ln Q(\boldsymbol{\theta})]$. We will now write expressions for the log joint $\ln P(\boldsymbol{y}, \boldsymbol{\theta})$ and the approximate posterior (i.e., the choice of $Q(\boldsymbol{\theta})$). As the true form of the posterior is unknown, and the form of the log joint may involve analytically difficult nonlinearities, we will use (local) quadratic or Laplace approximations to define them in a flexible manner.

4.3.1. Approximating $\ln P(\boldsymbol{y}, \theta)$

We will apply a quadratic approximation for the log joint density, which is reasonable because densities tend to look normal close to their mode. We define $\boldsymbol{\mu}$ to be the mode of the joint probability density, $\boldsymbol{\mu} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} P(\boldsymbol{y}, \boldsymbol{\theta})$. Note that this is the same as the mode of the posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$, because the posterior is simply a scaled version of the joint. (Also, $\boldsymbol{\mu}$ is the mode of the log of these densities, as taking the logarithm does not change the mode.) The approximation to the log joint is:

$$\begin{aligned} \ln P(\boldsymbol{y}, \boldsymbol{\theta}) &\approx \ln P(\boldsymbol{y}, \boldsymbol{\mu}) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\mu})^T \left[\partial_{\mu\mu} \ln P(\boldsymbol{y}, \boldsymbol{\mu})\right](\boldsymbol{\theta} - \boldsymbol{\mu}) \\ &= \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu}) \\ \boldsymbol{\Sigma}^{-1} &= -\partial_{\boldsymbol{\mu}\boldsymbol{\mu}} \ln P(\boldsymbol{y}, \boldsymbol{\mu}) = -\partial_{\boldsymbol{\mu}\boldsymbol{\mu}} \ln P(\boldsymbol{\mu}|\boldsymbol{y}) \end{aligned} \tag{19}$$

4.3.2. Choosing an approximate posterior Q

We will similarly apply a quadratic approximation to the log posterior $\ln P(\boldsymbol{\theta}|\boldsymbol{y})$ around the posterior mode $\boldsymbol{\mu}$. Under Laplace's method, this means we will be using a normal density, $Q(\boldsymbol{\theta})$, in order to approximate the posterior $P(\boldsymbol{\theta}|\boldsymbol{y})$:

$$\begin{aligned} \ln P(\boldsymbol{\theta}|\boldsymbol{y}) &\approx \ln P(\boldsymbol{\mu}|\boldsymbol{y}) - \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu}) \\ \Rightarrow P(\boldsymbol{\theta}|\boldsymbol{y}) &\approx Q(\boldsymbol{\theta}) = N(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \end{aligned} \tag{20}$$

4.3.3. Taking expectations

Next, we will apply the expectation operator[6] from Eq. (11) to the previous two densities under the approximate posterior:

$$E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}, \boldsymbol{\theta})] \approx \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{1}{2} E_{Q(\boldsymbol{\theta})}\left[(\boldsymbol{\theta} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu})\right] \tag{21}$$

$$E_{Q(\boldsymbol{\theta})}[\ln Q(\boldsymbol{\theta})] \approx -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi\right] - \frac{1}{2} E_{Q(\theta)}\left[(\boldsymbol{\theta} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\theta} - \boldsymbol{\mu})\right] \tag{22}$$

Where Eq. (22) uses the definition of the log of the multivariate normal density for a variable with dimension n.

To simplify these expressions, note that each quadratic term inside the square brackets is a scalar. This means we can use the 'trace trick', $tr(\boldsymbol{ABC}) = tr(\boldsymbol{CAB})$. Applying this gives the simpler expressions:

$$\begin{aligned} E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}, \boldsymbol{\theta})] &\approx \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{1}{2} E_{Q(\boldsymbol{\theta})}\left[tr\left(\underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})^T}_{A} \underbrace{\boldsymbol{\Sigma}^{-1}}_{B} \underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})}_{C}\right)\right] \\ &= \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{1}{2} tr\left(E_{Q(\boldsymbol{\theta})}\left[\underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})}_{C} \underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})^T}_{A}\right] \underbrace{\boldsymbol{\Sigma}^{-1}}_{B}\right) \\ &= \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{1}{2} tr\left(\boldsymbol{\Sigma}\, \boldsymbol{\Sigma}^{-1}\right) \\ &= \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{n}{2} \end{aligned} \tag{23}$$

$$\begin{aligned} E_{Q(\boldsymbol{\theta})}[\ln Q(\boldsymbol{\theta})] &= -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi\right] - \frac{1}{2} E_{Q(\theta)}\left[\underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})^T}_{A} \underbrace{\boldsymbol{\Sigma}^{-1}}_{B} \underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})}_{C}\right] \\ &= -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi\right] - \frac{1}{2} tr\left(E_{Q(\theta)}\left[\underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})}_{C} \underbrace{(\boldsymbol{\theta} - \boldsymbol{\mu})^T}_{A}\right] \underbrace{\boldsymbol{\Sigma}^{-1}}_{B}\right) \\ &= -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi\right] - \frac{1}{2} tr\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1}\right) \\ &= -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi\right] - \frac{n}{2} \\ &= -\frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi e\right] \end{aligned} \tag{24}$$

Substituting these expectations into Eq. (11), we get the free energy under the Laplace approximation:

$$\begin{aligned} F[Q(\boldsymbol{\theta})] &= E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y}, \boldsymbol{\theta}) - \ln Q(\boldsymbol{\theta})] \\ &= \ln P(\boldsymbol{y}, \boldsymbol{\mu}) - \frac{n}{2} + \frac{1}{2}\left[\ln(|\boldsymbol{\Sigma}|) + n \ln 2\pi e\right] \end{aligned} \tag{25}$$

Where $\boldsymbol{\Sigma}$ is the posterior covariance and n is the total number of parameters.

4.4. Free energy under a mean-field approximation

The previous section assumed that all parameters are treated equally. However, it is often helpful to separate them out into two (or more) types. The estimation scheme outlined here alternates between updating the estimate of the model's parameters $\boldsymbol{\beta}$ and the hyperparameters $\boldsymbol{\lambda}$ that control the precision of the observation noise. We previously lumped these two kinds of parameter together as $\boldsymbol{\theta}$ but now separate them out. We can re-write the joint probability as follows:

$$P(\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{\lambda}) = P(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\lambda}) P(\boldsymbol{\beta}) P(\boldsymbol{\lambda}) \tag{26}$$

with normal densities for the likelihood and priors Eqs. (7) and (8). The approximate posterior is chosen to factorise as follows:

$$Q(\boldsymbol{\theta}) = Q(\boldsymbol{\beta}, \boldsymbol{\lambda}) = Q(\boldsymbol{\beta}) Q(\boldsymbol{\lambda}) \tag{27}$$

This factorisation is known as a mean-field approximation. Each approximate posterior is a multivariate normal density:

$$\begin{aligned} Q(\boldsymbol{\beta}) &= N\left(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right) \\ Q(\boldsymbol{\lambda}) &= N(\boldsymbol{\mu}_{\boldsymbol{\lambda}}, \boldsymbol{\Sigma}_{\boldsymbol{\lambda}}) \end{aligned} \tag{28}$$

The free energy is easily extended for this factorisation (from Eq. (25)):

[6] Note that this step ensures the final result is the same regardless of whether $\boldsymbol{\mu}$ is the mode of $P(\boldsymbol{\theta}|\boldsymbol{y})$ (prior to optimisation of the approximate posterior, it may not be). The linear term in the expansion will still disappear under the expectation as it includes a factor of $E_{Q(\theta)}[\boldsymbol{\theta} - \boldsymbol{\mu}] = 0$, recovering Eq. 21.

$$\begin{aligned} F[Q(\boldsymbol{\theta})] &= \underbrace{\ln P\left(\boldsymbol{y}, \boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\mu}_{\boldsymbol{\lambda}}\right) - \frac{1}{2}(p+h)}_{E_{Q(\boldsymbol{\theta})}[\ln P(\boldsymbol{y},\boldsymbol{\theta})]} \\ &\quad \underbrace{+\frac{1}{2}\left[\ln\left(\left|\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right|\right) + \ln(|\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}|) + (p+h)\ln 2\pi e\right]}_{E_{Q(\boldsymbol{\theta})}[\ln Q(\boldsymbol{\theta})]} \\ &= \ln P\left(\boldsymbol{y}, \boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\mu}_{\boldsymbol{\lambda}}\right) + \frac{1}{2}\left[\ln\left(\left|\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right|\right) + \ln(|\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}|) + (p+h)\ln 2\pi\right] \end{aligned} \tag{29}$$

Where p is the number of parameters and h is the number of hyperparameters.

So, we now just need to define the log joint, which is the product of three normal densities in Eq. (26). The log of a normal density with mean $\boldsymbol{m}$ covariance matrix $\boldsymbol{S}$ for a variable of dimension k is defined as:

$$\begin{aligned} \ln N(\boldsymbol{x}; \boldsymbol{m}, \mathbf{S}) &= -\frac{1}{2}\left[\ln|\mathbf{S}| + (\boldsymbol{x}-\boldsymbol{m})^T\mathbf{S}^{-1}(\boldsymbol{x}-\boldsymbol{m}) + k\ln(2\pi)\right] \\ &= -\frac{1}{2}\left[\ln|\boldsymbol{S}| + \epsilon_{\boldsymbol{x}}^T\mathbf{S}^{-1}\epsilon_{\boldsymbol{x}} + k\ln(2\pi)\right] \end{aligned} \tag{30}$$

Where the *error term* is $\epsilon_{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{\mu}$. Substituting into Eq. (29) (with $\dim(\boldsymbol{y}) = k$):

$$\begin{aligned} F[Q(\boldsymbol{\theta})] &= \ln P(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\lambda}) + \ln P(\boldsymbol{\beta}) + \ln P(\boldsymbol{\lambda}) + \frac{1}{2}\left[\ln\left(\left|\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right|\right) + \ln(|\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}|) + (p+h)\ln 2\pi\right] \\ &= \underbrace{-\frac{1}{2}\left[\ln\left|\boldsymbol{\Pi}_{\boldsymbol{y}}^{-1}\right| + \epsilon_{\boldsymbol{y}}^T\boldsymbol{\Pi}_{\boldsymbol{y}}\epsilon_{\boldsymbol{y}}\right]}_{\text{Likelihood}} \\ &\quad \underbrace{-\frac{1}{2}\left[\ln\left|\boldsymbol{\Pi}_{\boldsymbol{\beta}}^{-1}\right| + \epsilon_{\boldsymbol{\beta}}^T\boldsymbol{\Pi}_{\boldsymbol{\beta}}\epsilon_{\boldsymbol{\beta}}\right]}_{\text{Prior (parameters)}} \\ &\quad \underbrace{-\frac{1}{2}\left[\ln\left|\boldsymbol{\Pi}_{\boldsymbol{\lambda}}^{-1}\right| + \epsilon_{\boldsymbol{\lambda}}^T\boldsymbol{\Pi}_{\boldsymbol{\lambda}}\epsilon_{\boldsymbol{\lambda}}\right]}_{\text{Prior (hyperparameters)}} \\ &\quad \underbrace{+\frac{1}{2}\left[\ln\left(\left|\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right|\right) + \ln(|\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}|)\right]}_{\text{Posterior entropy}} \\ &\quad \underbrace{-\frac{1}{2}k\ln(2\pi)}_{\text{Constants}} \end{aligned} \tag{31}$$

With error terms:

$$\begin{aligned} \boldsymbol{\epsilon}_y &= \boldsymbol{y} - \boldsymbol{g}\left(\boldsymbol{\mu}_\beta\right) \\ \boldsymbol{\epsilon}_\beta &= \boldsymbol{\mu}_\beta - \boldsymbol{\eta}_\beta \\ \boldsymbol{\epsilon}_\lambda &= \boldsymbol{\mu}_\lambda - \boldsymbol{\eta}_\lambda \end{aligned} \tag{32}$$

Where, to recap, $\boldsymbol{\Pi}_{\boldsymbol{\beta}}$, $\boldsymbol{\Pi}_{\boldsymbol{\lambda}}$ are prior precisions, $\boldsymbol{\eta}_\beta$, $\boldsymbol{\eta}_\lambda$ are prior expectations, $\boldsymbol{\Sigma}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$ are posterior covariances, $\boldsymbol{\mu}_\beta$, $\boldsymbol{\mu}_\lambda$ are posterior expectations and $\boldsymbol{\Pi}_{\mathbf{y}}$ is the (modelled) precision of the data. We then use Eq. (31) as the objective function (i.e., the measure of the model's quality), which we seek to maximize in order to approximate the log evidence and identify the model's parameters.

5. Estimation scheme

Next, we derive an algorithm for finding the posterior parameter density $Q(\boldsymbol{\theta})$ that maximizes the free energy $F[Q(\boldsymbol{\theta})]$. Full derivations of these equations are provided in the supplementary material.

5.1. Overview of gradient ascent and Gauss-Newton

The simplest approach to numerically maximizing the free energy is *gradient ascent*. Conceptually, the free energy forms a landscape, the dimensions of which are the parameters. Gradient ascent or descent takes small steps in the same direction as the gradient, as if climbing or descending a hill. Writing the function to be optimized generically as $f(\boldsymbol{\mu})$, the parameters $\boldsymbol{\mu}$ are updated on each iteration according to $\boldsymbol{\mu} = \boldsymbol{\mu} + \boldsymbol{\Delta\mu}$, where:

$$\boldsymbol{\Delta\mu} = \alpha\nabla_{\boldsymbol{\mu}}[f(\boldsymbol{\mu})] \tag{33}$$

and α is the step size (positive for ascent, and negative for descent). This is illustrated in Fig. 4A, for finding the minimum of a function using gradient descent. From an initial guess (labelled 1), small steps are taken towards a minimum, which in this example happens to be the global minimum (shaded green sphere).

Gradient ascent or descent is rarely sufficient in practice. One issue regards its speed: because the update is proportional to the gradient, estimates advance very slowly in shallow regions of the landscape (as can be seen in Fig. 4A). Conversely, in very steep regions, the estimates advance quickly – running the risk of taking too large a step and moving away from an optimum. We would ideally like the opposite situation – to move quickly in shallow regions, and slowly in steeper regions.

The Gauss-Newton algorithm takes a different approach and can converge far more quickly. To minimize a function, a parabola (or U-shaped plane) is fitted to $f(\boldsymbol{\mu})$ at the current estimate of the parameters (Fig. 4B). The algorithm then jumps to the minimum of the parabola, which becomes the new parameter estimate (labelled 2 in Fig. 4B) and the process repeats. This can enable fast progression from the initial estimate to the global or local optimum (Fig. 4C). For some vector of parameters $\boldsymbol{\mu}^*$, the parabola is a second-order quadratic approximation of $f(\boldsymbol{\mu})$ at the current estimate of the parameters $\boldsymbol{\mu}$:

$$f(\boldsymbol{\mu}^*) \approx f(\boldsymbol{\mu}) + \nabla_{\boldsymbol{\mu}} f(\boldsymbol{\mu})\cdot(\boldsymbol{\mu}^* - \boldsymbol{\mu}) + \frac{1}{2}(\boldsymbol{\mu}^* - \boldsymbol{\mu})\boldsymbol{H}_f(\boldsymbol{\mu})(\boldsymbol{\mu}^* - \boldsymbol{\mu}) \tag{34}$$

Where $\boldsymbol{H}_f$ is the Hessian matrix of second derivatives. This approximation has a unique minimum at $\boldsymbol{\mu} - \boldsymbol{H}_f(\boldsymbol{\mu})^{-1}\cdot\nabla_{\boldsymbol{\mu}} f(\boldsymbol{\mu})$, which becomes the new estimate of the parameters, giving rise to the update equation:

$$\boldsymbol{\Delta\mu} = -\boldsymbol{H}_f(\boldsymbol{\mu})^{-1}\cdot\nabla_{\boldsymbol{\mu}}[f(\boldsymbol{\mu})] \tag{35}$$

While Gauss-Newton is much faster than gradient ascent, it will perform poorly if the quadratic approximation is poor – a particular risk

Laplace approximation of a gamma distribution G

Pierre-Simon Laplace
1749-1827

1. Define the log of the function to be approximated:

$$f(x;a,b) = \ln G(x;a,b)$$

2. Find its mode (peak) x_0

$$x_0 = (a-1)/b$$

3. Approximate $f(x)$ at $x = x_0$ using Taylor approximation $T(x)$:

$$f(x;a,b) \approx T(x) = f(x_0) + \frac{1}{2}\partial_{x_0x_0} f(x_0)(x-x_0)^2$$

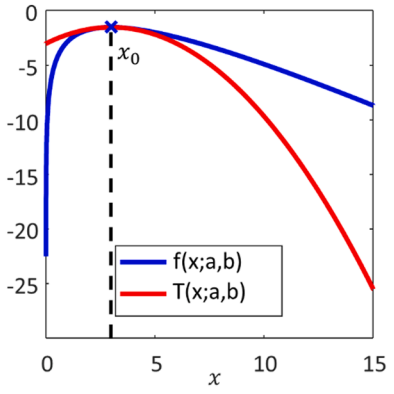
4. Take the exponential to approximate the underlying function:

$$L(x) = \exp[T(x;a,b)] = G(x_0;a,b)\exp\left[\frac{1}{2}\partial_{x_0x_0} f(x_0)(x-x_0)^2\right]$$

This is a (scaled) normal distribution with mean $\mu = x_0$ and precision $\pi = -\partial_{x_0x_0} f(x_0)$:

$$L(x) = N(x;\mu,\pi^{-1}) \cdot G(x_0;a,b) \cdot \sigma\sqrt{2\pi}$$

Log gamma f and Taylor approximation T



Gamma G and Laplace approximation L

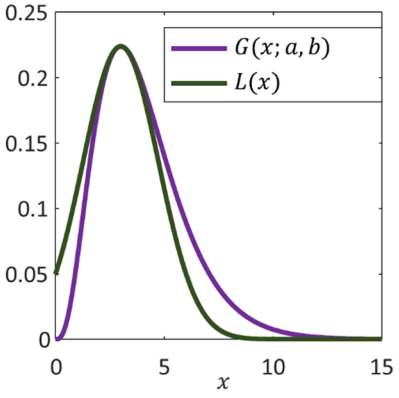


Fig. 3. Laplace approximation of a non-normal density. This example illustrates Laplace's method by approximating a gamma probability density function $G(x;a,b)$ over the variable x with shape parameter a and inverse scale parameter b. The result is a scaled normal density $N(x;\mu,\pi^{-1})$ with mean $\mu = x_0$ and precision $\pi = -\partial_{x_0x_0}\ln G(x_0)$, where x_0 is the mode of $\ln G$. After normalisation to ensure it is a proper probability density, the scaling becomes irrelevant, and we end up with a normal density. Portrait by James Posselwhite, in the public domain via *Wikipedia*, https://en.wikipedia.org/wiki/Pierre-Simon_Laplace.

when taking large steps that are far from the current estimate. In these situations, it would be ideal to dynamically switch to a gradient ascent or descent. A continuous transition between gradient ascent and Gauss-Newton like behaviour is achieved by the variational Laplace scheme, as we will return to shortly. As a first step towards this, we will derive a gradient ascent algorithm for the free energy.

5.2. Gradient ascent on free energy

The mean-field approximation $Q(\boldsymbol{\theta}) = Q(\boldsymbol{\beta})Q(\boldsymbol{\lambda})$ naturally gives rise to a gradient ascent algorithm that alternates between optimising the parameters and hyperparameters. With the mean-field approximation, the free energy (Eq. (11)) can be extended to:

$$F[Q(\boldsymbol{\theta})] = E_{Q(\boldsymbol{\beta})Q(\boldsymbol{\lambda})}\left[\ln\frac{P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})}{Q(\boldsymbol{\beta})Q(\boldsymbol{\lambda})}\right] \tag{36}$$

We will use $\delta_{Q(\boldsymbol{\beta})}F$ and $\delta_{Q(\boldsymbol{\lambda})}F$ to denote the *variation* of the free energy with respect to each factor of the approximate posterior, $Q(\boldsymbol{\beta})$ and $Q(\boldsymbol{\lambda})$, respectively. These are functional derivatives, encoding the rate of change in F that would result from infinitesimal adjustments to the form of each function—where the functions in question here are probability densities. Ignoring constants:

$$\delta_{Q(\boldsymbol{\beta})}F = -\ln Q(\boldsymbol{\beta}) + E_{Q(\boldsymbol{\lambda})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})]$$

$$\delta_{Q(\boldsymbol{\lambda})}F = -\ln Q(\boldsymbol{\lambda}) + E_{Q(\boldsymbol{\beta})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})] \tag{37}$$

Setting to zero and taking the exponential, we get the optimal approximate posteriors on each iteration of the algorithm:

$$\delta_{Q(\boldsymbol{\beta})}F = 0 \Leftrightarrow Q(\boldsymbol{\beta}) \propto \exp\left(E_{Q(\boldsymbol{\lambda})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})]\right)$$

$$\delta_{Q(\boldsymbol{\lambda})}F = 0 \Leftrightarrow Q(\boldsymbol{\lambda}) \propto \exp\left(E_{Q(\boldsymbol{\beta})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})]\right) \tag{38}$$

As before (Eq. (23)), quadratic approximations can be used for the terms inside the expectations:

$$E_{Q(\boldsymbol{\beta})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})] \approx \ln P(\boldsymbol{y},\boldsymbol{\mu}_\beta,\boldsymbol{\lambda}) + \frac{1}{2}tr\left(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\,\partial_{\mu_\beta\mu_\beta}\ln P(\boldsymbol{y},\boldsymbol{\mu}_\beta,\boldsymbol{\lambda})\right)$$

$$E_{Q(\boldsymbol{\lambda})}[\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\lambda})] \approx \ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\mu}_\lambda) + \frac{1}{2}tr\left(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}\,\partial_{\mu_\lambda\mu_\lambda}\ln P(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\mu}_\lambda)\right) \tag{39}$$

By definition, the modes of the approximate posterior densities over parameters $\boldsymbol{\mu}_\beta$ and hyperparameters $\boldsymbol{\mu}_\lambda$ must maximise the above quantities (i.e., the logarithms of the approximate posteriors), giving the following pair of equations:

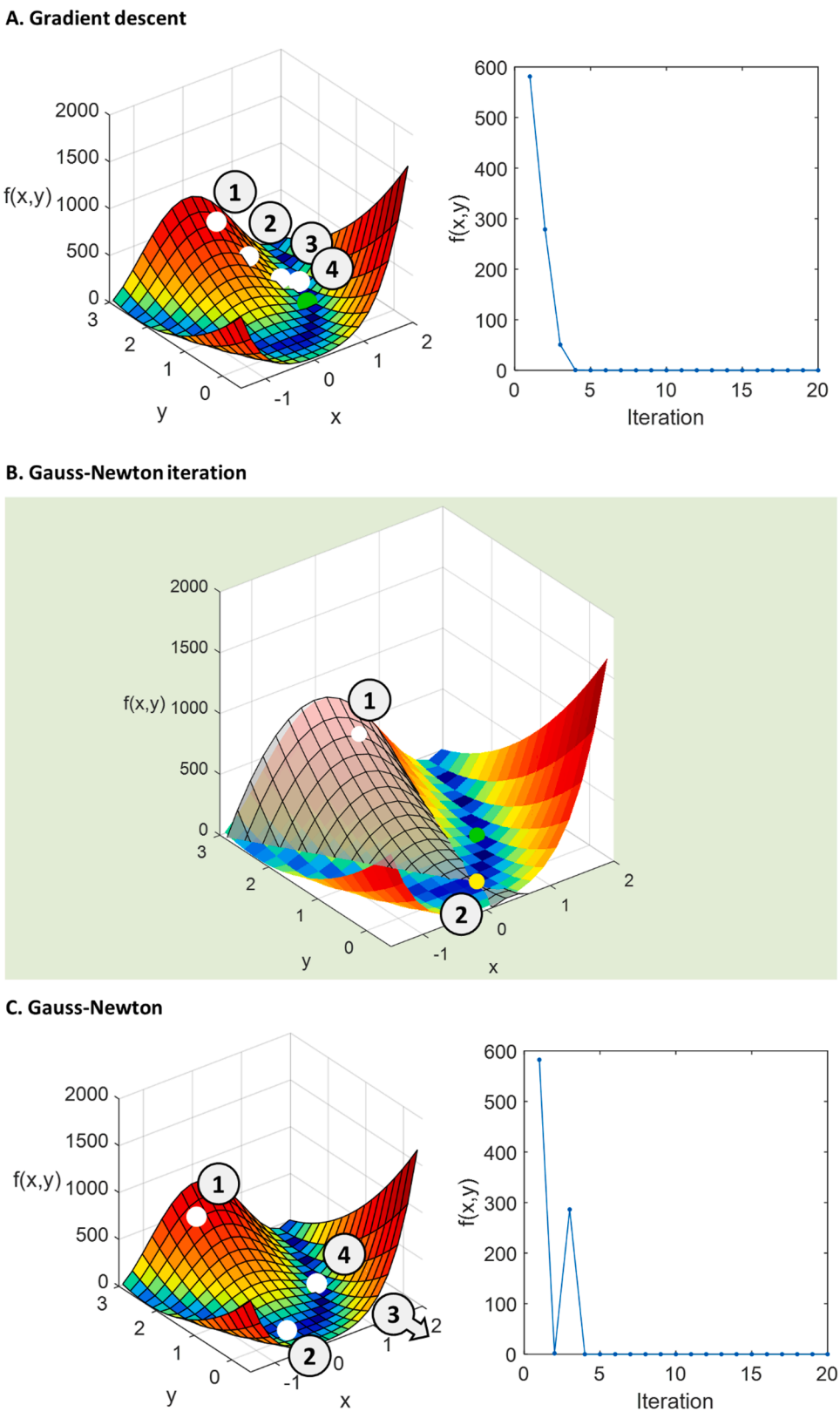


Fig. 4. Comparison of Gradient ascent and Gauss-Newton optimization. The left and centre panels illustrate an example function with two inputs $f(x,y)$ (the Rosenbrock function, with hyperparameters $a = 1$, $b = 100$). The global minimum is at $x = 1$, $y = 1$, indicated with a shaded green sphere in panel A. White spheres are estimates of the parameters (x,y) over successive iterations, indexed by the numbers in circles. The right panels show the evaluation of the function (i.e., the height of the surface) over successive iterations.

$$
\begin{aligned}
\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}} &= \nabla_{\mu_\beta} E_{Q(\lambda)}\left[\ln P\left(\mathbf{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\lambda}\right)\right] \\
&= \nabla_{\mu_\beta} \ln P\left(\mathbf{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right) + \nabla_{\mu_\beta}\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}\,\partial_{\mu_\lambda\mu_\lambda}\ln P\left(\mathbf{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right)\right) \\
&= \underbrace{\mathbf{J}_{\mathbf{g}}^{\mathrm{T}}\boldsymbol{\Pi}_{\mathbf{y}}\boldsymbol{\epsilon}_{\mathbf{y}}}_{\textit{Likelihood}} - \underbrace{\boldsymbol{\Pi}_{\boldsymbol{\beta}}\boldsymbol{\epsilon}_{\boldsymbol{\beta}}}_{\textit{Prior}} + \underbrace{\textstyle\sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{h}}\left[\left(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}\right)_{\mathbf{jj}}\mathbf{J}_{\mathbf{g}}^{\mathrm{T}}\mathbf{P}_{\mathbf{j}}\boldsymbol{\epsilon}_{\mathbf{y}}\right]}_{\textit{Hyperparameters }\boldsymbol{\lambda}} \\
\left(\Delta\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right)_{\mathrm{i}} &= \partial_{\mu_{\lambda_{\mathrm{i}}}}\mathrm{E}_{Q(\beta)}\left[\ln P\left(\mathbf{y},\boldsymbol{\beta},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right)\right] \\
&= \partial_{\mu_{\lambda_{\mathrm{i}}}}\ln P\left(\mathbf{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right) + \partial_{\mu_{\lambda_{\mathrm{i}}}}\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\partial_{\mu_\beta\mu_\beta}\ln P\left(\mathbf{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right)\right) \\
&= \underbrace{\frac{1}{2}\mathrm{tr}\left(\mathbf{P}_{\mathbf{i}}\boldsymbol{\Pi}_{\mathbf{y}}^{-\mathbf{1}}\right) - \frac{1}{2}\boldsymbol{\epsilon}_{\mathbf{y}}^{\mathrm{T}}\,\mathbf{P}_{\mathbf{i}}\boldsymbol{\epsilon}_{\mathbf{y}}}_{\textit{Likelihood}} - \underbrace{\partial_{\mu_{\lambda_{\mathrm{i}}}}\left(\boldsymbol{\epsilon}_{\boldsymbol{\lambda}}\right)^{\mathrm{T}}\boldsymbol{\Pi}_{\boldsymbol{\lambda}}\boldsymbol{\epsilon}_{\boldsymbol{\lambda}}}_{\text{Prior}} - \underbrace{\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\mathbf{J}_{\mathbf{g}}^{\mathrm{T}}\,\mathbf{P}_{\mathbf{i}}\mathbf{J}_{\mathbf{g}}\right)}_{\textit{Parameters }\boldsymbol{\beta}}
\end{aligned}
\tag{40}
$$

Where vector $\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}}$ is the change in the parameters at each iteration of the algorithm, $(\Delta\boldsymbol{\mu}_{\boldsymbol{\lambda}})_i$ is the change in the i-th hyperparameter at each iteration, $\nabla_{\boldsymbol{\mu}_{\boldsymbol{\beta}}}[\cdot]$ is the gradient with respect to the parameters and $\boldsymbol{J}_g$ is the Jacobian matrix[7] of first partial derivatives with $(\mathbf{J}_g)_{ij} = \partial_{\mu_j} g_i(\boldsymbol{\mu}_{\boldsymbol{\beta}})$ for observation i and parameter j. For hyperparameter i, the derivative term $\partial_{\mu_{\lambda_i}}(\epsilon_{\boldsymbol{\lambda}})$ is a vector with value of unity in index i and zero elsewhere, and $\boldsymbol{P_i} = \partial_{\mu_{\lambda_i}}(\boldsymbol{\Pi}_{\boldsymbol{y}}) = \exp(\lambda_i)\boldsymbol{\Pi_i}$. Terms including the second derivative of $g(\boldsymbol{\theta})$ have been omitted on the assumption that it is locally approximately linear. Similarly, the final term for the parameters is usually ignored, because its contribution is usually trivial in relation to the other terms, and is zero when the precision is linear in the hyperparameters.

These updates depend on first calculating the covariance of the parameters $\boldsymbol{\Sigma}_{\boldsymbol{\beta}}$ and hyperparameters $\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$, which are given by the expressions[8]:

$$
\begin{aligned}
\left(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\right)^{-1} &= -\partial_{\mu_\beta\mu_\beta}E_{Q(\lambda)}\left[\ln P\left(\boldsymbol{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\lambda}\right)\right] \\
&\approx \boldsymbol{J}_{\boldsymbol{g}}^{T}\boldsymbol{\Pi}_{\boldsymbol{y}}\boldsymbol{J}_{\boldsymbol{g}} + \boldsymbol{\Pi}_{\boldsymbol{\beta}} \\
\left[\left(\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}\right)^{-1}\right]_{i,i} &= -\partial_{\mu_{\lambda_i}\mu_{\lambda_i}}E_{Q(\beta)}\left[\ln P\left(\boldsymbol{y},\boldsymbol{\beta},\boldsymbol{\mu}_{\boldsymbol{\lambda}}\right)\right] \\
&= \left(\boldsymbol{\Pi}_{\boldsymbol{\lambda}}\right)_{ii} - \frac{1}{2}tr\left(\boldsymbol{P}_{\boldsymbol{i}}\boldsymbol{\Sigma}_{\boldsymbol{y}} - \boldsymbol{P}_{\boldsymbol{i}}\boldsymbol{\Sigma}_{\boldsymbol{y}}\boldsymbol{P}_{\boldsymbol{i}}\boldsymbol{\Sigma}_{\boldsymbol{y}}\right) + \frac{1}{2}\boldsymbol{\epsilon}_{\boldsymbol{y}}^{T}\boldsymbol{P}_{\boldsymbol{i}}\boldsymbol{\epsilon}_{\boldsymbol{y}} + \frac{1}{2}tr\left(\boldsymbol{\Sigma}_{\boldsymbol{\beta}}\boldsymbol{J}_{\boldsymbol{g}}^{T}\boldsymbol{P}_{\boldsymbol{i}}\boldsymbol{J}_{\boldsymbol{g}}\right)
\end{aligned}
\tag{41}
$$

Where $\boldsymbol{\Sigma}_{\boldsymbol{y}} = \boldsymbol{\Pi}_{\boldsymbol{y}}^{-1}$. The gradient ascent proceeds by alternately applying the two updates in Eq. (40). However, as mentioned above, a gradient ascent is rarely sufficient for robust performance, motivating an algorithm that dynamically switches between gradient ascent and Gauss-Newton-like updates. This is set out in the next section.

5.3. Updates in continuous time

Parameter updates in optimization are generally treated as occurring in discrete steps, $\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}}$ and $\Delta\boldsymbol{\mu}_{\boldsymbol{\lambda}}$, as described above. However, Friston et al. (2007) considered updates occurring in continuous time, which provides a principled way to transition between gradient ascent and Gauss-Newton-like updates. In continuous time, the gradient ascent on the parameters can then be written in terms of the time derivative $\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}(t)$ at time t:

$$
\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}(t) = \nabla_{\boldsymbol{\mu}_{\boldsymbol{\beta}}}\left[I_\beta\left(\boldsymbol{\mu}_{\boldsymbol{\beta}}[t]\right)\right] \tag{42}
$$

Where $I_\beta(\boldsymbol{\mu}_{\boldsymbol{\beta}}) = E_{Q(\lambda)}[\ln P(\boldsymbol{y},\boldsymbol{\mu}_{\boldsymbol{\beta}},\boldsymbol{\lambda})]$. Integrating Eq. (42) over time using *local linearization* (Ozaki, 1985), the update for a time interval t is:

$$
\begin{aligned}
\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}} &= \left(\exp\left[t\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}\right] - \boldsymbol{I}\right)\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^{-1}\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}} \\
\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}} &= \nabla_{\boldsymbol{\mu}_{\boldsymbol{\beta}}}\left[\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}\right] = \partial_{\mu_\beta\mu_\beta}I_\beta\left(\boldsymbol{\mu}_{\boldsymbol{\beta}}\right)
\end{aligned}
\tag{43}
$$

And a similar expression is applied to update the hyperparameters. When t is small and positive, the algorithm behaves like a gradient ascent, because the term $(\exp[t\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}] - \boldsymbol{I})$ in Eq. (43) regularizes the update (i.e., reduces its size). This follows because the second derivatives $\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$ are negative. As t increases, the matrix exponential term $\exp[t\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}]$ reduces to zero, resulting in a standard Gauss-Newton update (by around $t = 2$):

$$
\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}} = -\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^{-1}\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}} \tag{44}
$$

The integration time t can therefore be varied dynamically to adjust the behaviour of the algorithm. When the algorithm begins, the time is set to a small value, causing it to behave like a gradient scheme for stability. If the free energy has increased, i.e., improved as a result of an update, then regularization is decreased (by increasing t). Conversely, if the free energy has decreased, i.e., worsened, then regularization is increased (by decreasing t).

In practice, the step size t is generally specified with a (log) descent parameter v that is automatically scaled by the average curvature of the landscape α:

$$
\begin{aligned}
t &= \frac{\exp(v)}{\alpha} \\
\alpha &= \exp\left[\frac{Re\left(\ln\left|\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}\right|\right)}{n}\right]
\end{aligned}
\tag{45}
$$

Here, n is the number of parameters. A cautious, slow descent corresponds to $v = -4$ (the default starting value). As the average real eigenvalue of $\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$ increases (i.e., the curvature increases), the regularisation is increased by decreasing v. The algorithm stops when the change in free energy becomes sufficiently small. In more detail, the predicted value of $I_\beta(\boldsymbol{\mu}_{\boldsymbol{\beta}})$ after making the step $\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}}$ is given by: $\nabla_{\boldsymbol{\mu}_{\boldsymbol{\beta}}}[I_\beta(\boldsymbol{\mu}_{\boldsymbol{\beta}})]\cdot\Delta\boldsymbol{\mu}_{\boldsymbol{\beta}}$. If this is small over a series of iterations, then the algorithm is considered to have converged. Pseudocode for the complete VL algorithm is provided in Appendix 2, and MATLAB code accompanies this article.

5.4. Interim summary

This section described an algorithm that searches for a probability density over the parameters $Q(\boldsymbol{\theta})$ that maximizes the free energy. The algorithm ascends the free energy landscape, with an adaptive step size. Readers experienced with machine learning may note some similarity with the Levenberg-Marquardt algorithm, however the continuous time approach reviewed here is derived from first principles without requiring the introduction of an arbitrary regularization term: see Friston et al. (2007) for a detailed comparison.

6. Examples

This section presents worked examples using simulated data, where the same algorithm introduced above is applied to different kinds of model. MATLAB code for each example is provided with this paper.

6.1. Static models

We start with a simple linear regression model with parameters β:

$$
\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{46}
$$

[7] In the canonical implementation of the VL scheme (spm_nlsi_gn), the Jacobian is the negative of that defined here. For this reason, the sign of various terms is switched here compared to the original implementation.

[8] An important point here is that $\boldsymbol{\Sigma}$ is treated as constant with respect to the posterior expectation (seemingly contradicting the idea that the former is an analytic function of the latter). This is consistent with the Laplace approximation, as the assumption that the log joint is approximately quadratic implies that its curvature (and therefore the posterior covariance) is constant.

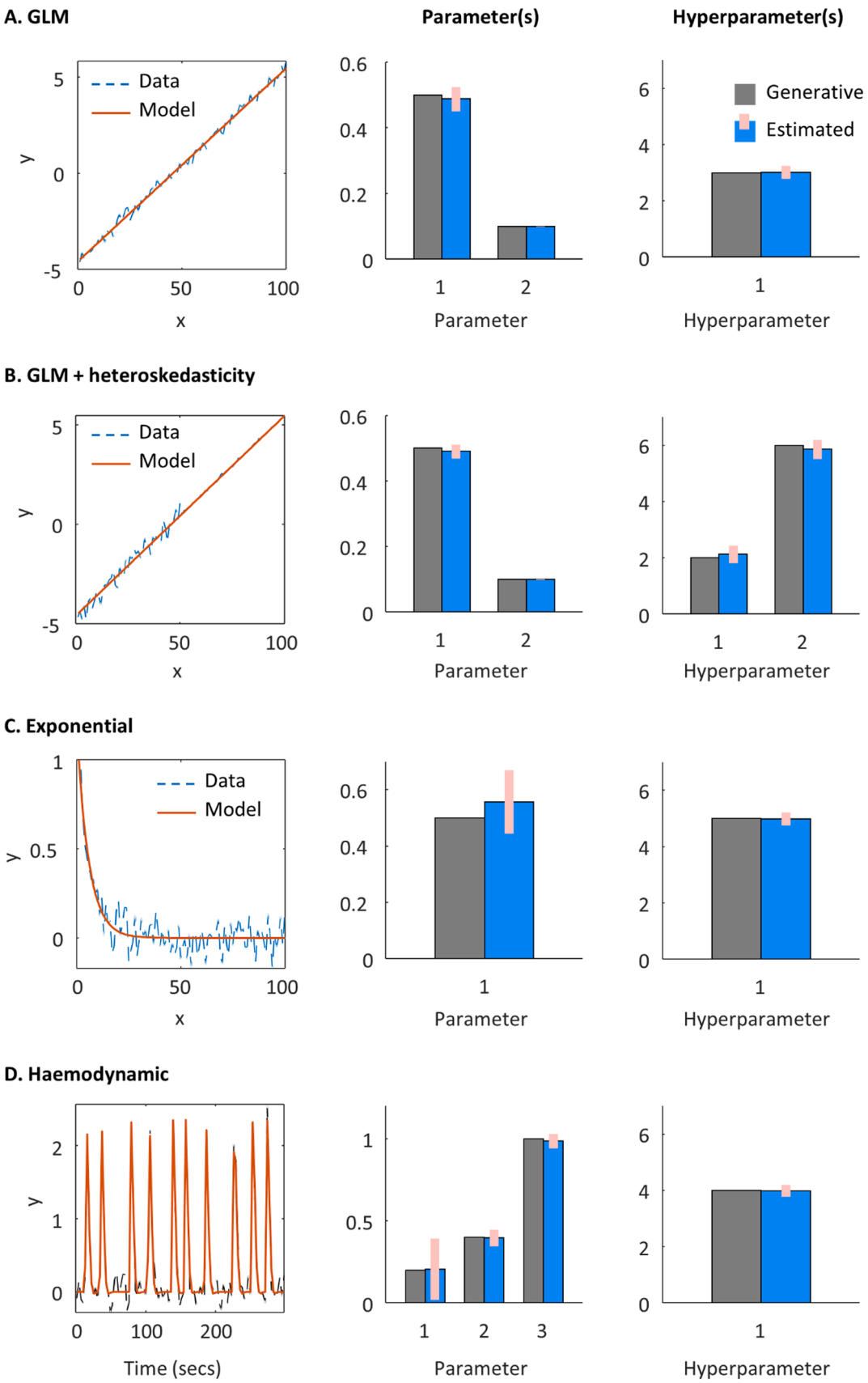


Fig. 5. Example applications of the VL scheme. A-D illustrate the VL scheme applied to different models described in the text. **Left panels**: The simulated data (dashed blue lines) and prediction from the model (solid red lines). **Middle and right panels:** Parameter or hyperparameter values used to generate the data (grey bars without error bars) and posterior estimated values (blue bars with error bars).

For this example, the design matrix $\boldsymbol{X}$ has two columns: the first is a column of ones and the second consists of 100 evenly spaced values between −50 and 50. Thus, the parameter β_1 encodes the mean and β_2 encodes the regression slope. The observation noise is encoded by a single precision component, controlled by log-precision parameter λ:

$$\boldsymbol{\epsilon} \sim \boldsymbol{N}\left(0, \boldsymbol{\Pi}_y^{-1}\right)$$

$$\boldsymbol{\Pi}_y = \exp(\lambda)\boldsymbol{I_n} \tag{47}$$

Note that a truly linear model would converge within a single iteration of the VL scheme, whereas here there is an exponential function in Eq. (47), in order to ensure positivity of the hyperparameter λ. This introduces non-linearity into the model, thereby requiring several iterations to converge. Fig. 5A shows that the parameters and hyperparameters used to generate the simulated data (grey bars) were successfully recovered (blue bars with 90% credible intervals). Note that the covariance amongst (hyper)parameters was also calculated by the VL scheme, but is not shown.

The second example considers the situation where there is *heteroskedasticity* – observations having different levels of observation noise. This is a common situation, for example when parts of the data come from different measurement channels. This can be modelled by having two precision components:

$$\boldsymbol{\Pi}_y = \exp(\lambda_1)\boldsymbol{Q}_1 + \exp(\lambda_2)\boldsymbol{Q}_2 \tag{48}$$

Here, the first precision component matrix, $\boldsymbol{Q}_1$ has values of one on the leading diagonal for the first half of the observations, and zeros for the second half. It therefore captures the precision of the first half of the observations. Similarly, $\boldsymbol{Q}_2$ captures the precision of the second half of the observations. As shown in Fig. 5B, the parameters and hyperparameters used to generate the data were correctly recovered. Next, we illustrate dynamic models, i.e., those modelling continuous changes over time.

6.2. Dynamic models

A typical application of the VL algorithm is to estimate the parameters of dynamic models that are specified as ordinary differential equations (ODEs). As introduced in the empirical example in Section 1.1, this is accomplished by splitting the model into two parts: a model f of the dynamics of the unobserved (latent) variables $\boldsymbol{x}$, and a static model g that translates the latent variables into observations $\boldsymbol{y}$:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{\beta})$$

$$\boldsymbol{y}(t) = g(\boldsymbol{x}(t)) + \boldsymbol{\epsilon} \tag{49}$$

Where $\boldsymbol{x}(t)$ is approximated using a standard numerical integration scheme, i.e.

$$\boldsymbol{x}(t) = \int_0^t \dot{\boldsymbol{x}}(t)\, dt \tag{50}$$

Here, we perform this integration using the local linearization approach of Ozaki (1985). A simple example is estimating the rate of an exponential decay of a single observed variable x, with rate parameter β, for which the model specification corresponds to:

$$f(x(t), \beta) = -\exp(\beta)x(t)$$

$$g(x(t)) = x(t) \tag{51}$$

The model fit and estimated parameters are shown in Fig. 5C.

Finally, we consider a more involved example – the haemodynamic model of Stephan et al. (2007), which is used in the analysis of functional magnetic resonance imaging (fMRI) data to infer neural dynamics. (This forms function g in Eq. (1), Section 1.1). For a detailed walkthrough of the physiology, please see Appendix 5 of Zeidman et al. (2019a). In brief, there are four hidden states (vasoactive signal s, blood inflow f_{in}, blood volume v, deoxyhaemoglobin q) and three time-invariant parameters that are estimated from the timeseries data (haemodynamic transit time τ_h, vasoactive signal decay rate κ and stimulus efficacy z). The dynamics of the four hidden states are governed by the following equations, which together constitute f in Eq. (49):

$$\begin{aligned}
\dot{f}_{in} &= s \\
\dot{s} &= z(t) - \kappa s - \gamma(f_{in} - 1) \\
\tau_h \dot{v} &= f_{in}(t) - f_{out}(v, t) \\
\tau_h \dot{q} &= f_{in}(t)\frac{1 - (1 - E_0)^{\frac{1}{f_{in}(t)}}}{E_0} - \frac{f_{out}(v, t)q(t)}{v(t)}
\end{aligned} \tag{52}$$

Where $f_{out}(v, t) = v(t)^{\frac{1}{\alpha}}$ and α, γ are fixed parameters. The final part of the model translates from the latent variables for blood flow v and deoxyhaemoglobin q to the fMRI timeseries y:

$$\begin{aligned}
y &= V_0\left(k_1(1 - q) + k_2\left(1 - \frac{q}{v}\right) + k_3(1 - v)\right) \\
k_1 &= 4.3 \cdot \vartheta_0 \cdot E_0 \cdot TE \\
k_2 &= \epsilon_h \cdot r_0 \cdot E_0 \cdot TE \\
k_3 &= 1 - \epsilon_h
\end{aligned} \tag{53}$$

Where for this example, $\vartheta_0, E_0, r_0, TE, \epsilon_h$ are fixed parameters. Fig. 5D shows that the parameters were recovered successfully, although the precision with which the transit time parameter could be recovered was lower than the other two parameters, as reflected in the larger 90% credible interval (pink bar).

6.3. Bayesian model comparison

The main purpose of the VL scheme is to enable Bayesian model comparison, which is comparing the relative evidence for different models, where the log evidence is approximated by the free energy. If each model encodes a hypothesis for how the data were generated, then Bayesian model comparison enables different hypotheses to be compared, in terms of how well they trade off accuracy and complexity (see Section 3.2).

Models to be compared can differ in their likelihood – i.e., the definition of their forward model – and/or in the specification of their priors. The only requirement is that all models have been fitted to the same data (where this is not the case, an alternative approach referred to as *Bayesian Data Comparison* may be considered, see Zeidman et al. (2019c)). Where models differ only in their priors, there is no need to fit each model separately to the data using the VL scheme – it is sufficient to fit one 'full' model, and use *Bayesian model reduction* to analytically

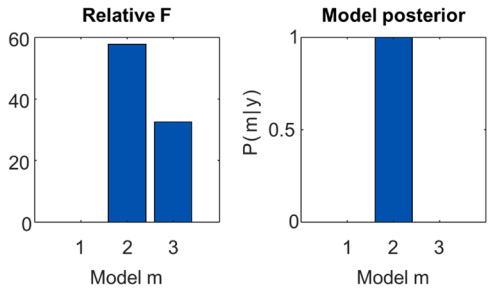


Fig. 6. Bayesian model comparison. Three general linear models (GLMs) were fitted to the data using the VL scheme, and then compared based on their free energy. The models differed in whether they had one, two or three precision components. **Left:** The log Bayes factor for each model relative to model 1. **Right:** The same results transformed to a posterior probability for each model.

compute the free energy and parameters of the alternative reduced models, under Laplace approximations (Friston et al., 2018). Through these methods, decisions such as whether to include particular variables in the model or how to capture their interactions can be decided in a principled manner.

To illustrate this, we will use GLM example above, where the first half of the observations were noisier than the second half (illustrated in Fig. 5B, dashed lines, left panel). We will then use Bayesian model comparison to ask which is the best explanation for the data: a model with one precision component (i.e., all observations had the same level of noise), two precision components (the first and second half of the observations had different levels of observation noise) or three components (each third of the data had a different level of noise). Including more precision components could increase the model's accuracy by introducing more degrees of freedom, but would also increase model complexity. To assess which of the three models optimized the accuracy / complexity trade-off, we specified each model and calculated their free energy using the VL scheme. The values were $F_1 = -18.21$, $F_2 = 39.61$, $F_3 = 14.32$, where a more positive free energy is better.

These free energies can then be compared in terms of the *Bayes factor* (Kass and Raftery, 1995) and posterior probabilities. The Bayes factor is the ratio of evidence for a model m_1 relative to another model m_2:

$$B_1 = \frac{p(y|m_1)}{p(y|m_2)} \tag{54}$$

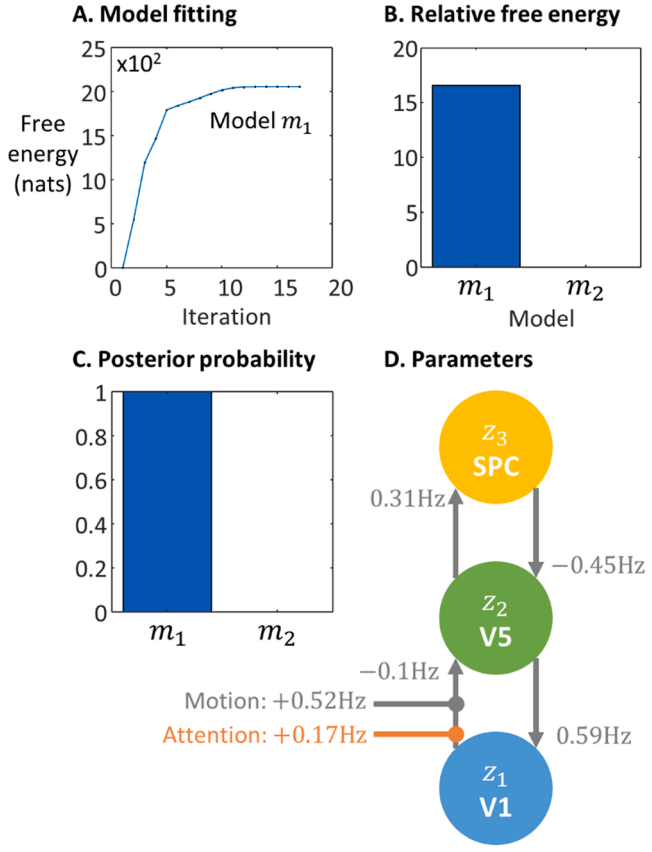


Fig. 7. Results of applying VL to the fMRI attention example. A. The free energy per iteration of the VL model estimation scheme for model m_1, relative to the free energy of the first iteration (i.e., this shows the increasing log Bayes factor). **B.** The free energy per model relative to m_2 which was set to zero. **C.** The posterior probability for each model. **D.** The expected values of the parameters. With reference to Appendix 1, which details the parameterisation of the model - numbers on the connections relate to parameter matrix A of the model, whereas the effects of motion and attention relate to the parameter matrices B of the model.

The larger the ratio, stronger the evidence in favour of model m_1. Taking the log, division becomes subtraction, therefore the log Bayes factor is simply the difference in log evidence. The log Bayes factor in favour of model 1 is:

$$\ln B_1 = \ln p(y|m_1) - \ln p(y|m_2) \approx F_1 - F_2 \tag{55}$$

The log Bayes factor can be computed for more than two models by selecting a model to serve as the baseline or reference. Here, we chose the worst model, m_1, as the reference model, and calculated log Bayes factor for models m_2 and m_3 relative to m_1, as shown in Fig. 6 (left panel). As expected, m_2 was the best (as the data were generated using two precision components), m_3 was the second best, and m_1 was the worst model. This example also makes a key point, that accuracy is not an apt measure of model quality. The third model here would have been the most accurate because it has more degrees of freedom. However, its added complexity was correctly penalized by the free energy, ensuring it would be discarded in favour of the model that is the simplest explanation for the data – but not too simple.

It can aid interpretation to report not only the log Bayes factor, but also the posterior probability for each model, e.g., $p(m_1|y)$. Under equal prior probability for each model, by application of Bayes rule, the posterior probabilities are given by a softmax function of the log Bayes factors:

$$\begin{aligned} p(m_i|y) &= \frac{p(y|m_i)}{p(y)} \\ &= \frac{1}{1+\exp(-\ln B_i)} \end{aligned} \tag{56}$$

This is illustrated in Fig. 6 (right panel). This demonstrates that m_2 had posterior probability close to unity, meaning that we could be extremely confident that it provided the best explanation for the data. This procedure may be applied with any number of models, and therefore forms the basis for hypothesis testing in Bayesian inference.

7. Empirical example of Bayesian model comparison

In Section 2 we introduced an example modelling problem, where the aim was to compare the evidence for two candidate models, as explanations for why visual region V5 of the brain is enhanced by visual attention. Here, we illustrate applying the VL scheme to the two candidate models, m_1 and m_2 and performing Bayesian model comparison.

Fig. 7A shows the free energy over iterations for model m_1 (relative to the first iteration). The algorithm converged after 17 iterations. The resulting free energies for the two models were $F_1 = -3277.61$ and $F_2 = -3294.20$ respectively, where a more positive free energy is better. These free energies were then taken forward for Bayesian model comparison.

The log Bayes factor was $\ln B_1 = F_1 - F_2 = 15.59$. Taking the exponential to undo the log, this means there was $\exp(15.59) = 5,897,269$ times the evidence for m_1 than m_2 (Fig. 7B). Naturally, therefore, the posterior probability in favour of m_1 approached unity (Fig. 7C).

Having reported the probability for each model, studies typically report the posterior estimates of the parameters from the winning model if there's a clear winner, or alternatively if there's no clear winner, the (precision-weighted) average of parameters across models (this is called Bayesian model averaging). Fig. 7D shows the posterior expected values from model m_1. It can be seen that under this model, the presence of visual motion boosted the connection from brain region V1 to V5 (by 0.52 Hz), and attention to visual motion further increased the strength of this connection (by 0.17 Hz).

Together, one may conclude that the data were best explained by hypothesis H1 – i.e., attentional modulation of V5 could be accounted for by bottom-up connectivity from V1. Examining the estimated parameters of the model demonstrated that attention had a gating effect on

feed-forward or ascending connections from primary visual cortex.

8. Discussion

The VL scheme described here underwrites thousands of neuroimaging studies, and is beginning to find applications in other fields. We considered it important, therefore, to clearly explain how it works, what assumptions it makes, and how to interpret the outputs. To this end, Sections 1-3 set out the key challenge of Bayesian inference – the intractable integral within the model evidence (Eq. (3)) – and how this can be resolved using variational Bayes (a.k.a., approximate Bayesian inference). This involves defining a lower bound on the log evidence (the free energy, Eq. (11)), and then identifying a probability density over the parameters that maximizes this bound, bringing it as close as possible to the log evidence. In effect, variational procedures convert an impossible integration or marginalisation problem into a tractable optimisation problem. Section 4 set out the implementation of variational Bayes typically used in neuroimaging (Eq. (31)), where Laplace approximations (i.e., normal densities) are used to approximate the free energy bound. Section 5 then described an efficient algorithm for maximizing this (approximate) free energy, as illustrated for static and dynamic models in Section 6. Worked derivations are provided in the supplementary materials, pseudocode is provided in the appendix and standalone MATLAB code accompanies this article.

This scheme has several advantages over alternative methods, both in terms of the free energy approximation of the log evidence and the algorithm used to maximize it. First, the free energy serves as a better approximation of the log evidence than other commonly used heuristics like the AIC and BIC. These approximations can fail dramatically even in relatively benign settings – see (Penny, 2012a) for some unsettling examples. While all three measures can be decomposed into accuracy and complexity terms, only the free energy takes into account uncertainty in the parameters for the accuracy term, as well as the covariance amongst parameters in the complexity term (Penny, 2012b). VL also has advantages over non-variational sampling methods. While sampling is highly effective for profiling the shape of a probability density, it does not provide a straightforward way to estimate the log evidence (one common approach, the *harmonic mean*, has been described as the "Worst Monte Carlo Method Ever" for its poor performance[9]). The lower computational cost of VL relative to sampling schemes, together with the fact that it is deterministic (so always provides the same results given the same data) provide further advantages. Regarding the algorithm described here for maximizing the free energy, the closest alternative is Expectation Maximization (EM). EM differs from variational Bayes in that EM ignores uncertainty about the hyperparameters. Thus, an advantage of VL is that it conveys the uncertainty of both the parameters and hyperparameters to the next iteration of the algorithm (i.e., EM is a special case of the variational Bayes where uncertainty about the hyperparameters is ignored.)

There are potential drawbacks of the VL scheme. First, the Laplace assumption may not be suitable for all applications. For instance, a Gaussian posterior may not be appropriate where the true posterior is multimodal, if a multimodal posterior is important for making inferences. To evaluate this for a particular application, the validity of the Laplace assumption can be assessed using sampling methods. This can be particularly useful when dealing with highly nonlinear models. Typically, variational Laplace accommodates nonlinearities by applying gaussian assumptions to nonlinear transformations of the parameters. A nice example of this is the treatment of hyperparameters above Eqs. (47)-(48). By taking the exponential of the hyperparameter, one is effectively assuming a log normal prior, which ensures positivity for scale parameters of this sort (a scale parameter is a nonnegative parameter, such as a rate or time constant, variance or distance measure). Using the same device in hierarchical and nonlinear models allows one to accommodate a large range of (weakly) non-linear models within variational Laplace. However, it is sometimes necessary to check the robustness to violations of the Laplace assumption with reference to sampling schemes.

Second, the algorithm presented here is highly likely to converge to an optimal value, but only if the initial values of the parameters are well chosen (typically, the prior expected values of the parameters are used). In other words, if there are multiple local optima, then the algorithm may not be able to escape the local optimum that is easiest to reach from the starting value (this is sometimes expressed as starting within the basin of attraction of a fixed point in the free energy landscape). To overcome this, multi-start approaches have been used in conjunction with the VL scheme. For example, in the analysis of neuroimaging data from multiple test subjects, a common approach is to iteratively restart the algorithm from the group average parameter values (Friston et al., 2015). Finally, it should be noted that variational Bayes methods commonly suffer from overconfidence in their posterior parameter estimates, as demonstrated in the context of DCM for fMRI by Daunizeau et al. (2012).

Various extensions and variants of the VL scheme have been developed to handle a broader range of models in the context of neuroimaging, which we have not had space to detail in this article. For example, VL has been applied to modelling data in the frequency domain (complex cross-spectra), which is routinely used in the analysis of electrophysiological data (Moran et al., 2009) and resting state fMRI data (Friston et al., 2014). Similar approaches have been introduced to invert stochastic differential equation models, namely Dynamic Expectation Maximization (DEM) and Generalised Filtering (GF), which estimate a model's hidden states and parameters, treating both as time-dependant random variables (Friston et al., 2010; Friston et al., 2008b). VL has also been applied to model voxel-wise fMRI data, yielding maps of parameters and posterior probabilities (Zeidman et al., 2018; Puckett et al., 2020). That approach leveraged parallelisation to estimate multiple voxels' timeseries independently; future work could improve performance by sharing parameters across voxels, such as those relating to haemodynamics or observation noise. More recently, the Parametric Empirical Bayes (PEB) framework was introduced to extend the VL scheme to hierarchical experimental designs, where for example, data have been sampled from multiple subjects at multiple time points (Zeidman et al., 2019b; Friston et al., 2016).

Since the introduction of the VL scheme, other algorithms and software tools for variational Bayesian inference have been introduced that serve a similar role. The Variational Message Passing (VMP) algorithm pre-dates VL (Winn et al., 2005) and is now used in the Microsoft infer.NET programming language and the ForneyLab Julia package (Cox et al., 2019). Like VL, this is a deterministic algorithm, which inverts models that can be expressed as *Bayesian networks* or *Forney Factor Graphs* (with the prerequisite that nodes are conjugate to their parent). The distributed nature of the VMP algorithm has also enabled its use as a model for how inference is performed in biological neural networks (Parr et al., 2019). Another prominent algorithm is automatic differentiation variational inference (ADVI) (Kucukelbir et al., 2017), which is implemented in multiple probabilistic programming frameworks including Stan (Kucukelbir et al., 2015), Turing.jl for Julia (Ge et al., 2018), PyMC3 for Python and Tensorflow Probability (TFP). Like VL, ADVI optimizes the free energy, however this is performed without Laplace approximations of the free energy functional. Instead, the log joint in Eq. (11) is evaluated automatically from a given graphical model, and the expected value is approximated using Monte Carlo integration methods, when the free energy is evaluated. Having defined the free energy functional, the next step is to maximize it, and VL and ADVI differ in how they do this. In VL, the gradient of the free energy under the Laplace approximation is given by Eq. (40), which depends on first computing the gradient of the function to optimized, $g(\boldsymbol{\theta})$, using

[9] https://radfordneal.wordpress.com/2008/08/17/the-harmonic-mean-of-the-likelihood-worst-monte-carlo-method-ever/

numerical methods (finite differences). ADVI estimates the gradient of the free energy using Automatic differentiation (AD), which involves reducing the free energy into a graph of elemental math operations and then repeatedly applying the chain rule (Griewank, 1989; Iri, 1984). These gradients are then supplied to a stochastic gradient descent algorithm (Hoffman et al., 2013). An interesting future direction would be to compare the performance of the VL scheme against VMP and ADVI for the kind of models typically applied in neuroimaging. We would predict that VL and VMP would be significantly faster because they eschew sampling, however ADVI would offer the most accurate posteriors in situations where the Laplace approximation is violated. Finally, VL and ADVI may also be compared against a recently developed scheme called Stochastic VB (sVB), which was introduced in the neuroimaging literature and also features stochastic gradient descent. It has been applied to discovering functional modes in neuroimaging data using hidden Markov models (Vidaurre et al., 2017) and quantifying variability in resting state networks across a large sample of the population in the UK Biobank (Farahibozorg et al., 2021).

The code accompanying this paper illustrates applications of the VL scheme with a variety of models. Readers interested in learning more about the applications of VL may wish to proceed to recent tutorials on modelling neuroimaging data using dynamic causal modelling (DCM) (Zeidman et al., 2019a) and behavioural data using active inference (Smith et al., 2022).

9. Code availability

MATLAB code accompanying this paper can be downloaded from https://github.com/pzeidman/vl-tutorial.

10. Data and code availability

All MATLAB code accompanying this paper can be downloaded from Github at https://github.com/pzeidman/vl-tutorial.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

The Wellcome Centre for Human Neuroimaging is supported by core funding from Wellcome [203147/Z/16/Z].

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.neuroimage.2023.120310.

Appendix 1. DCM for fMRI

This appendix details the parameterisation of connectivity matrices used in the attention to visual motion example. The neural connectivity matrix in Eq. (1) is defined as:

$$\boldsymbol{J}(t) := \left(\boldsymbol{A} + u_2(t)\boldsymbol{B}^{(MOTION)} + u_3(t)\boldsymbol{B}^{(ATTENTION)}\right)$$

Where binary indicator variables $u_2(t)$ and $u_3(t)$ denote, respectively, whether stimuli were in motion at time t and whether participants were instructed to pay attention to speed of the motion (Fig. 1A, top). The other terms are matrices of connectivity parameters, which have units of Hertz (s^{-1}) and need to be estimated from the data. For the first hypothesis, H1, the connectivity matrices are parameterised as follows:

$$\boldsymbol{A} := \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & a_{32} \\ 0 & a_{23} & a_{33} \end{bmatrix}$$

$$\boldsymbol{B}^{(MOTION)} := \begin{bmatrix} 0 & 0 & 0 \\ b_{21}^{(MOTION)} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B}^{(ATTENTION)} := \begin{bmatrix} 0 & 0 & 0 \\ b_{21}^{(ATTENTION)} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

In detail, matrix $\boldsymbol{A}$ encodes the coupling strength amongst brain regions, where element $a_{i,j}$ is the strength of the connection from region j to region i in units of hertz (s^{-1}). Thus, the columns of matrix $\boldsymbol{A}$ correspond to outgoing connections from regions V1, V5 and SPC, and the rows correspond to incoming connections. Parameters $b_{21}^{(MOTION)}$ and $b_{21}^{(ATTENTION)}$ encode the change in the V1→V5 connection due to motion and attention respectively. Finally, parameter matrix $\boldsymbol{C}$ encodes the sensitivity of each brain region to the driving effect of visual stimuli to region V1:

$$\boldsymbol{C} := \begin{bmatrix} c_{11} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

A model corresponding to the second hypothesis, H2, may be specified through a minor modification to the parameter matrix $\boldsymbol{B}^{(ATTENTION)}$, to indicate that attention should modulate the SPC→V5 connection rather than the V1→SPC connection:

$$\boldsymbol{B}^{(ATTENTION)} := \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & b_{23}^{(ATTENTION)} \\ 0 & 0 & 0 \end{bmatrix}$$

This second model is illustrated in Fig. 1B (right). In practice, the elements of these parameter matrices were either switched on (i.e., informed by the data) or switched off (fixed at zero) by selecting appropriate prior means and variances.

Appendix 2. Variational Laplace pseudocode

Inputs:
Generative model $g(\boldsymbol{\mu}_{\boldsymbol{\beta}})$
Data vector $\boldsymbol{y}$
Starting values for (hyper)parameters $\boldsymbol{\mu}_{\boldsymbol{\beta}}$ and $\boldsymbol{\mu}_{\boldsymbol{\lambda}}$
Prior expected values $\boldsymbol{\eta}_{\beta}$and $\boldsymbol{\eta}_{\lambda}$
Prior precision matrices $\boldsymbol{\Pi}_{\boldsymbol{\beta}}$ and $\boldsymbol{\Pi}_{\boldsymbol{\lambda}}$
Precision component matrices $\boldsymbol{Q}_1 \ldots \boldsymbol{Q}_k$
Outputs:
Posterior expected values $\boldsymbol{\mu}_{\boldsymbol{\beta}}$ and $\boldsymbol{\mu}_{\boldsymbol{\lambda}}$
Posterior covariances $\boldsymbol{\Sigma}_{\boldsymbol{\beta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\lambda}}$
Data precision matrix $\boldsymbol{\Pi}_y$
Free energy F
// Initialize to a high level of regularization
$v = -4$
until convergence
// Compute numerical derivative of g wrt parameters
$(\mathbf{J}_g)_{ij} = \partial_{\mu_j} g_i(\boldsymbol{\mu}_{\boldsymbol{\beta}})$
/* Optionally, if the Jacobian is unstable, make small parameter updates and re-try computing the Jacobian */
// Optimize hyperparameters
until convergence
// Compute data precision $\boldsymbol{\Pi}_y$ and data precision per component $\boldsymbol{P}_i$
$\boldsymbol{P}_i = \exp(\boldsymbol{\mu}_{\boldsymbol{\lambda}})_i \boldsymbol{Q}_i$
$\boldsymbol{\Pi}_y = \sum_i \boldsymbol{P}_i$
// Compute posterior covariance over parameters
$\boldsymbol{\Sigma}_{\boldsymbol{\beta}} = (\boldsymbol{J}_g^T \boldsymbol{\Pi}_y \boldsymbol{J}_g + \boldsymbol{\Pi}_{\boldsymbol{\beta}})^{-1}$
// Compute error terms
$\epsilon_{\lambda} = \boldsymbol{\mu}_{\lambda} - \boldsymbol{\eta}_{\lambda}$
$\epsilon_y = \boldsymbol{y} - \boldsymbol{g}(\boldsymbol{\mu}_{\beta})$
// Compute 1st and 2nd derivatives of the expected hyperparameters wrt g
$\dot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}} = \partial_{\mu_{\lambda_i}} E_{Q(\beta)}[\ln P(\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{\lambda})]$
$= \frac{1}{2} tr(\boldsymbol{P}_i \boldsymbol{\Pi}_y^{-1}) - \frac{1}{2} \epsilon_y^T \boldsymbol{P}_i \epsilon_y - \partial_{\mu_{\lambda_i}} (\epsilon_{\lambda})^{\mathrm{T}} \boldsymbol{\Pi}_{\boldsymbol{\lambda}} \epsilon_{\lambda} - \frac{1}{2} tr(\boldsymbol{\Sigma}_{\boldsymbol{\beta}} \boldsymbol{J}_g^T \boldsymbol{P}_i \boldsymbol{J}_g)$
$\ddot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}} = \partial_{\mu_{\lambda_i} \mu_{\lambda_i}} E_{Q(\beta)}[\ln P(\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{\lambda})]$
$= -(\boldsymbol{\Pi}_{\boldsymbol{\lambda}})_{ii} + \frac{1}{2} tr(\boldsymbol{P}_i \boldsymbol{\Sigma}_y \boldsymbol{P}_i \boldsymbol{\Sigma}_y) - \frac{1}{2} \epsilon_y^T \boldsymbol{P}_i \epsilon_y - \frac{1}{2} tr(\boldsymbol{\Sigma}_{\boldsymbol{\beta}} \boldsymbol{J}_g^T \boldsymbol{P}_i \boldsymbol{J}_g)$
// Scale a relaxed regularization value ($v = 4$) by the curvature
$t = \exp(4 - Re(\ln|\ddot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}|)/n)$
// Update hyperparameters
$\boldsymbol{\mu}_{\boldsymbol{\lambda}} = \boldsymbol{\mu}_{\boldsymbol{\lambda}} + (\exp[t \times \ddot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}] - \boldsymbol{I}) \ddot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}^{-1} \dot{\boldsymbol{\mu}}_{\boldsymbol{\lambda}}(t)$
end
// Calculate free energy
$F = \ldots$ // (see Eq. (30))
// Assess performance
if F has improved or we're early in the optimization then
// Compute errors
$\epsilon_{\beta} = \boldsymbol{\mu}_{\beta} - \boldsymbol{\eta}_{\beta}$
$\epsilon_y = \boldsymbol{y} - \boldsymbol{g}(\boldsymbol{\mu}_{\beta})$
/* Compute 1st and 2nd derivatives of the expected parameters wrt g for parameter update */
$\dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}} = \mathbf{J}_g^T \boldsymbol{\Pi}_y \epsilon_y - \boldsymbol{\Pi}_{\boldsymbol{\beta}} \epsilon_{\boldsymbol{\beta}}$
$\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}} = -\boldsymbol{J}_g^T \boldsymbol{\Pi}_y \boldsymbol{J}_g - \boldsymbol{\Pi}_{\boldsymbol{\beta}}$
// Decrease regularization
$v = v + 1/2$
else
// Free energy has got worse
$\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\mu}_{\boldsymbol{\lambda}} \leftarrow$ restore previous parameter estimates
// Increase regularization
$v =$ v - 2
end
// Scale the log-regularization v by the curvature
$t = \exp(v - Re(\ln|\ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}|)/n)$
// Update parameters
$\boldsymbol{\mu}_{\boldsymbol{\beta}} = \boldsymbol{\mu}_{\boldsymbol{\beta}} + (\exp[t \times \ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}] - \boldsymbol{I}) \ddot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^{-1} \dot{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$

(continued on next page)

(*continued*)

```
    if the change in F is repeatedly below criterion
    // convergence
        Return
  end
End
```

References

Büchel, C., Josephs, O., Rees, G., Turner, R., Frith, C.D., Friston, K.J, 1998. The functional anatomy of attention to visual motion. A functional MRI study. Brain: a journal of neurology 121, 1281–1294.

Beal, M.J., 2003. PhD Thesis. University of London.

Bishop, C.M., 1998. Latent Variable models. Learning in Graphical Models. Springer.

Chappell, M., Groves, A. & Woolrich, M. 2016. The FMRIB variational Bayes tutorial: Variational Bayesian inference for a non-linear forward model [Online]. Available: https://ora.ox.ac.uk/objects/uuid:8a90a2a5-4748-4557-a6f2-4eee5f8b07ae [Accessed 08/06/2022].

Cox, M., Van De Laar, T., De Vries, B., 2019. A factor graph approach to automated design of Bayesian signal processing algorithms. International Journal of Approximate Reasoning 104, 185–204.

Daunizeau, J., Stephan, K.E., Friston, K.J., 2012. Stochastic dynamic causal modelling of fMRI data: should we care about neural noise? Neuroimage 62, 464–481.

Daunizeau, J., Adam, V., Rigoux, L., 2014. VBA: a probabilistic treatment of nonlinear models for neurobiological and behavioural data. PLoS computational biology 10, e1003441.

Daunizeau, J. 2017. The variational Laplace approach to approximate Bayesian inference. arXiv preprint.

Farahibozorg, S.R., Bijsterbosch, J.D., Gong, W., Jbabdi, S., Smith, S.M., Harrison, S.J., Woolrich, M.W., 2021. Hierarchical modelling of functional brain networks in population and individuals from big fMRI data. NeuroImage 243, 118513.

Feynman, R., 1972. Statistical Mechanics. W. A. Benjamin.

Frässle, S., Aponte, E.A., Bollmann, S., Brodersen, K.H., Do, C.T., Harrison, O.K., Harrison, S.J., Heinzle, J., Iglesias, S., Kasper, L, 2021. TAPAS: an open-source software package for translational neuromodeling and computational psychiatry. Frontiers in psychiatry 12, 680811.

Friston, K.J., Harrison, L., Penny, W., 2003. Dynamic causal modelling. Neuroimage 19, 1273–1302.

Friston, K.J., Mattout, J., Trujillo-Barreto, N., Ashburner, J., Penny, W., 2007. Variational free energy and the Laplace approximation. Neuroimage 34, 220–234.

Friston, K., Harrison, L., Daunizeau, J., Kiebel, S., Phillips, C., Trujillo-Barreto, N., Henson, R., Flandin, G., Mattout, J., 2008a. Multiple sparse priors for the M/EEG inverse problem. NeuroImage 39, 1104–1120.

Friston, K.J., Trujillo-Barreto, N., Daunizeau, J., 2008b. DEM: a variational treatment of dynamic systems. Neuroimage 41, 849–885.

Friston, K., Stephan, K., Li, B., Daunizeau, J., 2010. Generalised filtering. Mathematical Problems in Engineering 2010.

Friston, K.J., Kahan, J., Biswal, B., Razi, A., 2014. A DCM for resting state fMRI. Neuroimage 94, 396–407.

Friston, K., Zeidman, P., Litvak, V., 2015. Empirical Bayes for DCM: A Group Inversion Scheme. Front Syst Neurosci 9, 164.

Friston, K.J., Litvak, V., Oswal, A., Razi, A., Stephan, K.E., Van Wijk, B.C., Ziegler, G., Zeidman, P., 2016. Bayesian model reduction and empirical Bayes for group (DCM) studies. Neuroimage 128, 413–431.

Friston, K., Parr, T. & Zeidman, P. 2018. Bayesian model reduction. arXiv preprint.

Friston, K.J., Parr, T., Zeidman, P., Razi, A., Flandin, G., Daunizeau, J., Hulme, O.J., Billig, A.J., Litvak, V., Moran, R.J., Price, C.J., Lambert, C., 2020. Dynamic causal modelling of COVID-19. Wellcome Open Research 5.

Ge, H., Xu, K., Ghahramani, Z., 2018. Turing: a language for flexible probabilistic inference. In: International conference on artificial intelligence and statistics. PMLR, pp. 1682–1690.

Griewank, A., 1989. On automatic differentiation. Mathematical Programming: recent developments and applications 6, 83–107.

Hinton, G., Van Camp, D., 1993. Keeping neural networks simple by minimizing the description length of the weights. In: Proc. of the 6th Ann. ACM Conf. on Computational Learning Theory. Citeseer.

Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J., 2013. Stochastic variational inference. Journal of Machine Learning Research.

Iri, M., 1984. Simultaneous computation of functions, partial derivatives and estimates of rounding errors—Complexity and practicality—. Japan Journal of Applied Mathematics 1, 223–252.

Jaynes, E.T., 1957. Information theory and statistical mechanics. Physical review 106, 620.

Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K., 1999. An introduction to variational methods for graphical models. Machine learning 37, 183–233.

Kass, R.E., Raftery, A.E., 1995. Bayes factors. Journal of the american statistical association 90, 773–795.

Kucukelbir, A., Ranganath, R., Gelman, A., Blei, D., 2015. Automatic variational inference in Stan. Advances in neural information processing systems 28.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., Blei, D.M., 2017. Automatic differentiation variational inference. Journal of machine learning research.

Lanillos, P. & Van Gerven, M. 2021. Neuroscience-inspired perception-action in robotics: applying active inference for state estimation, control and self-perception. arXiv preprint.

Lanillos, P., Meo, C., Pezzato, C., Meera, A.A., Baioumy, M., Ohata, W., Tschantz, A., Millidge, B., Wisse, M. & Buckley, C.L. 2021. Active Inference in Robotics and Artificial Agents: Survey and Challenges. arXiv preprint.

Mackay, D.J., 1995a. Free energy minimisation algorithm for decoding and cryptanalysis. Electronics Letters 31, 446–447.

Mackay, D.J., 1995b. Probable networks and plausible predictions—A review of practical Bayesian methods for supervised neural networks. Network: computation in neural systems 6, 469–505.

Moran, R.J., Stephan, K.E., Seidenbecher, T., Pape, H.C., Dolan, R.J., Friston, K.J., 2009. Dynamic causal models of steady-state responses. Neuroimage 44, 796–811.

Neal, R.M., Hinton, G.E., 1998. A View of the EM Algorithm That Justifies incremental, sparse, and Other variants. Learning in Graphical Models. Springer.

Ostwald, D., Kirilina, E., Starke, L., Blankenburg, F., 2014. A tutorial on variational Bayes for latent linear stochastic time-series models. Journal of Mathematical Psychology 60, 1–19.

Ozaki, T. 1985. Non-linear time series models and dynamical systems. Handbook of Statistics.

Parr, T., Markovic, D., Kiebel, S.J., Friston, K.J., 2019. Neuronal message passing using Mean-field, Bethe, and Marginal approximations. Scientific reports 9, 1–18.

Penny, W.D., 2012a. Comparing dynamic causal models using AIC, BIC and free energy. Neuroimage 59, 319–330.

Penny, W.D., 2012b. Comparing dynamic causal models using AIC, BIC and free energy. Neuroimage 59, 319–330.

Puckett, A.M., Bollmann, S., Junday, K., Barth, M., Cunnington, R., 2020. Bayesian population receptive field modeling in human somatosensory cortex. Neuroimage 208, 116465.

Smith, R., Friston, K.J., Whyte, C.J., 2022. A step-by-step tutorial on active inference and its application to empirical data. Journal of Mathematical Psychology 107, 102632.

Starke, L., Ostwald, D., 2017. Variational Bayesian parameter estimation techniques for the general linear model. Frontiers in neuroscience 11, 504. https://doi.org/10.3389/fnins.2017.00504.

Stephan, K.E., Friston, K.J., Penny, W.D., 2005. Technical report. Wellcome Department of Imaging Neuroscience, ION, UCL.

Stephan, K.E., Weiskopf, N., Drysdale, P.M., Robinson, P.A., Friston, K.J., 2007. Comparing hemodynamic models with DCM. Neuroimage 38, 387–401.

Vidaurre, D., Smith, S.M., Woolrich, M.W., 2017. Brain network dynamics are hierarchically organized in time. Proceedings of the National Academy of Sciences 114, 12827–12832.

Winn, J., Bishop, C.M., Jaakkola, T., 2005. Variational message passing. Journal of Machine Learning Research 6.

Zeidman, P., Silson, E.H., Schwarzkopf, D.S., Baker, C.I., Penny, W., 2018. Bayesian population receptive field modelling. NeuroImage 180, 173–187.

Zeidman, P., Jafarian, A., Corbin, N., Seghier, M.L., Razi, A., Price, C.J., Friston, K.J., 2019a. A guide to group effective connectivity analysis, part 1: First level analysis with DCM for fMRI. NeuroImage 200, 174–190.

Zeidman, P., Jafarian, A., Seghier, M., Litvak, V., Cagnan, H., Price, C.J., Friston, K., 2019b. A guide to group effective connectivity analysis, part 2: Second level analysis with PEB. NeuroImage 200, 12–25.

Zeidman, P., Kazan, S.M., Todd, N., Weiskopf, N., Friston, K.J., Callaghan, M.F., 2019c. Optimizing data for modeling neuronal responses. Frontiers in neuroscience 12, 986.