NTNU
Norwegian University of
Science and Technology

The Expectation Propagation Algorithm for use in Approximate Bayesian Analysis of Latent Gaussian Models

Christian Skar

Master of Science in Physics and Mathematics
Submission date: June 2010
Supervisor: Håvard Rue, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

The purpose of this master thesis is to explore the merging of the Laplace approximation and Expectation Propagation with the aim of doing approximate Bayesian inference in the class of latent Gaussian Markov random field models.

Assignment given: 15. January 2010
Supervisor: Håvard Rue, MATH

Preface

This Master's thesis is written under the course code *TMA4905: Master's thesis in statistics* for the Department of Mathematical Sciences at the Norwegian University of Science and Technology (NTNU). It marks the end of a five year study program in *Applied Physics and Mathematics*, with a specialization in Industrial Mathematics, and is worth 30 credit points.

The work on this thesis has been both challenging and rewarding, and has provided me with valuable insight in the field of approximate Bayesian analysis and Gaussian Markov random fields. I am lucky to have had a supervisor, Håvard Rue, with a lot of knowledge on this field, and would like to thank Håvard for all the excellent help I have received during this work.

I also wish to thank my office companions, Elisabeth Larsen and Johan Fatnes, for many interesting debates over the past five years and for all the input they have provided. Lastly, I wish to thank my girlfriend, Sara Linn Hansen, for all of her support this final semester of my studies.

Christian Skar,
June, 2010,
Trondheim.

Abstract

Analyzing latent Gaussian models by using approximate Bayesian inference methods has proven to be a fast and accurate alternative to running time consuming Markov chain Monte Carlo simulations. A crucial part of these methods is the use of a Gaussian approximation, which is commonly found using an asymptotic expansion approximation. This study considered an alternative method for making a Gaussian approximation, the expectation propagation (EP) algorithm, which is known to be more accurate, but also more computationally demanding. By assuming that the latent field is a Gaussian Markov random field, specialized algorithms for factorizing sparse matrices was used to speed up the EP algorithm. The approximation methods were then compared both with regards to computational complexity and accuracy in the approximations. The expectation propagation algorithm was shown to provide some improvements in accuracy compared to the asymptotic expansion approximation when tested on a binary logistic regression model. However, tests of computational time requirement for computing approximations in simple examples show that the EP algorithm is as much as 15-20 times slower than the alternative method.
Keywords: Expectation propagation, approximate Bayesian inference, latent Gaussian models, Gaussian Markov random fields, asymptotic expansion approximations.

Contents

1 Introduction

One of the main topics in modern statistics is studying problems using Bayesian inference. This often involves large hierarchical models which are typically analyzed using simulation methods such as Markov chain Monte Carlo (MCMC) methods, see Gamerman and Lopes (2006). These methods are relatively easy to use, even for large complex Bayesian models. Though, owing to their computer intensive nature MCMC simulations can for large problems use hours or even days to complete (Rue et al., 2009). For a special class of Bayesian models, namely the *latent Gaussian models*, Rue et al. (2009) proposed to bypass MCMC completely by employing deterministic approximation methods instead. The methods suggested by Rue et al. (2009) were shown to be remarkable fast and accurate, and provides a valuable tool in Bayesian statistics as the use of latent Gaussian models is widespread in practical applications. However, some parts of the approximation procedure were under a bit of scrutiny in the discussion part of Rue et al. (2009, pp. 353-388) as approximations methods from the machine learning literature were not properly tested with this procedure. In particular the expectation propagation (EP) algorithm was mentioned as a possible option for improving the accuracy in the approximations. This consideration forms the starting point of the study presented here.

The main purpose of this paper is to study the expectation propagation algorithm for use in approximate Bayesian inference of latent Gaussian models. Special attention will be on accuracy gain versus computational time increase compared to the method used in Rue et al. (2009).

The paper will have the following structure. Section 2 will give an overview of Gaussian Markov random fields and particularly numerical methods for dealing with GMRFs. The most important point is how to efficiently deal with sparse symmetric positive definite matrices, and easy to implement algorithms for band matrices are presented. Use of GMRFs in hierarchical Bayesian models, yielding the so-called latent Gaussian models, is treated in Section 3. As we will see, latent Gaussian models provides a powerful modelling tool which can be applied to a wide range of problems, see Rue et al. (2009) for examples. Analysis of these models using the approximate Bayesian inference scheme developed by Rue et al. (2009) is presented in Section 4. This section will discuss why these models are particularly well suited for approximate inference and present the basic steps involved in the approximation process. Particularly we will see the need of computing an unnormalized Gaussian approximation. Two methods for dealing with this task are presented in Sec. 5. The first method is based on an asymptotic expansion and is the method used by Rue et al. (2009). The second method is the expectation propagation algorithm developed by Minka (2001), and is based on optimization. This algorithm has become quite popular in the machine learning community, where it is often used with models where the latent Gaussian field is not assumed to have the Markov property (Minka, 2001; Rasmussen and Williams, 2006). The EP algorithm is known to provide better Gaussian approximations than the asymptotic expansion method, see Minka (2001) and Rasmussen and Williams (2006), however it is also known to be slower (Rue et al., 2009). By allowing the latent field to be a GMRF one can improve the efficiency of EP by incorporating the numerical methods from Sec. 2. This paper will

investigate if the computational complexity is reduced to such an extent that EP can be used for fast inference when applied in the approximate scheme from Sec. 4. If so, the approximation scheme can enjoy the improved accuracy in cases where the asymptotic expansion method does not work so well. The analysis presented in Sec. 6 will address the speed and accuracy tradeoff associated with the two methods. Computational time requirement is studied by looking at examples where the band algorithms from Sec. 2 can be applied. Both methods and the linear algebra routines used by the methods are implemented from scratch in C, and the idea is to study the two methods on an equal footing. The accuracy gain by applying EP is studied by analyzing a data set where the asymptotic expansion method does not give the best approximations. Hopefully this will shed some light on the issues raised by machine learning researchers in the discussion part of Rue et al. (2009, pp. 353-388).

1.1 Notation

The notation used in text is more or less identical to what is used in Rue et al. (2009). Probability distributions are denoted such that $\pi(x)$ is the distribution of a random variable (r.v.) x and $\pi(x|y)$ is the distribution of a r.v. x conditioned on a second r.v. y. The Gaussian distribution holds a special place in this text and the following convention is used: $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ means that the $\boldsymbol{x}$ is Gaussian distributed with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. A Gaussian probability density function (pdf) with these moments, evaluated at $\boldsymbol{x}$, is denoted $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Scalars and vectors are denoted by small Latin or Greek letters and are identified by the use of boldface fonts for vectors, e.g. $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$. Matrices are written with bold face capital letters, and to indicate that a matrix $\boldsymbol{A}$ have elements A_{ij} (for suitable combinations of i and j) we write $\boldsymbol{A} = (A_{ij})$. Similarly, a matrix with column vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$ is written $\boldsymbol{A} = (\boldsymbol{a}_j)$. In algorithms the process of solving for $\boldsymbol{x}$ in a linear system of equations $\boldsymbol{Ax} = \boldsymbol{b}$ is denoted $\boldsymbol{x} = \boldsymbol{A}\backslash\boldsymbol{b}$ in which it should be clear from context what algorithm to use for this purpose. Also, the process of solving $\boldsymbol{x}$ is sometimes shorted down to 'solving $\boldsymbol{A}^{-1}\boldsymbol{b}$' (thus it is not $\boldsymbol{b}$ we are solving for). In vectors, matrices and sets the removal of a subsets of elements is indicated by a minus in the subscript, say if $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ then $\boldsymbol{x}_{-\{1,3\}} = (x_2, x_4, x_5, \ldots, x_n)^{\mathrm{T}}$. Lastly, regular sequences of numbers are given the usual way, that is if $i, j \in \mathbb{N}$ and $i < j$, then $i : j = (i, i+1, \ldots, j-1, j)^{\mathrm{T}}$.

The rest of the notation used should be self-explanatory or be introduced along the way.

2 Gaussian Markov Random Fields

The overall goal in this text is to study the expectation propagation algorithm for use in approximate Bayesian analysis of latent Gaussian models. The key feature of the models discussed here is that the observations are assumed to depend on an underlying (or unobserved, also referred to as *latent*) field with special properties. Throughout this text the underlying field is a *Gaussian Markov random field* (GMRF), which turns out to have

some highly desirable properties with regards to reducing the computational workload involved in the analysis. Using GMRFs in this context is a fairly new phenomenon in computational statistics and the first complete account of this topic is given in Rue and Held (2005). This section presents a selection of the theory concerning Gaussian Markov random fields and is basically a summary of Rue and Held (2005, Ch. 2).

2.1 Preliminaries

In short a Gaussian Markov random field is a collection of random variables which are jointly Gaussian distributed and have conditional independence properties given by a graph. In order to gain an understanding of what this means a quick introduction of the Gaussian distribution and some ideas from graph theory is needed, starting with the multivariate Gaussian distribution.

2.1.1 The Gaussian distribution

Definition 2.1 (Multivariate Gaussian distribution)**.** Let $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ be a n-dimensional random vector with expectation $\mathrm{E}(\boldsymbol{x}) = \boldsymbol{\mu}$ and (symmetric positive definite) covariance matrix $\mathrm{cov}(\boldsymbol{x}) = \boldsymbol{\Sigma}$. We say that $\boldsymbol{x}$ is multivariate Gaussian distributed, written $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, if the probability density function is on the form

$$\pi(\boldsymbol{x}) = (2\pi)^{-n/2}(\det \boldsymbol{\Sigma})^{-1/2} \exp\Big\{ -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \Big\}, \quad \boldsymbol{x} \in \mathbb{R}^n. \tag{2.1}$$

This distribution is perhaps the most widely used of all multivariate distributions in statistics and some of its popularity is due to a few highly useful analytical results. A summary of the most important properties with the Gaussian distribution is given in Result 2.1.

Result 2.1. *Let $\boldsymbol{x}$ be a Gaussian random vector with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ and consider any partition of the elements in $\boldsymbol{x}$ and the corresponding partitions in $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, given as*[1]

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_A \\ \boldsymbol{x}_B \end{pmatrix}, \quad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{pmatrix} \quad \textit{and} \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{AA} & \boldsymbol{\Sigma}_{AB} \\ \boldsymbol{\Sigma}_{BA} & \boldsymbol{\Sigma}_{BB} \end{pmatrix}. \tag{2.2}$$

Then the following results apply

1. *Linear transformation: Let $\boldsymbol{A} \in \mathbb{R}^{k \times n}$, $k < n$, be a rank k matrix and define $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$. Then $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{\mu}, \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^{\mathrm{T}})$.*
2. *Marginalization: $\boldsymbol{x}_A \sim \mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_{AA})$.*
3. *Independence: $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$ are independent $\iff \boldsymbol{\Sigma}_{AB} = \boldsymbol{0}$.*
4. *Conditional distribution: $\boldsymbol{x}_A | \boldsymbol{x}_B \sim \mathcal{N}(\boldsymbol{\mu}_{A|B}, \boldsymbol{\Sigma}_{A|B})$ where $\boldsymbol{\mu}_{A|B} = \boldsymbol{\mu}_A + \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}(\boldsymbol{x}_B - \boldsymbol{\mu}_B)$ and $\boldsymbol{\Sigma}_{A|B} = \boldsymbol{\Sigma}_{AA} - \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}\boldsymbol{\Sigma}_{BA}$.*

[1] Ordering of the elements in $\boldsymbol{x}$ is irrelevant so all partitions can be written this way.

A proof of the first property can be found in Kingman and Taylor (2008, pp. 372-373) and the rest of the properties are treated in Johnson and Wichern (2007). Another useful (and non-statistical) property with the Gaussian distribution function is that the product of two Gaussian pdfs of the same argument is an un-normalized Gaussian pdf in that argument. By considering the product of $\mathcal{N}(\boldsymbol{x}|\boldsymbol{a}, \boldsymbol{A})$ and $\mathcal{N}(\boldsymbol{x}|\boldsymbol{b}, \boldsymbol{B})$ we have

$$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{a}, \boldsymbol{A}) \cdot \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{b}, \boldsymbol{B}) = Z^{-1}\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{c}, \boldsymbol{C}), \tag{2.3}$$

where $\boldsymbol{c} = \boldsymbol{C}(\boldsymbol{A}^{-1}\boldsymbol{a} + \boldsymbol{B}^{-1}\boldsymbol{b})$, $\boldsymbol{C} = (\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1})^{-1}$ and

$$Z^{-1} = (2\pi)^{-d/2}(\det(\boldsymbol{A} + \boldsymbol{B}))^{-1/2} \exp\Big\{ -\frac{1}{2}(\boldsymbol{a} - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{A} + \boldsymbol{B})^{-1}(\boldsymbol{a} - \boldsymbol{b})\Big\}.$$

This identity can be verified by simply multiplying the pdfs and grouping together terms in the exponent so that the only term involving $\boldsymbol{x}$ is quadratic. The result then follows by comparing with the pdf in Eq. (2.1). This result also shows that a ratio of Gaussian distribution functions is Gaussian, which is useful in the derivation of the expectation propagation algorithm in Sec. 5.2.

2.1.2 Undirected graphs and the Markov property

Before specifying what a Markov random field is a few basic ideas about graphs must be introduced. The reason for this is that graphs are very useful for defining the Markov property for a random field.

A graph $\mathcal{G}$ is an ordered pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of *vertexes* (or *nodes*) and $\mathcal{E}$ is a set of *edges* connecting elements in $\mathcal{V}$. This text will only consider graphs that are undirected and labeled. An *undirected* graph has the property that the edges are symmetric, i.e.

$$(v_1, v_2) = (v_2, v_1) \quad \text{for all} \quad (v_1, v_2) \in \mathcal{E}, \quad v_1, v_2 \in \mathcal{V}.$$

In a *labeled* graph the elements in $\mathcal{V}$ are labeled by the natural numbers so that $\mathcal{V} = \{1, 2, \ldots, |\mathcal{V}|\}$. An example of an undirected labeled graph is shown in Figure 2.1. In this graph the set of vertexes is $\mathcal{V} = \{1, 2, 3, 4\}$ and the set of edges is $\mathcal{E} = \{(1, 2), (2, 4), (3, 4), (1, 3)\}$. Two vertexes i and j in $\mathcal{V}$ are considered *neighbors*, denoted by $i \sim j$, if the edge (i, j)

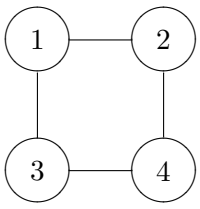


Figure 2.1: Example of an undirected graph.

is contained in $\mathcal{E}$. The collection of all the neighbors of a vertex i is denoted $\text{ne}(i)$, i.e

$$\text{ne}(i) = \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E}\}, \quad i \in \mathcal{V}.$$

For instance in Figure 2.1 we have $\text{ne}(1) = \{2,3\}$. A *path* p in a graph $\mathcal{G}$ is a sequence of distinct vertexes $v_1, v_2, \ldots, v_{j-1}, v_j \in \mathcal{V}$, so that $\mathcal{E}$ contains edges between successive vertexes, i.e. $(v_k, v_{k+1}) \in \mathcal{E}$ for $1 \leq k < j$. Examples of different paths in Fig. 2.1 are for instance $p = (1,2,4)$ and $p = (3,1,2)$. By letting A be a non-empty subset of $\mathcal{V}$ we can define a *sub-graph* of $\mathcal{G}$ as

$$\mathcal{G}_A = (A, \mathcal{E}_A), \quad A \subseteq \mathcal{V}, \quad \mathcal{E}_A = \{(i,j) \in \mathcal{E} \mid i,j \in A\}.$$

If A, B and C are disjoint subsets of $\mathcal{V}$ then C is said to *separate* A and B in $\mathcal{G}$ if there are no paths going from a vertex in A to a vertex in B that does not contain a vertex from C. For this to make sense the sets A and B are required to be non-empty, however C is not restricted to be non-empty. In the case where C is empty, A and B are separated in $\mathcal{G}$ to begin with.

The Markov property defines a special conditional independence structure in a random field. By definition the random vectors $\boldsymbol{u}$ and $\boldsymbol{w}$ are conditionally independent given the value of a random vector $\boldsymbol{v}$ if

$$\pi(\boldsymbol{u}, \boldsymbol{w} \mid \boldsymbol{v}) = \pi(\boldsymbol{u} \mid \boldsymbol{v})\pi(\boldsymbol{w} \mid \boldsymbol{v}),$$

or equivalently if

$$\pi(\boldsymbol{u} \mid \boldsymbol{w}, \boldsymbol{v}) = \pi(\boldsymbol{u} \mid \boldsymbol{v}) \quad (\iff \pi(\boldsymbol{w} \mid \boldsymbol{u}, \boldsymbol{v}) = \pi(\boldsymbol{w} \mid \boldsymbol{v})),$$

in which case we write $\boldsymbol{u} \perp \boldsymbol{w} | \boldsymbol{v}$. A random field $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ is said to be a Markov random field (MRF) with respect to $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if $\pi(\boldsymbol{x})$ satisfies a Markov property. Rue and Held (2005) lists three different types of Markov properties which is given in the following list.

1. *Pairwise Markov property:*

$$x_i \perp x_j \mid \boldsymbol{x}_{-ij} \quad \text{for } i \neq j \text{ and } i \nsim j \text{ in } \mathcal{G}.$$

2. *Local Markov property:*

$$x_i \perp \boldsymbol{x}_{-\text{ne}(i)\cup\{i\}} \mid \boldsymbol{x}_{\text{ne}(i)} \quad \text{for every } i \in \mathcal{V}.$$

3. *Global Markov property:*

$$\boldsymbol{x}_A \perp \boldsymbol{x}_B \mid \boldsymbol{x}_C, \quad A, B, C \subset \mathcal{V}$$

for every disjoint A, B and C, where A and B are non-empty and C separates A and B in $\mathcal{G}$.

Note that the global Markov property implies both the local and pairwise property. Also, according to Rue and Held (2005), it is possible to show that under certain conditions the three properties are equivalent.

One of the simplest examples of a Markov random field is a discrete time Markov chain with $\text{ne}(i) = \{i-1, i+1\}$ for $i \in \mathcal{V}$, as shown in Fig. 2.2. Common practice with Markov chains of this type is to define the Markov property by

$$\pi(x_{n+1} \mid \boldsymbol{x}_{i:n}) = \pi(x_{n+1} \mid x_n), \quad 1 \leq i \leq n,$$

which has the intuitive interpretation that a future state only depend on the most recent observation when conditioning on the entire (or parts of the) past. This notion is generalized for MRFs with more complex graphs, that is, elements in a random field with the Markov property is conditionally independent 'far away' data when given sufficient amount of 'close' data.

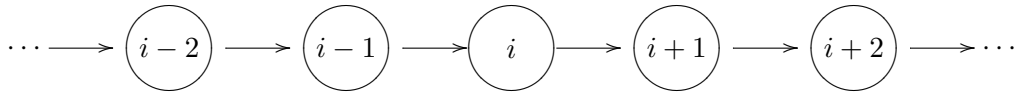


Figure 2.2: Graphical representation of a discrete time Markov chain.

2.2 GMRF

A Gaussian Markov random field is simply a Markov random field $\boldsymbol{x}$ with respect to an undirected labeled graph $\mathcal{G}$ with the additional property that $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. It turns out that in a Gaussian random field conditional independence between two elements given the rest of the field is given directly in the precision matrix (inverse covariance matrix) as is stated in Theorem 2.2.

Theorem 2.2. *Let* $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{Q}^{-1})$. *Then*

$$x_i \perp x_j \mid \boldsymbol{x}_{-ij} \quad \Longleftrightarrow \quad Q_{ij} = 0 \quad \textit{for all } i \neq j. \tag{2.4}$$

A proof can be found in Rue and Held (2005, p. 22). One direct consequence of this theorem is that for a GMRF with graph $\mathcal{G}$ the off-diagonal terms Q_{ij} will be zero unless i and j are neighbors in $\mathcal{G}$ (by the local Markov property). This result is very useful when it comes to the computational aspect of dealing with GMRFs as it implies that if the graph $\mathcal{G}$ is sparse (i.e. $\text{ne}(i) \ll n$ for most $i \in \mathcal{V}$) then $\boldsymbol{\Sigma}^{-1} = \boldsymbol{Q}$ is sparse as well (i.e. many elements are zero). Theorem 2.2 and the connection with the graph giving the conditional independence structure is used directly in the definition of a GMRF by Rue and Held (2005). This definition is adopted here.

Definition 2.2 (Gaussian Markov Random Field)**.** A collection of random variables $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}} \in \mathbb{R}^n$ is called a *Gaussian Markov Random Field* with respect to a

labeled graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with mean $\boldsymbol{\mu}$ and precision matrix $\boldsymbol{Q} > 0$ if its density is on the form

$$\pi(\boldsymbol{x}) = (2\pi)^{-n/2}(\det \boldsymbol{Q})^{1/2} \exp\Big\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{Q}(\boldsymbol{x}-\boldsymbol{\mu})\Big\} \tag{2.5}$$

and

$$Q_{ij} = 0 \quad \Longleftrightarrow \quad (i,j) \notin \mathcal{E} \quad \text{for all } i \neq j.$$

It may appear as though this defines a Markov random field with the local Markov property but in the case of GMRFs it turns out that the three Markov properties are equivalent, see Theorem 2.4 in Rue and Held (2005). Unlike the elements in the covariance matrix the precision matrix elements do not have any specific interpretation for single elements in a random field. However, the precision matrix elements can be interpreted in light of the full conditionals of $\boldsymbol{x}$ as is shown in the next theorem, which is found in Rue and Held (2005, p. 22).

Theorem 2.3. *Let $\boldsymbol{x}$ be a GMRF with respect to a graph $\mathcal{G}$, and let $\mathrm{E}(\boldsymbol{x}) = \boldsymbol{\mu}$ and $\mathrm{prec}(\boldsymbol{x}) = \boldsymbol{Q} > 0$. Then,*

$$\begin{aligned}
\mathrm{E}(x_i \mid \boldsymbol{x}_{-i}) &= \mu_i - \frac{1}{Q_{ii}} \sum_{j:j\sim i} Q_{ij}(x_j - \mu_j), \\
\mathrm{prec}(x_i \mid \boldsymbol{x}_{-i}) &= Q_{ii}, \quad \textit{and} \\
\mathrm{corr}(x_i, x_j \mid \boldsymbol{x}_{-ij}) &= \frac{Q_{ij}}{\sqrt{Q_{ii}Q_{jj}}} \quad i \neq j.
\end{aligned}$$

The diagonal elements of $\boldsymbol{Q}$ are interpreted as the precision in the correspond full conditional distributions, and the off-diagonal elements give, when properly scaled, the correlations between two elements given the rest of field.

The perhaps simplest non-trivial examples of a Gaussian Markov random field is the auto-regressive process (with Gaussian inventions) of order 1 shown in the next example.

Example 2.1 (The AR(1) process)**.** The AR(1) process is a stochastic process,

$$\begin{aligned}
x_1 &\sim \mathcal{N}(0, \sigma_x^2), \\
x_{t+1} &= \phi x_t + \varepsilon_t, \quad t = 2, \ldots, n,
\end{aligned} \tag{2.6}$$

where $|\phi| < 1$ (a stationary requirement) and

$$\varepsilon_t \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_x^2), \quad t = 1, \ldots, n.$$

The joint distribution of $\boldsymbol{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ is Gaussian, $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$, and the conditional dependence structure of this process is given by the graph shown in Figure 2.2. This means that the precision matrix elements satisfy $Q_{ij} = 0$ if $|i-j| > 1$ for $i, j = 1, \ldots, n$. Thus, $\boldsymbol{Q}$ has *bandwidth* 1.

The auto-regressive processes are well-known from time series modelling and it turns out that the precision matrix of an AR(p) process has bandwidth p as is discussed in the next example.

Example 2.2 (The AR(p) process)**.** The AR(p) process with Gaussian inventions can be written as

$$x_{t+1} \mid \boldsymbol{x}_{1:t} \sim \mathcal{N}(\phi_1 x_t + \cdots + \phi_p x_{t-p}, \sigma_x^2), \quad t = 1, \ldots, n, \tag{2.7}$$

where we set $x_0 = x_{-1} = \cdots = x_{-p+1} = 0$, see Rue and Martino (2007, p. 3182). Just as with the AR(1) process the weights $\phi_1, \ldots, \phi_p$ must satisfy certain conditions for the process to be stationary, though these conditions will not be discussed here. The precision matrix for an AR(p) process can be found by expanding the joint distribution of $\boldsymbol{x}$ in the following way

$$\pi(\boldsymbol{x}) = \pi(x_n|\boldsymbol{x}_{1:n-1}) \times \pi(x_{n-1}|\boldsymbol{x}_{1:n-2}) \times \cdots \times \pi(x_2|x_1) \times \pi(x_1). \tag{2.8}$$

By using that $\pi(x_{t+1} \mid \boldsymbol{x}_{1:t}) \propto \exp\big(-\frac{1}{2\sigma_x^2}(x_{t+1} - \phi_1 x_t - \cdots - \phi_p x_{t-p})^2\big)$ and that the joint distribution can be written as

$$\pi(\boldsymbol{x}) \propto \exp\Big(-\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}\Big),$$

we get $Q_{ij} = 0$ for $|i - j| > p$.[2] Thus, an AR(p) process have a precision matrix with bandwidth p. To illustrate how this effects the graph of such a process Figure 2.3 shows the graph of an AR(2) process.

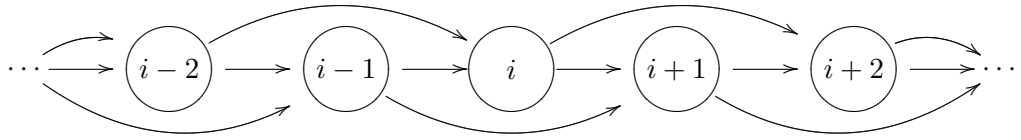


Figure 2.3: The graph $\mathcal{G}$ of an AR(2) process. Since $Q_{ij} = 0$ for $|i - j| > 2$, there are no edges between vertexes that are separated by more than two other vertexes. Also notice how k and $k + 3$ are separated by the set $\{k + 1, k + 2\}$, a feature which is discussed in more detail in Sec. 2.3.3.

Rue and Held (2005) lists three types of typical applications for GMRF models, which are: GMRF models in time or on a line, spatial GMRF models and spatiotemporal GMRF models. The GMRF models in time, such as the auto-regressive processes, usually have precision matrices with a band structure and the following sections includes algorithms specialized for dealing with these types of sparse matrices. More general algorithms for sparse matrices are just briefly mentioned as these are much more complicated.

[2] To see why just compare $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} = \sum_{ij} x_i x_j Q_{ij}$ with the factorization we get from (2.8) and observe that there are no cross terms $x_i x_j$ for $|i - j| > p$, i.e. we must have $Q_{ij} = 0$.

2.3 Linear algebra and GMRFs

Linear algebra problems, like solving $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ for invertible $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{x}, \boldsymbol{b} \in \mathbb{R}^n$, often arise when dealing with Gaussian fields, see Sec. 5 and also Rue and Held (2005, pp. 33-40). Typically these problems involve the precision matrix or the covariance matrix depending on the parameterization used. The issue with linear algebra problems is that general algorithms usually have complexities like $\mathcal{O}(n^3)$ or $\mathcal{O}(n^2)$, where n is the dimension of the problem, which is problematic when n is large. Thus, recognizing and exploiting special structures in the problems we are looking is of great importance as this can reduce the computational workload significantly. First, however, a brief introduction about symmetric positive definite matrices is given.

2.3.1 Symmetric positive definite matrices

As we can see from Definition 2.1 the covariance matrix of a multivariate Gaussian variable is required to be *symmetric positive definite* (SPD). This type of matrices are in a sense very well behaved as they have some particularly favorable properties.

Definition 2.3 (Positive definite matrix)**.** Let $\boldsymbol{A}$ be a (real) square matrix with dimension $\dim(\boldsymbol{A}) = n \times n$. Then $\boldsymbol{A}$ is said to be positive definite (PD), denoted $\boldsymbol{A} > 0$, if it satisfies

$$\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} > 0 \quad \text{for all} \quad \boldsymbol{x} \in \mathbb{R}^n \quad \text{where} \quad \boldsymbol{x} \neq \boldsymbol{0}.$$

A PD matrix satisfying $\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{A}$ is said to be symmetric positive definite.

From this definition it should be clear that a positive definite matrix is always invertible (i.e. the inverse matrix exists) since the null space (or kernel) of a PD matrix only contains the zero vector. The following theorem summarize the most important properties with SPD matrices

Theorem 2.4. *Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be symmetric. Then the following is equivalent*

1. *$\boldsymbol{A}$ is SPD.*
2. *All the eigenvalues of $\boldsymbol{A}$ are real and strictly positive.*
3. *There exists a lower triangular $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ such that $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$. If all the diagonal elements of $\boldsymbol{L}$ are positive then $\boldsymbol{L}$ is unique.*

Proofs and comments for this theorem are available in for instance Strang (2006, pp. 318-320). The lower triangular $\boldsymbol{L}$ is commonly referred to as the Cholesky triangle of $\boldsymbol{A}$ and the factorization $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ is called the Cholesky decomposition of $\boldsymbol{A}$. Other useful properties of positive definite matrices are given in Result 2.5.

Result 2.5. *Let $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$ be PD matrices, $\boldsymbol{C} \in \mathbb{R}^{k\times n}$ be a rank k matrix where $k \leq n$, and α and β be strictly positive scalars. Then the following results apply*

1. *$\boldsymbol{A}^{-1}$ is PD.*

2. $\alpha\boldsymbol{A} + \beta\boldsymbol{B}$ *is PD.*

3. $\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^{\mathrm{T}}$ *is PD.*

4. *Any principal sub matrices of* $\boldsymbol{A}$ *is PD.*

All of these results follow almost immediately from Definition 2.3 and are crucial in for instance Result 2.1.

2.3.2 Numerical methods for SPD systems

The focus in this section will be on computing the Cholesky decomposition of a SPD matrix and usage of the Cholesky triangle. For now let $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ denote a general symmetric positive definite matrix and let its Cholesky decomposition be $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ where $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is lower triangular, i.e. if $\boldsymbol{L} = (L_{ij})$ then $L_{ij} = 0$ for $j > i$. A general algorithm for computing $\boldsymbol{L}$ is provided in Alg. 2.1 which obtained from Rue and Held (2005). Other versions of the Cholesky factorization can be found in Golub and Van Loan (1996). The complexity of this algorithm is, as argued by Rue and

Algorithm 2.1 Cholesky decomposition of a SPD matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$.

```
1: input  Q > 0
2: for j = 1 to n do
3:    v_{j:n} = Q_{j:n,j}
4:    for k = 1 to j − 1 do
5:       v_{j:n} = v_{j:n} − L_{j,k} L_{j:n,j}
6:    end for
7:    L_{j:n,j} = v_{j:n} / sqrt(v_j)
8: end for
9: return  L
```

Held (2005), $\mathcal{O}(n^3/3)$ in the general case. Numerical stability and error bounds when using the Cholesky factorization for computing determinants, solving linear systems and computing the inverse is considered in Martin et al. (1965). They showed that the Cholesky factorization is numerically stable (elements in $\boldsymbol{L}$ are bounded) and that the errors involved in computations using $\boldsymbol{L}$ can be bounded by the condition number (based on the 2-norm) of $\boldsymbol{Q}$,

$$\mathrm{cond}(\boldsymbol{Q}) = \|\boldsymbol{Q}\|_2 \left\|\boldsymbol{Q}^{-1}\right\|_2 = \lambda_{\max}/\lambda_{\min},$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues of $\boldsymbol{Q}$, respectively. Thus unless $\boldsymbol{Q}$ is ill-conditioned ($\mathrm{cond}(\boldsymbol{Q}) \gg 1$) it is safe to use Cholesky triangle.

The reason for computing $\boldsymbol{L}$ is that solving systems of equations and computing determinants are much easier to do for triangular matrices than for general ones. Consider the problem of solving for $\boldsymbol{x}$ in $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{b}$ is known and $\boldsymbol{Q}$ is SPD. By using the factorization $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ the unknown vector can be computed by first solving

$\boldsymbol{L}\boldsymbol{u} = \boldsymbol{b}$ and then solving $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{u}$. Triangular systems of equations are treated in Golub and Van Loan (1996, pp. 88-89) where upper triangular systems are solved by *forward-substitution* and lower triangular systems are solved by *back-substitution*, both of which algorithms have complexity $\mathcal{O}(n^2)$ in general. The inverse of $\boldsymbol{Q}$ can be obtained by solving n systems of equations $\boldsymbol{Q}\boldsymbol{x}^{(i)} = \boldsymbol{e}^{(i)}$, using the above method, where

$$e_j^{(i)} = \begin{cases} 0 & \text{for } i \neq j, \\ 1 & \text{for } i = j. \end{cases}$$

Then $\boldsymbol{Q}^{-1} = (\boldsymbol{x}^{(i)})$ and cost of computing the inverse is $\mathcal{O}(n^3)$. By using that the determinant of a triangular matrix is the product of all the diagonal elements we can find the determinant of $\boldsymbol{Q}$ as

$$\det(\boldsymbol{Q}) = \det(\boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}) = \det(\boldsymbol{L})^2 = \prod_{i=1}^{n} L_{ii}^2.$$

The complexity of this algorithm, assuming $\boldsymbol{L}$ is known, is simply $\mathcal{O}(n)$.

2.3.3 Numerical methods for sparse systems

In Section 2.2 it was briefly mentioned that for a GMRF with a sparse graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ the precision matrix, $\boldsymbol{Q}$, will also be sparse. It turns out that the sparse structure of $\boldsymbol{Q}$ induces a certain sparse structure in the Cholesky triangle $\boldsymbol{L}$ which we can read off from $\mathcal{G}$ using the following theorem and corollary.

Theorem 2.6. *Let $\boldsymbol{x}$ be a GMRF wrt to $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with mean $\boldsymbol{\mu}$ and precision $\boldsymbol{Q} > 0$. Let $\boldsymbol{L}$ be the lower Cholesky triangle of $\boldsymbol{Q}$ and define for $1 \leq i < j \leq n$ the set*

$$F(i, j) = \{i, i+1, \ldots, j-1, j+1, \ldots, n\},$$

which is the future of i except j. Then

$$x_i \perp x_j \mid \boldsymbol{x}_{F(i,j)} \quad \Longleftrightarrow \quad L_{ji} = 0. \tag{2.9}$$

This is Theorem 2.8 in Rue and Held (2005) where a proof is also provided. As pointed out by Rue and Held (2005) this property is really not that helpful since it requires us to verify that $x_i \perp x_j | \boldsymbol{x}_{F(i,j)}$. However, by using the equivalence between the Markov properties (recall the last part of Section 2.1.2) for GMRFs Rue and Held (2005) use the global Markov property to identify zeros in $\boldsymbol{L}$, which is summarized by the following corollary to Thm. 2.6.

Corollary. *If $F(i, j)$ separates $i < j$ in $\mathcal{G}$ then $L_{ji} = 0$.*

The implication of this result is that we only need to compute L_{ji}, $i < j$, when i and j are not separated by $F(i, j)$, which can offer great improvements of efficiency when utilized in Alg. 2.1. As Rue and Held (2005) points out, this result also show that $\boldsymbol{L}$ is more or equally dense than the lower triangular part of $\boldsymbol{Q}$ since $F(i, j)$ never separates

$i < j$ if i and j are neighbors. The difference between the number of non-zero elements in $\boldsymbol{L}$ and $\boldsymbol{Q}$ are referred to in Rue and Held (2005) as the *fill-in*.

For the auto-regressive process of order p, see Example 2.2, the precision matrix was shown to be a band matrix with bandwidth p, thus $Q_{ij} = 0$ for $|i - j| > p$. This result implies that $F(i, j)$ separates $i < j$ in $\mathcal{G}$ if $j - i > p$, or in other words, there are no paths going from i to j not containing an element from $F(i, j)$ (see Fig. 2.3). In turn, this implies that $\boldsymbol{L}$ will have a lower bandwidth p, i.e. $L_{ji} = 0$, $j - i > p$, which is, as mentioned by Rue and Held (2005), the well-known result that the Cholesky triangle inherits the band structure from $\boldsymbol{Q}$, see also Golub and Van Loan (1996, p. 152). The Cholesky factorization algorithm can easily be modified to handle band matrices efficiently. The modified algorithm, which can be found in Rue and Held (2005), is given in Alg. 2.2. This algorithm requires $\mathcal{O}(b_w^2 n)$ flops to factorize a SPD matrix with bandwidth b_w, which is linear in the problem dimension, n. Back- and forward-substitution algorithms for banded triangular systems are given in Golub and Van Loan (1996, p. 153) and their complexities are both $\mathcal{O}(b_w n)$ (bandwidth b_w and dimension n) when solving $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{b}$ and $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b}$.

Algorithm 2.2 Cholesky decomposition of $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$, SPD with bandwidth b_w.

input $\boldsymbol{Q} > 0$ with $\boldsymbol{Q}_{ij} = 0$ if $|i - j| > b_w$
for $j = 1$ to n **do**
  $\lambda = \min(j + b_w, n)$
  $\boldsymbol{v}_{j:\lambda} = \boldsymbol{Q}_{j:\lambda,j}$
  for $k = \max(1, j - b_w)$ to $j - 1$ **do**
    $i = \min(k + b_w, n)$
    $\boldsymbol{v}_{j:i} = \boldsymbol{v}_{j:i} - \boldsymbol{L}_{j,k}\boldsymbol{L}_{j:i,j}$
  end for
  $\boldsymbol{L}_{j:\lambda,j} = \boldsymbol{v}_{j:\lambda}/\sqrt{\boldsymbol{v}_j}$
end for
return $\boldsymbol{L}$

One important point to remember is that the labeling of a GMRF is not unique. Permuting the elements in a random vector $\boldsymbol{x}$ is formally the same as a multiplication by a permutation matrix $\boldsymbol{P}$. It is not difficult to show that (use $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{P} = \boldsymbol{I}$)

$$\mathcal{N}(\boldsymbol{P}\boldsymbol{x} \mid \boldsymbol{P}\boldsymbol{\mu}, (\boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^{\mathrm{T}})^{-1}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1})$$

which means that the permuted vector and the original are identically distributed. For a GMRF with respect to a graph $\mathcal{G}$ a reordering of the random vector simply corresponds to a relabeling of the vertexes. The Cholesky triangle of the precision matrix of a GMRF does, however, depend on the labeling (or ordering) of the vertexes, especially with regards to the sparseness. Rue and Held (2005) presents two ideas for dealing with GMRFs with more general precision matrices, both based on reordering of $\boldsymbol{x}$ and factorization of the permuted precision matrix.

The first method, referred to as *bandwidth reduction*, reorders the elements in $\boldsymbol{x}$ with the goal to reduce the bandwidth of the resulting precision matrix, which can then be

factorized using Alg. 2.2. Note that in the case where $\mathcal{G}$ contains global vertexes, for instance if $i \in \mathcal{V}$ has $\text{ne}(i) = \mathcal{V} \setminus \{i\}$, then the precision matrix for the reordered GMRF will always have a large bandwidth (worst-case being $b_w = n - 1$) and a factorization using Alg. 2.2 requires $\mathcal{O}(n^3)$ flops as in the general case. The second method, called *nested dissection*, reorders the vertexes a bit differently by focusing on reducing the fill-in and not the bandwidth (Rue and Held, 2005). The basic idea is that by finding a set of vertexes which separates parts of the graph and ordering so that the separating vertexes have a higher index than the vertexes in the separated sets will reduce the fill-in. This is illustrated quite well in an example in Rue and Held (2005, p. 49).

One important detail to notice with the re-ordering schemes is that the permutations are done with regards to the sparse structure of the precision matrix and not the value of the non-zero elements. Thus, if a suitable permutation has been obtained, this can be used in the factorizations of matrices with the same sparse structure, i.e. we only compute the permutation once.

A point made by Rue and Held (2005) is that sparse matrix factorization routines based on reordering schemes are far from trivial implement, though already implemented libraries are available (Rue and Held, 2005, pp. 52-53).

2.4 Solving for marginal variances

As seen from the previous sections using the precision matrix, $\boldsymbol{Q}$, rather than the covariance matrix, $\boldsymbol{\Sigma}$, in the parameterization of the distribution of a GMRF can be quite beneficial. However, some of the elements in the covariance matrix may be of interest in the analysis, such as the marginal variances which are found on the diagonal, e.g. $\Sigma_{ii} = \text{var}(x_i)$. One possible option for obtaining certain elements of $\boldsymbol{\Sigma}$ is of course to compute $\boldsymbol{Q}^{-1}$. Though, this is un-necessarily expensive to do if one, for instance, only want $\text{diag}(\boldsymbol{\Sigma})$. This section will present a faster way of computing some of the elements of $\boldsymbol{\Sigma}$ by avoiding computations of other elements.

In the following let $\boldsymbol{x}$ be a zero mean GMRF with respect to a graph $\mathcal{G}$ and let $\boldsymbol{Q} > 0$ denote its precision matrix. The Cholesky factorization of $\boldsymbol{Q}$ is given as $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$. The idea is based on the following result, see Rue and Martino (2007, p. 3180),

Result 2.7. *Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}}$ (SPD). Then the solution of $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{z}$ has a Gaussian distribution with precision matrix $\boldsymbol{Q}$, i.e. $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$.*

This result can be verified by using Property 1 in Result 2.1 and actually provides a way of sampling $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$ by first sampling a vector of n independent standard normals, $\boldsymbol{z}$, and then solving $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{z}$. Writing out the n equations in this system and isolating for x_i yields

$$x_i = \frac{z_i}{L_{ii}} - \frac{1}{L_{ii}} \sum_{k+i}^{n} L_{ki} x_k, \quad i = n, \ldots, 1. \tag{2.10}$$

Now the trick is to multiply these equations by x_j, $j \geq i$, and take the expectation, $\text{E}(\cdot)$, on both sides. By recalling that $\boldsymbol{x}$ has zero mean and noting that x_j only depends on

z_k for $k \geq j$ we get

$$\Sigma_{ij} = \frac{\delta_{ij}}{L_{ii}^2} - \frac{1}{L_{ii}} \sum_{k=i+1}^{n} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1, \tag{2.11}$$

where δ_{ij} is the Kronecker delta ($\delta_{ij} = 0$ for $i \neq j$ and $\delta_{ii} = 1$). Using Equation (2.11) to compute Σ_{ij} for $j = n, \ldots, i$, and $i = n, \ldots, 1$, is the same as computing the whole of $\mathbf{\Sigma}$, but it turns out that not all the Σ_{ij}'s have to be computed if we are only interested in diag($\mathbf{\Sigma}$).

First we recall from Section 2.3.3 that $L_{ki} = 0$ for $k > i$ if $F(i, k)$ separates i and k in $\mathcal{G}$. Thus, in (2.11) the sum only have to include terms where L_{ki} are not known to be zero. For fixed i, the set of indexes k for which L_{ki} are not known to be zero is defined as

$$\mathcal{I}(i) = \{k \mid k > i,\ F(i, k) \text{ does not separate } i \text{ and } k \text{ in } \mathcal{G}\}.$$

Summing over this set in (2.11) gives us

$$\Sigma_{ij} = \frac{\delta_{ij}}{L_{ii}^2} - \frac{1}{L_{ii}} \sum_{k \in \mathcal{I}(i)} L_{ki}\Sigma_{kj}, \quad j \geq i,\ i = n, \ldots, 1. \tag{2.12}$$

The elements of interest are the marginal variances $\Sigma_{11}, \ldots, \Sigma_{nn}$ and the goal is to compute these by using Eq. (2.12) as a recursion while not computing any unnecessary off-diagonal elements, Σ_{ij}, $i \neq j$, in the process. It is obvious that in order for Eq. (2.12) to define a recursion Σ_{kj} (or Σ_{jk}), $k \in \mathcal{I}(i)$, must be computed before Σ_{ij} for fixed i, j s.t. $j \geq i$. In other words if $\mathcal{J}$ defines the set of (unordered) pairs of indexes $\{i, j\}$ of the elements Σ_{ij} that must be computed in the recursion then $\mathcal{J}$ must contain $\{i, i\}$ for $1 \leq i \leq n$, but also satisfy

$$j \geq i, \quad \{i, j\} \in \mathcal{J} \quad \Longrightarrow \quad \{k, j\} \in \mathcal{J} \quad \text{for all } k \in \mathcal{I}(i),$$

see Rue and Martino (2007, p. 3181). The size of $\mathcal{J}$ determines how efficient the recursions can be solved and of course we want $|\mathcal{J}|$ to be as small as possible. Rue and Martino (2007) shows that by taking

$$\mathcal{J} = \{\{i, j\} \in \mathcal{V} \times \mathcal{V} \mid j \geq i,\ F(i, j) \text{ does not separate } i \text{ and } j \text{ in } \mathcal{G}\},$$

the recursions are *solvable* (both of the above conditions are satisfied) and $\mathcal{J}$ is minimal. By taking $j \in \mathcal{I}(i) \cup \{i\}$ in decreasing order for $i = n, \ldots, 1$ and computing Σ_{ij} from Eq. (2.12) defines solvable recursions which do not compute any obsolete Σ_{ij}'s.

In particular if $\boldsymbol{Q}$ is a band matrix with bandwidth b_w the set of indexes of computed L_{ji}, for fixed i where $j \geq i$, is $\mathcal{I}(i) = \{i + 1, \ldots, \min(i + b_w, n)\}$, see Sec. 2.3.3. The recursions for the case when $\boldsymbol{Q}$ is a band matrix is given in Algorithm 2.3 and the complexity is $\mathcal{O}(b_w^2 n)$ (Rue and Martino, 2007). Beware that this algorithm implicitly use the symmetric property of $\mathbf{\Sigma}$, i.e. $\Sigma_{kj} = \Sigma_{jk}$.

Algorithm 2.3 Solving for $\mathrm{diag}(\boldsymbol{\Sigma})$ where $\boldsymbol{Q} = \boldsymbol{\Sigma}^{-1}$ is a (SPD) band matrix with bandwidth b_w.

input $\boldsymbol{L}$ (where $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}} > 0$ symmetric)
$\Sigma := 0$
for $i = n$ to 1 **do**
  $\lambda = \min(i + b_w, n)$
  for $j = \lambda$ to i **do**
    $\Sigma_{ij} = \delta_{ij}/L_{ii}^2 - (\sum_{k=i+1}^{\lambda} L_{ki}\Sigma_{kj})/L_{ii}$
  end for
end for
return $\mathrm{diag}(\boldsymbol{\Sigma}) = (\Sigma_{11}, \ldots, \Sigma_{nn})^{\mathrm{T}}$ (or $\boldsymbol{\Sigma}_{\mathcal{J}}$)

3 Latent Gaussian models

Latent Gaussian models are examples of hierarchical Bayesian models where observations are assumed to depend on a latent field for which a Gaussian prior is used. In Rue et al. (2009) this framework is used with so-called *structured additive models* and the result is a powerful and flexible modelling tool.

In structured additive models the observations $\boldsymbol{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$ are assumed to have a distribution belonging to the exponential family. Like with the generalized linear models, the mean of observation y_i is linked to a linear predictor through a monotonic link function $g(\cdot)$, i.e. $g(\mathrm{E}\, y_i) = \eta_i$ where

$$\eta_i = \alpha + \sum_{j=1}^{n_f} f^{(j)}(u_{ij}) + \sum_{j=1}^{n_\beta} \beta_k z_{ki} + \varepsilon_i, \quad i = 1, \ldots, N. \tag{3.1}$$

In Eq. (3.1) α is a common intercept term, β_k is the weight of the covariate z_{ki}, $f^{(j)}(\cdot)$ is an unknown function of covariate u_{ij} and ε_i is the noise associated with measurement i. The difference between the structured additive models and the generalized linear models is the added flexibility of letting η_i depend on the covariates u_{ij} through unspecified functions, opening these models for a wealth of applications. Examples of such applications can be found in Rue et al. (2009, pp. 321-322) and in Section 6.2. The latent field in the Bayesian model is taken to be all the unobserved terms in (3.1), i.e. α, $\{f^{(j)}(u_{ij})\}$, $\{\beta_k\}$ and $\{\eta_i\}$, and we denote this field by $\boldsymbol{x}$. The prior for $\boldsymbol{x}$ is as mentioned a (zero mean) Gaussian distribution which depend on some hyperparameters $\boldsymbol{\theta}_1$. The observational distribution can also depend on parameters, denoted by $\boldsymbol{\theta}_2$, which are not a part of the Gaussian latent field, and the observations are assumed to be conditionally independent given $\boldsymbol{x}$ and $\boldsymbol{\theta}_2$. Parameters which are not in the latent field are collected in the vector $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^{\mathrm{T}}, \boldsymbol{\theta}_2^{\mathrm{T}})^{\mathrm{T}}$ and according to Bayesian formalism a prior distribution is also chosen

for $\boldsymbol{\theta}$. The full model can then be formulated as,

$$\begin{aligned} \boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod_{i=1}^{N} \pi(y_i \mid \eta_i, \boldsymbol{\theta}_2), \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{W}^{-1}(\boldsymbol{\theta}_1)), \\ \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}). \end{aligned} \tag{3.2}$$

Note that the Gaussian prior for the latent field has been parameterized using the precision matrix rather than the covariance matrix, i.e. $\text{prec}(\boldsymbol{x}|\boldsymbol{\theta}) = \boldsymbol{W}(\boldsymbol{\theta})$. This is a practical formulation when applying an additional assumption to the latent field, namely that it is a Gaussian Markov random field which, as discussed in Section 2.2, implies a certain sparse structure in $\boldsymbol{W}$. As the latent field often is of large dimension,

$$\dim(\boldsymbol{x}) = n = N + n_f + n_\beta \sim 10^2\text{–}10^5,$$

see Rue et al. (2009), assuming that $\boldsymbol{x}$ is a GMRF is quite useful as the approximate inference method presented in Sec. 4 inevitably involves factorizing $\boldsymbol{W}$, which can then be done by the methods described in Sec. 2.3.3.

In the following sections treating approximate Bayesian analysis the modelling will be left more in the background and we will assume that the dimension of the latent field is the same as the number of observations to simplify notation. Also, the dependence of $\boldsymbol{y}$ on the hyperparameters will be suppressed. The simplified model is then given as

$$\begin{aligned} \boldsymbol{y} \mid \boldsymbol{x} &\sim \prod_{i=1}^{n} \pi(y_i \mid x_i), \\ \boldsymbol{x} \mid \boldsymbol{\theta} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{W}^{-1}(\boldsymbol{\theta})), \\ \boldsymbol{\theta} &\sim \pi(\boldsymbol{\theta}). \end{aligned} \tag{3.3}$$

Examples 3.1-3.3 show three possible scenarios where (3.3) (and (3.2)) can be applied.

Example 3.1 (Real valued data)**.** For real-valued data, i.e. $y_i \in \mathbb{R}$ for $i = 1, \ldots, n$, a possible choice for the conditional distribution for the observations is simply using the Gaussian distribution with (assumed) known variance σ_y^2 and mean equal to the latent field value,

$$y_i \mid x_i \sim \mathcal{N}(x_i, \sigma_y^2), \quad i = 1, \ldots, n. \tag{3.4}$$

Example 3.2 (Counting data)**.** For observations taking values in $\mathbb{N}$, e.g. counting data, a possible conditional distribution is the Poisson distribution

$$y_i \mid x_i \sim \mathit{Poisson}(\mathrm{e}^{x_i}), \quad i = 1, \ldots, n. \tag{3.5}$$

Example 3.3 (Binary data)**.** For binary data, i.e. $y_i \in \{0, 1\}$, the conditional distribution is usually defined with success probability given by a *sigmoid function*, see Rasmussen and Williams (2006), $\sigma : \mathbb{R} \to [0, 1]$ so that

$$y_i \mid x_i \sim \mathit{Bin}(\sigma(x_i)) \quad i = 1, \ldots, n. \tag{3.6}$$

Possible choices of $\sigma(\cdot)$ include any cumulative distribution function, like the Gaussian,

$$\sigma(x) = \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-z^2/2} \mathrm{d}z, \quad x \in \mathbb{R}, \tag{3.7}$$

or for instance the inverse logistic function

$$\sigma(x) = \mathrm{logit}^{-1}(x) = \frac{1}{1 + \mathrm{e}^{-x}}, \quad x \in \mathbb{R}. \tag{3.8}$$

4 Approximate Bayesian Inference

In Bayesian analysis the idea is to do inference about unobserved elements in a model (e.g. a latent field and hyperparameters) using posterior distributions, that is distributions where the observations are fixed. In models such as (3.3) posterior distributions of interest are for instance the joint distribution of the hyperparameters, $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ and marginal distributions for the latent field elements, e.g. $\pi(x_i|\boldsymbol{y})$. The posterior distributions in a Bayesian model can be written in terms of the data likelihoods (data distributions as functions of the parameters) and prior distributions by using Bayes' rule, say if $\boldsymbol{y}$ are observations with known distribution $\pi(\boldsymbol{y}|\boldsymbol{\theta})$ for parameters $\boldsymbol{\theta}$, then

$$\pi(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{\pi(\boldsymbol{\theta}, \boldsymbol{y})}{\pi(\boldsymbol{y})} \propto \pi(\boldsymbol{y} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta}), \tag{4.1}$$

where $\pi(\boldsymbol{\theta})$ is a prior. By normalizing the latter expression the posterior for $\boldsymbol{\theta}$ is determined and can be used for estimation, computing expectations of functions of $\boldsymbol{\theta}$ and making probability statements (e.g. to find confidence intervals). The problem in Bayesian analysis is that one often encounters integrals (or sums) which cannot be computed analytically. Also, different deterministic numerical integration schemes are infeasible options due to computational complexities that are often exponential in the parameter dimension. In general Bayesian models are most commonly studied using simulation based integration methods such as Markov chain Monte Carlo (MCMC) methods (Gamerman and Lopes, 2006). The strength of MCMC methods for doing Bayesian analysis is that these methods can be applied (relatively) easily to quite complex models and the error associated with methods can be made arbitrary small by doing long simulation runs. The main problem with MCMC is related to their computer intensive nature as running simulations for large models can require extensive amount of time (Rue et al., 2009).

An alternative to MCMC is to use *approximate Bayesian analysis* as discussed by Rue et al. (2009). The basic idea in this approach is to combine parametric and numerical approximations to different distributions in a way that eliminates the need of computing high dimensional integrals by MCMC. Rue et al. (2009) argues that approximate analysis is particularly well suited for latent Gaussian models as will be motivated here. To illustrate the methodology consider the inference problem of computing the expectation of $h(\boldsymbol{\theta})$, for a well-behaved function $h(\cdot)$, under the posterior distribution of $\boldsymbol{\theta}$ from the

model in (3.3). That is, computing

$$\mathrm{E}(h(\boldsymbol{\theta}) \mid \boldsymbol{y}) = \int h(\boldsymbol{\theta})\pi(\boldsymbol{\theta} \mid \boldsymbol{y})\mathrm{d}\boldsymbol{\theta}. \tag{4.2}$$

This is a difficult task for two reasons, $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ is not, or may not be, known parametrically and even if it were the integral could be impossible to compute analytically. A remedy for the latter problem is to use numerical integration, and if $m = \dim(\boldsymbol{\theta})$ is not too large (Rue et al. (2009) suggest $m \leq 6$ as a reasonable limitation) this can be done by a deterministic approximation, commonly referred to as a quadrature integration scheme,

$$\int h(\boldsymbol{\theta})\pi(\boldsymbol{\theta} \mid \boldsymbol{y})\mathrm{d}\boldsymbol{\theta} \approx \sum_k h(\boldsymbol{\theta}_k)\pi(\boldsymbol{\theta}_k \mid \boldsymbol{y})\Delta_k. \tag{4.3}$$

In this approximation the collection $\{\boldsymbol{\theta}_k\}$ forms a grid of values to interpolate the integrand function in the sample space of $\boldsymbol{\theta}$, and Δ_k is the area weight associated with grid point $\boldsymbol{\theta}_k$. Now, the functional form of $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ is, as mentioned, not necessarily known so in general it cannot be evaluated in (4.3). Though we can make pointwise approximations $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ for fixed $\boldsymbol{\theta}$. The idea is to consider the factorization in Eq. (4.1) and focus on the factor $\pi(\boldsymbol{y}|\boldsymbol{\theta})$, referred to as the *evidence*. By using the total law of probability we find that

$$\pi(\boldsymbol{y} \mid \boldsymbol{\theta}) = \int \pi(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta})\mathrm{d}\boldsymbol{x}. \tag{4.4}$$

Note that $\pi(\boldsymbol{y}|\boldsymbol{\theta})$ is viewed as a function of $\boldsymbol{\theta}$, as in (4.1), so $\boldsymbol{y}$ is fixed. Contrary to the posterior distribution of hyperparameters, the integrand in (4.4) is in general known parametrically from the factorization $\pi(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta}) = \pi(\boldsymbol{y}|\boldsymbol{x})\pi(\boldsymbol{x}|\boldsymbol{\theta})$, where both factors are given in the model, see Eq. (3.3). The problem with (4.4) is the large dimension of the latent field, so if the integral cannot be computed analytically there are few other options than Monte Carlo integration. However, for latent Gaussian models there is one possible bypass of simulation based integration which is to fix $\boldsymbol{\theta}$ and approximate the integrand in (4.4) by a function of $\boldsymbol{x}$ which we know how to integrate analytically, i.e.

$$\pi(\boldsymbol{y} \mid \boldsymbol{\theta}) \approx I(\boldsymbol{y}, \boldsymbol{\theta}) = \int f(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta})\mathrm{d}\boldsymbol{x}, \quad \text{where} \quad \pi(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta}) \approx f(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}).$$

The natural choice of function to approximate $\pi(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta})$ turns out to be an unnormalized Gaussian distribution in $\boldsymbol{x}$, so that

$$f(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}) = \tilde{\pi}(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta}) = Z\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}). \tag{4.5}$$

By using this approximation the integration is trivial since a Gaussian distribution normalizes (i.e. integrates to 1), thus

$$\tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta}) = \int \tilde{\pi}(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta})\mathrm{d}\boldsymbol{x} = Z.$$

In order to see why it is natural to approximate $\pi(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta})$ by an unnormalized Gaussian distribution we consider a factorization obtained by using Bayes' rule,

$$\pi(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta}) = \pi(\boldsymbol{y} \mid \boldsymbol{\theta})\pi(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}). \tag{4.6}$$

By comparing factors in (4.5) and (4.6) it is clear that the approximation has the following interpretation,

$$\tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta}) = Z \quad \text{and} \quad \tilde{\pi}(\boldsymbol{x} \mid \boldsymbol{y}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}),$$

thus the posterior of latent field conditioned on the hyperparameters is approximated parametrically by a proper Gaussian distribution and $\tilde{\pi}(\boldsymbol{y}|\boldsymbol{\theta})$ is simply 'what is left' (or what does not depend on $\boldsymbol{x}$). As we can see it is important that the Gaussian approximation captures as much of the probability mass of $\pi(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\theta})$ as possible since any deviance will contribute to an error in $\tilde{\pi}(\boldsymbol{y}|\boldsymbol{\theta})$. Due to the Gaussian prior assigned to $\boldsymbol{x}|\boldsymbol{\theta}$ in (3.3) it is natural to assume that $\pi(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\theta})$ will be 'close' to Gaussian if $\boldsymbol{y}$ is not too informative, and this is usually the case for real problems and data sets, according to Rue et al. (2009). It is this property, combined with the availability of fast and accurate methods for computing the unnormalized Gaussian approximation to $\pi(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta})$ (see Sec. 5), that makes this approximation scheme so appealing for use with latent Gaussian models such as (3.3).

It is important to stress that $\tilde{\pi}(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta})$ is a parametric approximation as a function of $\boldsymbol{x}$, so this can only be used for pointwise approximations of $\pi(\boldsymbol{y}|\boldsymbol{\theta})$. However, this is exactly what we need in (4.3) as the numerical integration only involves interpolated values of $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ for $\boldsymbol{\theta} \in \{\boldsymbol{\theta}_k\}$. These are then replaced by

$$\tilde{\pi}(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{1}{K}\tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta}), \quad \boldsymbol{\theta} \in \{\boldsymbol{\theta}_k\},$$

where $K = \tilde{\pi}(\boldsymbol{y})$ is the approximated normalizing constant of $\pi(\boldsymbol{\theta}|\boldsymbol{y})$. It is very important to compute K so that $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ is properly scaled, and this is done using the same integration technique outlined in (4.3), i.e.

$$K = \sum_k \tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta}_k)\pi(\boldsymbol{\theta}_k)\Delta_k.$$

Now, just to recapture the process, an approximation to $\mathrm{E}(h(\boldsymbol{\theta})|\boldsymbol{y})$, denoted $\tilde{\mathrm{E}}(h(\boldsymbol{\theta})|\boldsymbol{y})$, is computed using the following scheme:

$$\tilde{\mathrm{E}}(h(\boldsymbol{\theta})|\boldsymbol{y}) = \sum_k h(\boldsymbol{\theta}_k)\tilde{\pi}(\boldsymbol{\theta}_k \mid \boldsymbol{y})\Delta_k, \tag{4.7}$$

where

$$\tilde{\pi}(\boldsymbol{\theta}_k \mid \boldsymbol{y}) = \frac{1}{K}\tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta}_k)\pi(\boldsymbol{\theta}_k), \tag{4.8}$$

and $\tilde{\pi}(\boldsymbol{y}|\boldsymbol{\theta}_k)$ is the constant Z in the unnormalized Gaussian approximation

$$\tilde{\pi}(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\theta}_k) = Z\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}).$$

The normalization constant K and the grid $\{\boldsymbol{\theta}_k\}$ must of course be computed in advance.

There are still a few loose ends which need to be commented on. First of all, how to make the unnormalized Gaussian approximation to $\pi(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{\theta})$ has not been discussed,

however this is treated in Section 5. Secondly, we have not seen how to make a grid of values $\{\boldsymbol{\theta}_k\}$ to interpolate $h(\boldsymbol{\theta})\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$. A method of finding such a gird is discussed in Rue et al. (2009) where the key point is to explore $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ in a clever way to determine where it is significantly different from zero. This method will not be discussed any further here, though we can note that once a grid has been determined it can be used for all inference tasks involving the integration of $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ over the sample space (or parts of the sample space) of $\boldsymbol{\theta}$. Particularly, if we can make pointwise approximations to $\pi(x_i|\boldsymbol{y},\boldsymbol{\theta})$, denoted $\tilde{\pi}(x_i|\boldsymbol{y},\boldsymbol{\theta})$, we can approximate the posterior marginal of x_i by

$$\pi(x_i \mid \boldsymbol{y}) = \int \pi(x_i \mid \boldsymbol{y},\boldsymbol{\theta})\pi(\boldsymbol{\theta}|\boldsymbol{y})\mathrm{d}\boldsymbol{\theta} \approx \sum_k \tilde{\pi}(x_i \mid \boldsymbol{y},\boldsymbol{\theta}_k)\tilde{\pi}(\boldsymbol{\theta}_k \mid \boldsymbol{y})\Delta_k,$$

where $\{\boldsymbol{\theta}_k\}$ can be taken to be the same as in Eq. (4.7). Making pointwise approximations $\tilde{\pi}(x_i|\boldsymbol{y},\boldsymbol{\theta})$ is treated in detail in Rue et al. (2009) and Cseke and Heskes (2010).

4.1 The Laplace approximation

One of the key points in the previous section was the approximation of $\pi(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta})$ by an unnormalized Gaussian distribution (when $\boldsymbol{y}$ and $\boldsymbol{\theta}$ are fixed), which can easily be integrated analytically over $\mathbb{R}^n$. As was shown, this yields the approximated $\tilde{\pi}(\boldsymbol{y}|\boldsymbol{\theta})$ which can be used in

$$\tilde{\pi}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \tilde{\pi}(\boldsymbol{y} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta}),$$

the (unnormalized) approximated posterior of $\boldsymbol{\theta}$. In Rue et al. (2009) we find that the posterior of the hyperparameters is approximated by

$$\tilde{\pi}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \left.\frac{\pi(\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{y})}{\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{y})}\right|_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{\theta})}, \tag{4.9}$$

where $\boldsymbol{x}^*(\theta) = \mathrm{argmax}_{\boldsymbol{x}}\, \pi(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$ and $\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$ is the Gaussian approximation obtained by matching the mode and curvature at the mode of $\pi(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$, see also Sec. 5.1. As pointed out by Rue et al. (2009) this is equivalent to the Laplace approximation suggested by Tierney and Kadane (1986).[3] The expression in (4.9) looks deviously similar to the exact identity

$$\pi(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \frac{\pi(\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{y})}{\pi(\boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{y})}, \tag{4.10}$$

which is valid for any $\boldsymbol{x}$. Authors in the machine learning community, see for instance Minka (2001) and Rasmussen and Williams (2006), have reported that improvements to the Gaussian approximation in (4.9) is possible by applying a different approximation scheme, the expectation propagation algorithm, see Sec. 5.2. A tempting approach

[3] Actually, the Laplace approximation used by Tierney and Kadane (1986) is an approximation to the *normalized* $\pi(\boldsymbol{\theta}|\boldsymbol{y})$, as opposed to (4.9) where the normalizing constant is found by numerical integration. The name 'Laplace approximation' stems from the use of Laplace's method for integrals, a technique for analytically approximating integrals by using an asymptotic expansion of the integrand, see Tierney and Kadane (1986).

for improving $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$, justified by the identity in (4.10), is to simply replace the Gaussian approximation used in (4.9) by the improved Gaussian approximation, denoted $\tilde{\pi}_{\text{EP}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$, and evaluate the whole expression in the mode of $\tilde{\pi}_{\text{EP}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$, i.e

$$\tilde{\pi}_{\text{EP}}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \left.\frac{\pi(\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{y})}{\tilde{\pi}_{\text{EP}}(\boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{y})}\right|_{\boldsymbol{x}=\boldsymbol{\mu}(\boldsymbol{\theta})}. \tag{4.11}$$

The idea is that since $\tilde{\pi}_{\text{EP}}(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$ is an improved approximation to $\pi(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{y})$, then $\tilde{\pi}_{\text{EP}}(\boldsymbol{\theta}|\boldsymbol{y})$ should be an improved approximation to $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ as well.

It may be difficult to see at first glance, but there is a fundamental difference between the approximation approach used in (4.9) and the one used in (4.11) other than the way the Gaussian approximation is computed. Even though both approaches seem to be based on (4.10), this is only the case for the expectation propagation approach, the Laplace approximation is actually based on the principle used in the previous section (i.e. approximating $\tilde{\pi}(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta})$ by an unnormalized Gaussian distribution). The result, as will be shown in an example in Sec. 6.2 is devastating for the expectation propagation based approximation given in (4.11). However, as we will see in Sec. 5.2 the EP algorithm also computes the approximation $\tilde{\pi}(\boldsymbol{y}|\boldsymbol{\theta})$ and can therefore be applied in the approximate Bayesian analysis in the same way as the Laplace approximation.

5 Approximation Methods

The approximate Bayesian analysis scheme discussed in Sec. 4 relies heavily on the point that for latent Gaussian models the distribution $\pi(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta})$ ($\boldsymbol{y}$ fixed) can be approximated well by a function $\tilde{\pi}(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta}) = Z\mathcal{N}(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{Q}^{-1})$. As was pointed out the constant Z can be interpreted as an approximation to $\pi(\boldsymbol{y}|\boldsymbol{\theta})$ and the Gaussian distribution can be seen as an approximation of $\pi(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\theta})$. The key point, as was also mentioned, is that $\pi(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\theta})$ will be close to Gaussian due to the Gaussian prior chosen for $\boldsymbol{x}|\boldsymbol{\theta}$. All of this would of course be of no value to us if there where no methods available for computing the approximation $\tilde{\pi}(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta})$ in a fast and accurate way, but luckily there are. This section will present two possible options for computing $\tilde{\pi}(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{\theta})$, one based on a asymptotic expansion and one based on iterative optimization.

In order to reduce the clutter in this presentation a simplified notation is introduced. The data likelihoods, $\pi(y_i|x_i)$, will throughout this section be viewed as functions of the latent field x_i and denoted by $t_i(x_i)$. Furthermore the following simplifications will be used

$$\begin{aligned} \pi(\boldsymbol{x},\boldsymbol{y} \mid \boldsymbol{\theta}) &= \pi(\boldsymbol{x}), \\ \pi(\boldsymbol{x} \mid \boldsymbol{y},\boldsymbol{\theta}) &= \pi_{\text{post}}(\boldsymbol{x}), \\ \pi(\boldsymbol{y} \mid \boldsymbol{\theta}) &= Z_{\boldsymbol{y}|\boldsymbol{\theta}}. \end{aligned} \tag{5.1}$$

All approximations are identified by a tilde, e.g. $\pi(\boldsymbol{x}) \approx \tilde{\pi}(\boldsymbol{x})$ and $t_i(x_i) \approx \tilde{t}_i(x_i)$, and the (proper) Gaussian approximation will be denoted $\tilde{\pi}_{\text{G}}(\boldsymbol{x})$, thus

$$\tilde{\pi}(\boldsymbol{x}) = \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu},\boldsymbol{Q}^{-1}) = \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}\tilde{\pi}_{\text{G}}(\boldsymbol{x}). \tag{5.2}$$

5.1 Analytical approximation

The analytical approximation method for computing $\tilde{\pi}(\boldsymbol{x})$ is a well familiar approximation technique used for instance by Tierney and Kadane (1986) in their Laplace approximation (see also Sec. 4.1) and by Rasmussen and Williams (2006) for computing a Gaussian approximation $\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x})$.[4] The basic idea is quite simple, approximate $\log \pi(\boldsymbol{x})$ by a second order Taylor expansion around its mode. If $\boldsymbol{x}^*$ denote the (highest) mode of $\pi(\boldsymbol{x})$, i.e. $\boldsymbol{x}^* = \operatorname{argmax}_{\boldsymbol{x}} \pi(\boldsymbol{x})$, then the analytical approximation is defined as

$$\log \tilde{\pi}(\boldsymbol{x}) = \log \pi(\boldsymbol{x}^*) - \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^*)^{\mathrm{T}} \boldsymbol{Q} (\boldsymbol{x} - \boldsymbol{x}^*), \tag{5.3}$$

where $\boldsymbol{Q} = -\nabla^2 \log \pi(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^*}$. Note that $\nabla \log \pi(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^*} = \boldsymbol{0}$ since $\boldsymbol{x}^*$ is a stationary point of $\pi(\boldsymbol{x})$. Factorizing the expression in (5.3) so that $\tilde{\pi}(\boldsymbol{x})$ is the product of a constant and a proper Gaussian density in $\boldsymbol{x}$ gives us the following approximations

$$\begin{aligned} \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}} &= \frac{\pi(\boldsymbol{x}^*)}{(2\pi)^{-n/2} \det(\boldsymbol{Q})}, \\ \tilde{\pi}_{\mathrm{G}}(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{x}^*, \boldsymbol{Q}^{-1}). \end{aligned} \tag{5.4}$$

The denominator in $\tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}$ can be identified as the Gaussian approximation evaluated at the mode, and thus the approximated posterior distribution of the hyperparameters can be written

$$\tilde{\pi}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}} \pi(\boldsymbol{\theta}) = \frac{\pi(\boldsymbol{x}^*)\pi(\boldsymbol{\theta})}{\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x}^*)},$$

This expression is equivalent to Eq. (4.9), the Laplace approximation suggested by Tierney and Kadane (1986).

As we can see from (5.4) the Gaussian approximation resulting from this approach has mean equal to the mode of $\pi(\boldsymbol{x})$ and precision equal to the negative Hessian matrix evaluated at the mode.[5] For this to be a valid Gaussian density the precision matrix must be positive definite, however this is not a problem as long as $\pi(\boldsymbol{x})$ is twice differentiable at $\boldsymbol{x}^*$ as the Hessian is always negative definite (i.e. the negative Hessian is PD) when evaluated at a maxima (a generalization of the second derivative test in calculus). A particularly useful result with the precision matrix of $\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x})$ is that the sparse structure of the prior precision $\boldsymbol{W}$ is inherited in $\boldsymbol{Q}$ (see Eq. (5.6)). This means that if $\boldsymbol{x}$ is a GMRF with respect to a graph $\mathcal{G}$ then the Gaussian approximation will define a GMRF with respect to the same graph. It is important that the Gaussian approximation corresponds well with the conditional posterior for the latent field in order for the approximate inference to be accurate. Before discussing the implementation we consider two cases

[4] In machine learning literature, e.g. Rasmussen and Williams (2006), $\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x})$ is often referred to as the Laplace approximation. This must not be confused with the Laplace approximation suggested by Tierney and Kadane (1986) where the Gaussian approximation is integrated out to obtain approximations such as $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$.

[5] The Hessian of a scalar function taking a vector argument is defined as the (symmetric) matrix of second derivatives, $\nabla^2 h(\boldsymbol{t}) = \left(\frac{\partial^2 h(\boldsymbol{t})}{\partial t_i \partial t_j} \right)$.

where the analytical approximation is not the best choice, one where it fails and one where it can be inaccurate.

It is not difficult to construct examples where the analytical approach runs in to problems, though it involves relaxing the assumption that $t_i(x_i)$ is a member of the exponential family. A prime example, considered for instance by Cseke and Heskes (2010), is letting $t_i(x_i)$ be the Laplace distribution

$$t_i(x_i) = \frac{\lambda}{2} \mathrm{e}^{-|x_i - y_i|/\lambda}, \quad y_i \in \mathbb{R},$$

i.e. a symmetric exponential distribution defined on the real line. This distribution is continuously differentiable with respect to x_i everywhere *except* at the mode, y_i. As a result, the Hessian

$$\nabla^2 \log \pi(\boldsymbol{x}) = \nabla^2 \log \pi_0(\boldsymbol{x}) + \sum_{i=1}^{n} \frac{\partial^2 \log t_i(x_i)}{\partial x_i^2}$$

does not exists if any of the elements in the mode $\boldsymbol{x}^*$ satisfy $x_i^* = y_i$.

An important point to keep in mind when using a Gaussian distribution as an approximation is that a Gaussian distribution has certain properties which, to some extent, should be present in the approximated distribution as well. For instance, a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is symmetric around the axes determined by the eigenvectors of $\boldsymbol{\Sigma}$ when using $\boldsymbol{\mu}$ as a reference point (Johnson and Wichern, 2007). Even though the true distribution we are trying to approximate, $\pi_{\text{post}}(\boldsymbol{x})$, is assumed to be almost Gaussian it does not have to be symmetric, it can for instance be slightly skewed. Since an approximation is trying to capture as much of the probability mass of $\pi_{\text{post}}(\boldsymbol{x})$ as possible, fitting a Gaussian with mean equal to the mode may not be a good idea if the true distribution is skewed. To illustrate this, consider Figure 5.1 where a Gaussian distribution has been fitted to approximate a skew-normal distribution using the analytical method. As we can see from this figure the Gaussian distribution has too little probability mass on one side and too much on the opposite side. A better Gaussian approximation in this case can be obtained by using moment matching which is the basis for the expectation propagation algorithm considered in Section 5.2.

5.1.1 Implementation

The way the analytical approximation is obtained in practice is by using some variant of the Newton-Raphson algorithm to solve

$$\nabla \log \pi(\boldsymbol{x}) = \mathbf{0}, \tag{5.5}$$

with respect to $\boldsymbol{x}$ in order to find the mode and then evaluating $\nabla^2 \log \pi(\boldsymbol{x})$ at this point (Rasmussen and Williams, 2006; Rue et al., 2009). Note that a solution to (5.5) is not guaranteed to be the mode of $\pi(\boldsymbol{x})$, but can be any type of stationary point like a minima or a saddle point. For general likelihoods, $\pi(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} t_i(x_i)$, the solution of (5.5) should be verified to make sure that it is in fact the argument maximizing $\pi(\boldsymbol{x})$,

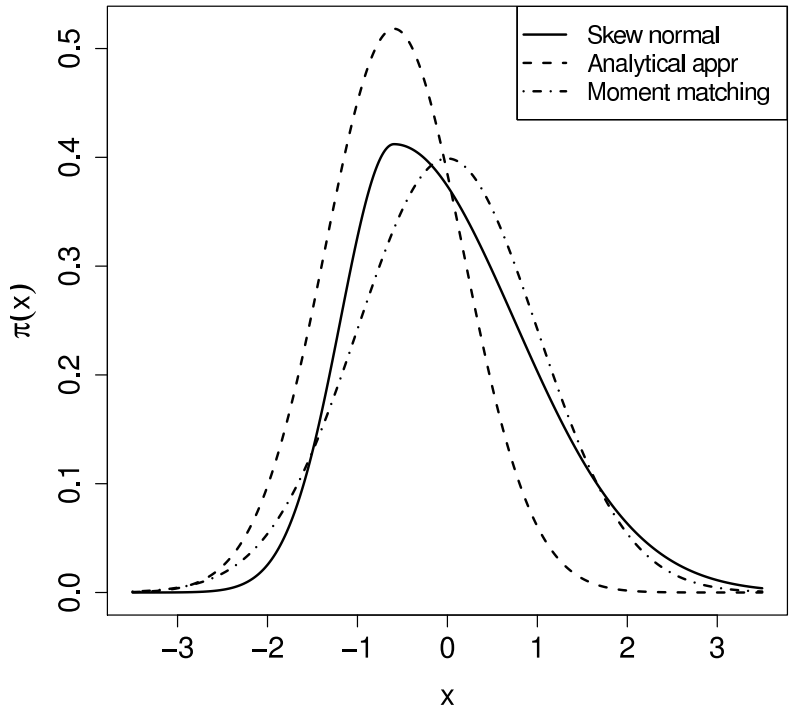


Figure 5.1: Gaussian approximations to a skew-normal distribution with mean 0, standard deviation 1 and scale parameter 1.5.

however, for *log-concave* likelihoods this is not necessary. The reason for this is that concave functions only have one stationary point which is the maxima.

Implementing the analytical method is fairly straightforward. The Newton-Raphson method used for solving $\arg_{\boldsymbol{x}}\{g(\boldsymbol{x}) = \boldsymbol{0}\}$, where $g : \mathbb{R}^n \to \mathbb{R}^n$ is a vector function, involves iterating

$$\boldsymbol{x}_{m+1} = \boldsymbol{x}_m - [\nabla g(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_m}]^{-1} g(\boldsymbol{x}_m), \quad m = 0, 1 \ldots,$$

until some convergence criterion is satisfied. In our case $g(\boldsymbol{x}) = \nabla \log \pi(\boldsymbol{x})$ and by using that

$$\log \pi(\boldsymbol{x}) = \text{const} - \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{x} + \sum_{i=1}^{n} \log t_i(x_i),$$

we get[6]

$$\begin{aligned} \nabla \log \pi(\boldsymbol{x}) &= -\boldsymbol{W}\boldsymbol{x} + \boldsymbol{v}, \\ \nabla^2 \log \pi(\boldsymbol{x}) &= -\boldsymbol{W} - \boldsymbol{C}, \end{aligned} \tag{5.6}$$

where $\boldsymbol{v} = (v_1, \ldots, v_n)^{\mathrm{T}}$ and $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$ are given by

$$v_i = \frac{\partial \log t_i(x_i)}{\partial x_i} \quad \text{and} \quad c_i = -\frac{\partial^2 \log t_i(x_i)}{\partial x_i^2} \quad \text{for } i = 1, \ldots, n.$$

The Newton-Raphson iterations can then be written as

$$\begin{aligned} \boldsymbol{x}_{m+1} &= \boldsymbol{x}_m + (\boldsymbol{W} + \boldsymbol{C}_m)^{-1}(-\boldsymbol{W}\boldsymbol{x}_m + \boldsymbol{v}_m), \\ &= (\boldsymbol{W} + \boldsymbol{C}_m)^{-1}(-\boldsymbol{W}\boldsymbol{x}_m + (\boldsymbol{W} + \boldsymbol{C}_m)\boldsymbol{x}_m + \boldsymbol{v}_m), \\ &= (\boldsymbol{W} + \boldsymbol{C}_m)^{-1}(\boldsymbol{C}_m\boldsymbol{x}_m + \boldsymbol{v}_m), \\ &= \boldsymbol{Q}_m^{-1}\boldsymbol{b}_m, \end{aligned} \tag{5.7}$$

where $\boldsymbol{Q}_m = \boldsymbol{W} + \boldsymbol{C}_m$ and $\boldsymbol{b}_m = \boldsymbol{C}_m\boldsymbol{x}_m + \boldsymbol{v}_m$. The subscript m is used for all the terms that depend on $\boldsymbol{x}_m$. Iterating according to Eq. (5.7) until convergence yields

$$\lim_{m\to\infty} \boldsymbol{x}_m = \boldsymbol{x}^* \quad \text{and} \quad \lim_{m\to\infty} \boldsymbol{Q}_m = \boldsymbol{Q},$$

where $\boldsymbol{x}^*$ is the mode of $\log \pi(\boldsymbol{x})$, which is used as the mean in the Gaussian approximation, and the matrix $\boldsymbol{Q}$ evaluated at this point is used as the precision matrix. As we can see from (5.7) using the Newton-Raphson routine is simply a matter of solving the system $\boldsymbol{Q}_m^{-1}\boldsymbol{b}_m$ several times. As mentioned in Section 2.3.1 one can do this by first computing the Cholesky triangle $\boldsymbol{L}$ of $\boldsymbol{Q}$ and then solving two triangular systems, though this requires $\boldsymbol{Q}$ to be symmetric positive definite for every $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots$, in the iterations.[7]

[6] The gradient and Hessian of quadratic forms are found from $\nabla(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{x}) = 2\boldsymbol{W}\boldsymbol{x}$ and $\nabla(\boldsymbol{W}\boldsymbol{x}) = \boldsymbol{W}$, respectively. See for instance Rasmussen and Williams (2006).

[7] If $\boldsymbol{Q}$ is not SPD for all values of $\boldsymbol{x}$ one can of course still solve $\boldsymbol{Q}_m^{-1}\boldsymbol{b}_m$ though other methods than the Cholesky factorization must be applied. Possible options are for instance the *LU-factorization* or one of the iterative *Krylov space methods*, see Golub and Van Loan (1996) and Saad (2003).

This is not guaranteed in the general, but if $t_i(x_i)$ is log-concave ($\log t_i(x_i)$ is concave) then

$$\frac{\partial^2 \log t_i(x_i)}{\partial x_i^2} < 0 \quad \text{for all } x_i \in \mathbb{R},$$

which means that $\text{diag}(\boldsymbol{c})$ is SPD[8] and thus $\boldsymbol{Q} = \boldsymbol{W} + \text{diag}(\boldsymbol{c})$ will be SPD as well by Property 2 in Result 2.5. This is used in Algorithm 5.1 which shows a possible implementation of the analytical approach for computing an unnormalized Gaussian approximation to $\pi(\boldsymbol{x})$. The most computationally expensive part of this algorithm is

Algorithm 5.1 Compute a Gaussian approximation to $\pi(\boldsymbol{x})$.

1: **input** $\boldsymbol{W} > 0$ and $\pi(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^n t_i(x_i)$ (log-concave)
2: **repeat**
3: $\quad \boldsymbol{C} := -\nabla^2 \log \pi(\boldsymbol{y}|\boldsymbol{x})$
4: $\quad \boldsymbol{b} := \boldsymbol{C}\boldsymbol{x} + \nabla \log \pi(\boldsymbol{y}|\boldsymbol{x})$
5: $\quad \boldsymbol{Q} = \boldsymbol{W} + \boldsymbol{C}$
6: $\quad \boldsymbol{L} = \text{chol}(\boldsymbol{Q})$
7: $\quad \boldsymbol{x} = \boldsymbol{L}^{\mathrm{T}} \backslash \boldsymbol{L} \backslash \boldsymbol{b}$
8: **until** convergence
9: $\log \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}} = \sum_{i=1}^n \log t(x_i) + \log(\det(\boldsymbol{W})) - \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{x} - \log(\det(\boldsymbol{Q}))$
10: **return** $\boldsymbol{x}$, $\boldsymbol{L}$ (or $\boldsymbol{Q}$) and $\log Z_{\boldsymbol{y}|\boldsymbol{\theta}}$

the repeated Cholesky factorization of $\boldsymbol{Q}$ which in general has complexity $\mathcal{O}(n^3)$. If $\boldsymbol{x}$ is a GMRF, however, this can be done much faster by the methods discussed in Section 2.3.3 as $\boldsymbol{Q}$ has the same sparse structure as $\boldsymbol{W}$ since adding $\boldsymbol{C}$ only changes the diagonal.

5.2 The expectation propagation algorithm

The expectation propagation (EP) algorithm provides an alternative to the analytical approach for computing an unnormalized Gaussian approximation $\tilde{\pi}(\boldsymbol{x})$ to $\pi(\boldsymbol{x})$. This method has gained much popularity in the machine learning community, due to its improved accuracy for computing a Gaussian approximation when compared with for instance the analytical approach (Minka, 2001; Rasmussen and Williams, 2006).

The idea in the EP algorithm is to use a suitable approximation to $\pi(\boldsymbol{x})$, $\tilde{\pi}(\boldsymbol{x}) = Z\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{Q}^{-1})$, that depends on some free parameters and optimize the approximation with respect to these parameters. By being able to tune the free parameters we have in way better control over the approximation than with the analytical approach which relies on an asymptotic expansion.

The starting point in deriving the expectation propagation algorithm is the factorization

$$\pi(\boldsymbol{x}) = \pi_0(\boldsymbol{x}) \prod_{i=1}^n t_i(x_i). \tag{5.8}$$

[8] For a diagonal matrix to be SPD all the elements must be larger than zero. This is easy to see by observing that: $\boldsymbol{x}^{\mathrm{T}}\text{diag}(a_1, \ldots, a_n)\boldsymbol{x} = \sum_{i=1}^n a_i x_i^2 > 0 \quad \forall \quad \boldsymbol{x} \neq \boldsymbol{0} \iff a_i > 0 \quad i = 1, \ldots, n.$

The idea is to approximate the likelihoods by un-normalized Gaussian functions in x_i, i.e.

$$t_i(x_i) \approx \tilde{t}_i(x_i) = \tilde{Z}_i \mathcal{N}(x_i \mid \tilde{\mu}_i, \sigma_i^2), \quad i = 1, \ldots, n. \tag{5.9}$$

The parameters $(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)$ are referred to as *site parameters* and these are the parameters we use to tune the approximation. As pointed out by Rasmussen and Williams (2006), the likelihood approximations should not be normalized as functions of the latent field since the exact likelihoods do not have this property (recall that $t_i(x_i)$ is a distribution for y_i not x_i). The product of the likelihood approximations can now be written in terms of a multivariate Gaussian distribution in $\boldsymbol{x}$ times a constant, i.e.

$$\prod_{i=1}^{n} \tilde{t}_i(x_i) = \mathcal{N}(\boldsymbol{x} \mid \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}) \prod_{i=1}^{n} \tilde{Z}_i, \tag{5.10}$$

where $\tilde{\boldsymbol{\mu}} = (\tilde{\mu}_1, \ldots, \tilde{\mu}_n)^{\mathrm{T}}$ and $\tilde{\boldsymbol{\Sigma}} = \mathrm{diag}(\tilde{\sigma}_1^2, \ldots, \tilde{\sigma}_n^2)$. The latent field prior is also a Gaussian pdf, $\pi_0(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, \boldsymbol{W}^{-1})$, and by using the identity for product of Gaussian pdfs in Eq. (2.3), the approximation to $\pi(\boldsymbol{x})$ is found as

$$\begin{aligned}
\tilde{\pi}(\boldsymbol{x}) &= \pi_0(\boldsymbol{x}) \prod_{i=1}^{n} \tilde{t}_i(x_i), \\
&= \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{0}, \boldsymbol{W}^{-1}) \mathcal{N}(\boldsymbol{x} \mid \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}) \prod_{i=1}^{n} \tilde{Z}_i, \\
&= \Big[\prod_{i=1}^{n} \tilde{Z}_i \Big] \mathcal{N}(\boldsymbol{0} \mid \tilde{\boldsymbol{\mu}}, (\boldsymbol{W}^{-1} + \tilde{\boldsymbol{Q}}^{-1})) \times \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}),
\end{aligned} \tag{5.11}$$

where $\tilde{\boldsymbol{Q}} = \tilde{\boldsymbol{\Sigma}}^{-1} = \mathrm{diag}(\tilde{\sigma}_1^{-2}, \ldots, \tilde{\sigma}_n^{-2})$. The precision and mean in the Gaussian distribution, $\mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}, \boldsymbol{Q}^{-1})$, are given respectively as

$$\boldsymbol{Q} = \boldsymbol{W} + \tilde{\boldsymbol{Q}} \quad \text{and} \quad \boldsymbol{\mu} = \boldsymbol{Q}^{-1} \tilde{\boldsymbol{Q}} \tilde{\boldsymbol{\mu}}. \tag{5.12}$$

By comparing the factors in (5.11) with the interpretation that $\tilde{\pi}(\boldsymbol{x})$ is the product of the approximated evidence and a Gaussian approximation to the conditional posterior of the latent field,

$$\tilde{\pi}(\boldsymbol{x}) = \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}} \tilde{\pi}_{\mathrm{G}}(\boldsymbol{x}),$$

we identify the following approximations

$$\begin{aligned}
\tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}} &= \Big[\prod_{i=1}^{n} \tilde{Z}_i \Big] \mathcal{N}(\boldsymbol{0} \mid \tilde{\boldsymbol{\mu}}, (\boldsymbol{W}^{-1} + \tilde{\boldsymbol{Q}}^{-1})), \\
\tilde{\pi}_{\mathrm{G}}(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}).
\end{aligned} \tag{5.13}$$

Note that since $\tilde{\boldsymbol{Q}}$ is a diagonal matrix the precision matrix $\boldsymbol{Q}$ will have the same sparse structure as $\boldsymbol{W}$, thus the Markov property of $\boldsymbol{x}$ is preserved in the approximated Gaussian distribution.

The difference between the expectation propagation algorithm and the analytical approach discussed in the previous section is the way the parameters in the local likelihood approximations are computed. Whereas the analytical approach is based on the Taylor expansion, which is an asymptotical technique, the expectation propagation algorithm optimize the site parameters $(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)$, $i = 1, \ldots, n$, by minimizing the difference between $\tilde{\pi}(\boldsymbol{x})$ and $\pi(\boldsymbol{x})$ using a distance measure. A popular distance measure used for this purpose is the *Kullback-Leibler* divergence, or relative entropy, which for two distributions π_1 and π_2 is defined as

$$\mathrm{KL}(\pi_1 \,||\, \pi_2) = \mathrm{E}_{\pi_1}\Big[\log \frac{\pi_2(\boldsymbol{x})}{\pi_1(\boldsymbol{x})}\Big]. \tag{5.14}$$

A few comments about the KL-divergence can be found in Appendix A. Common practice with the KL-divergence is to let π_1 play the role of the approximating distribution and let π_2 be the exact distribution, and then minimize $\mathrm{KL}(\pi_1||\pi_2)$ by tuning the parameters of π_1. This is the basis of the variational Bayes approximation schemes (Rue et al., 2009). In the expectation propagation algorithm this is done in reverse by letting π_2 be the Gaussian approximation and π_1 be the original distribution and then minimizing $\mathrm{KL}(\pi_1||\pi_2)$. However, according to Rasmussen and Williams (2006) minimizing $\mathrm{KL}(\pi(\boldsymbol{x})||\tilde{\pi}(\boldsymbol{x}))$, where the approximation $\tilde{\pi}(\boldsymbol{x})$ is Gaussian, turns out to be analytically intractable. To simplify the problem the expectation propagation algorithm optimize the site parameters by minimizing the KL-divergence between marginal distributions, i.e. by minimizing

$$\mathrm{KL}(\pi(x_i) \,||\, \tilde{\pi}(x_i)), \quad i = 1, \ldots, n.$$

There are still some problems with this procedure like the fact that the marginals $\pi(x_i)$, $i = 1, \ldots, n$, are not readily available. In the EP algorithm this problem is circumvented by working with pseudo-exact marginals, based on the Gaussian approximation, rather than the exact marginals. This is what makes up the core of the EP procedure as is explained in the next section.

5.2.1 Outline of the EP algorithm

The expectation propagation algorithm is an iterative procedure where in one step the site parameters of a single local likelihood approximation are updated. Assuming that $\tilde{\pi}(\boldsymbol{x})$ is an intermediate (or not optimal) approximation to $\pi(\boldsymbol{x})$ an iteration step in the EP algorithm is done according to the following procedure, see Minka (2001, p. 20),

1. Choose one $\tilde{t}_i(x_i)$ to update.

2. Compute the cavity distribution of x_i, $\pi_{-i}(x_i)$, which is the normalized ratio between the posterior marginal of x_i under $\tilde{\pi}(\boldsymbol{x})$ and $\tilde{t}_i(x_i)$

$$\tilde{\pi}_{-i}(x_i) \propto \frac{\tilde{\pi}(x_i)}{\tilde{t}_i(x_i)}.$$

3. Define $p_i(x_i)$, the pseudo-exact posterior marginal distribution of x_i, as

$$p_i(x_i) = t_i(x_i)\tilde{\pi}_{-i}(x_i).$$

4. Fit an un-normalized Gaussian distribution, $\hat{q}(x_i) = \hat{Z}_i\mathcal{N}(x_i|\hat{\mu}_i, \hat{\sigma}_i^2)$, to $p_i(x_i)$ by minimizing the KL-divergence, i.e. compute

$$(\hat{Z}_i, \hat{\mu}_i, \hat{\sigma}_i^2) = \underset{(\hat{Z}_i, \hat{\mu}_i, \hat{\sigma}_i^2)}{\operatorname{argmin}} \mathrm{KL}(p_i(x_i) \,||\, \hat{q}(x_i)).$$

5. Update $(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)$ so that

$$\tilde{t}_i(x_i) = \frac{\hat{q}_i(x_i)}{\tilde{\pi}_{-i}(x_i)}.$$

The interpretation of $p_i(x_i)$ as a pseudo-exact marginal likelihood can be justified by considering the posterior marginal of x_i under $\tilde{\pi}(\boldsymbol{x})$,

$$\tilde{\pi}(x_i) = \int \tilde{\pi}(\boldsymbol{x})\mathrm{d}\boldsymbol{x}_{-i} = \tilde{t}_i(x_i) \int \pi_0(\boldsymbol{x}) \prod_{i \neq j} \tilde{t}_j(x_j)\mathrm{d}x_j \propto \tilde{t}_i(x_i)\tilde{\pi}_{-i}(x_i).$$

Likewise, the true posterior marginal of x_i is

$$\pi(x_i) = t_i(x_i) \int \pi_0(\boldsymbol{x}) \prod_{i \neq j} t_j(x_j)\mathrm{d}x_j \propto t_i(x_i)\pi_{-i}(x_i).$$

Step 3 in the above procedure is thus the same as replacing $\tilde{t}_i(x_i)$ by its exact counterpart in the posterior marginal (under the Gaussian approximation) of x_i, which is the formulation used by Minka (2001). The resulting marginal, $p_i(x_i)$, is then a hybrid of the approximated and true marginal distribution of x_i. The normalization requirement of the cavity distribution might seem a bit odd considering that we are in fact computing an un-normalized Gaussian approximation. One might think that we loose some information about the constants $\tilde{Z}_j$, $j \neq i$, though these are accounted for automatically in Step 5 of the update procedure, which can be shown by skipping the normalization of $\tilde{\pi}_{-i}(x_i)$.

Another observation to make is that Step 4 and 5 of the update procedure is just a way of optimizing the marginal posterior distribution $\tilde{\pi}(x_i)$ by only tuning the site parameters of $\tilde{t}_i(x_i)$, i.e by computing

$$(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2) = \underset{(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)}{\operatorname{argmin}} \mathrm{KL}(p_i(x_i) \,||\, \tilde{\pi}(x_i)).$$

One should note that each set of site parameters $(\tilde{Z}_i, \tilde{\mu}_i, \tilde{\sigma}_i^2)$ generally depends on all the other site parameters through the cavity distribution. This means that once the parameters of $\tilde{t}_i$ are optimized according to the above procedure, the parameters of all the other $\tilde{t}_j$'s, $j \neq i$, must be re-optimized. In practice $\tilde{t}_i$, $i = 1, \ldots, n$, are usually updated sequentially and this process is repeated several times, see Rasmussen and

Williams (2006, p. 58). Ideally, this works as a fixed-point iteration where the site parameters eventually converge, though according to Rasmussen and Williams (2006, p. 59) there is no formal guarantee of convergence for the EP algorithm. However, the algorithm usually works fine for latent Gaussian models (Rasmussen and Williams, 2006).

5.2.2 Details on the update procedure

In order to implement the EP algorithm we need to have a closer look at the update procedure from the last section. Before an update the Gaussian approximation is

$$\tilde{\pi}(\boldsymbol{x}) = \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{Q}^{-1}).$$

Assuming that $\tilde{t}_i$ is chosen for update the marginal distribution of x_i must be determined, which can be done by using Property 2 in Result 2.1. If μ_i denotes element i in $\boldsymbol{\mu}$ and σ_i^2 is the ith diagonal element of $\boldsymbol{\Sigma} = \boldsymbol{Q}^{-1}$ then the marginal distribution of x_i is $\tilde{\pi}(x_i) = \text{const} \times \mathcal{N}(x_i|\mu_i, \sigma_i^2)$. Recalling that $\tilde{t}_i(x_i) = \tilde{Z}_i\mathcal{N}(x_i|\tilde{\mu}_i, \tilde{\sigma}_i^2)$ and using that the product of two Gaussian distributions is Gaussian, which of course also applies for ratios, the cavity distribution is found to be

$$\begin{aligned} \tilde{\pi}_{-i}(x_i) &= \mathcal{N}(x_i|\mu_{-i}, \sigma_{-i}^2) \\ \text{where} \quad \mu_{-i} = \sigma_{-i}^2(\sigma_i^{-2}\mu_i - \tilde{\sigma}_i^{-2}\tilde{\mu}_i) &\quad \text{and} \quad \sigma_{-i}^2 = (\sigma_i^{-2} - \tilde{\sigma}_i^{-2})^{-1}, \end{aligned} \tag{5.15}$$

by using Eq. (2.3) and normalizing. Step 3 of the update procedure is simply defining the pseudo exact marginal distribution $p_i(x_i) = t_i(x_i)\tilde{\pi}_{-i}(x_i)$, and one should note that $p_i(x_i)$ will not be normalized. Step 4 require us to minimize the Kullback-Leibler divergence between $p_i(x_i)$ and an un-normalized Gaussian distribution, $\hat{q}_i(x_i) = \hat{Z}_i\mathcal{N}(x_i|\hat{\mu}_i, \hat{\sigma}_i^2)$, by tuning the parameters of $\hat{q}_i(x_i)$. From Appendix A it is clear that this is the same as matching the moments of $p_i(x_i)$ and $\hat{q}_i(x_i)$, i.e.

$$\begin{aligned} \hat{Z}_i &= \int p_i(x_i)\mathrm{d}x_i, \\ \hat{\mu}_i &= \hat{Z}_i^{-1}\int x_i\, p_i(x_i)\mathrm{d}x_i, \\ \hat{\sigma}_i^2 &= \hat{Z}_i^{-1}\int x_i^2\, p_i(x_i)\mathrm{d}x_i - \hat{\mu}_i^2. \end{aligned} \tag{5.16}$$

Since neither $p_i(x_i)$ nor $\hat{q}_i(x_i)$ are normalized, also the zeroth moment is required to match as argued by Rasmussen and Williams (2006). The site parameters of $\tilde{t}_i(x_i)$ is then updated in step 5 by letting $\tilde{t}_i(x_i) = \hat{q}_i(x_i)/\tilde{\pi}_{-i}(x_i)$ which means that

$$\begin{aligned} \tilde{\sigma}_i^2 &= (\hat{\sigma}_i^{-2} - \sigma_{-i}^{-2})^{-1}, \\ \tilde{\mu}_i &= \tilde{\sigma}_i^2(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_{-i}^{-2}\mu_{-i}), \\ \tilde{Z}_i &= \hat{Z}_i\sqrt{2\pi(\sigma_{-i}^2 + \tilde{\sigma}_i^2)}\exp\left(\tfrac{1}{2}(\mu_{-i} - \tilde{\mu}_i)^2/(\sigma_{-i}^2 + \tilde{\sigma}_i^2)\right), \end{aligned} \tag{5.17}$$

again by using Eq. (2.3).

In order to include the updated set of site parameters in the full approximation the precision matrix and mean vector of $\tilde{\pi}(\boldsymbol{x})$ must be updated with the new $(\tilde{\mu}_i, \tilde{\sigma}_i^2)$. This is easy to do for $\boldsymbol{Q}$ as it only involves re-computing element i on the diagonal, i.e. $Q_{ii} = W_{ii} + \tilde{\sigma}_i^{-2}$, but the mean vector is more involved to update since this must be done by solving $\boldsymbol{\mu} = \boldsymbol{Q}^{-1}\tilde{\boldsymbol{Q}}\tilde{\boldsymbol{\mu}}$ with the updated $\boldsymbol{Q}$. The constant $\tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}$ does not play an active role in the update procedure (due to the normalization of the cavity distribution) and can be computed after the algorithm has converged.

5.2.3 Implementation

There are two details in the expectation propagation algorithm which requires extra attention with regards to the implementation. Those are,

1. At what point do we include the updated site parameters in the full approximation?
2. How do we deal with the integrals in Eq. (5.16)?

This first question is the most crucial for making an efficient implementation of the EP algorithm as updating the full approximation is the most computationally expensive operation, at least when n is large. The issue is that newly updated site parameters, e.g. $(\tilde{\mu}_i, \tilde{\sigma}_i^2)$, do not influence the update of a second local likelihood approximation $\tilde{t}_j$, $j \neq i$, until they are included in $\boldsymbol{Q}$ and $\boldsymbol{\mu}$. Still, updating several $\tilde{t}_i$'s before including the new information in the full approximation does have its merits as the cost of updating $\boldsymbol{Q}$ and $\boldsymbol{\mu}$ only grows linearly with the number new sets of site parameter to include. We must also keep in mind that $\boldsymbol{Q}$ must be partly inverted as the marginal variances, i.e. $\text{diag}(\boldsymbol{\Sigma})$, is needed in the update procedure of $\tilde{t}_i$. Since $\boldsymbol{x}$ is a Gaussian Markov random field this can be done (relatively) efficiently by using the method outlined in Sec. 2.4 where the sparse structure of $\boldsymbol{Q}$ was exploited. The problem is that, just as with $\boldsymbol{\mu}$, all the elements in $\boldsymbol{\Sigma}$ are in general sensitive to changes to the diagonal of $\boldsymbol{Q}$, even if only one element is changed. One option for an EP implementation is to update all the $\tilde{t}_i$'s independently and then update the full approximation in one go. The update of $\tilde{\pi}(\boldsymbol{x})$ is then done by first computing

$$\boldsymbol{Q} = \boldsymbol{W} + \tilde{\boldsymbol{Q}}_{\text{new}},$$

then compute the Cholesky factorization $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^{\text{T}}$ which is used for solving $\boldsymbol{\mu} = \boldsymbol{Q}^{-1}\tilde{\boldsymbol{Q}}\tilde{\boldsymbol{\mu}}$ and computing $\text{diag}(\boldsymbol{\Sigma})$. This approach is used in Cseke and Heskes (2010), where it is called a *parallel* expectation propagation scheme, referring to the fact that the $\tilde{t}_i$'s are (or can be) updated in parallel. An implementation using this approach is given in Algorithm 5.2. Note that if the Gaussian approximation is parameterized using the covariance matrix rather than the precision matrix, as done by Rasmussen and Williams (2006, pp. 57-58), other issues arise. On the positive side the marginal variances are not necessary to compute as these are already available, though the new problem is how to update the covariance matrix with a newly updated $\tilde{\sigma}_i^2$. In Rasmussen and Williams (2006) this is done by using a rank-one update once a newly updated pair $(\tilde{\mu}_i, \tilde{\sigma}_i^2)$ is available. This way new information is used immediately in the following updates which can speed up convergence. Once all the local approximations have been visited the full

covariance matrix is recomputed from $\boldsymbol{\Sigma} = (\boldsymbol{W} + \tilde{\boldsymbol{Q}}_{\text{new}})^{-1}$ as the rank-one updates are known to be slightly inaccurate, see Rasmussen and Williams (2006). The problem with this procedure is that the rank-one updates are computational expensive with a cost of $\mathcal{O}(n^2)$ flops, and also any Markov property assumption can not be utilized as this does not imply sparseness of $\boldsymbol{\Sigma}$. Thus, the total cost of the EP algorithm using the covariance matrix is always $\mathcal{O}(n^3)$.

Moving on to the question of how to implement the moment matching procedure, Eq. (5.16), we see that the choice is either to use numerical integration or to solve the integrals analytically. Rasmussen and Williams (2006) chose the latter approach in an example using Gaussian latent fields for binary classification, and it is clear that solving the three integrals analytically can involve much preliminary work. Using numerical integration for solving (5.16) is inevitable in most cases, and a good choice of method is the Gauss-Hermite quadrature. The Gauss-Hermite quadrature rule is usually written as

$$\int_{-\infty}^{\infty} f(z)\mathrm{e}^{-z^2/2}\mathrm{d}z \approx \sum_{z_i} w_i f(z_i),$$

where the abissca points $\{z_i\}$ are the zeros in the kth order Hermite polynomial and the weights are given in terms of the $(k-1)$th order Hermite polynomial (Abramowitz and Stegun, 1964). Since all the integrals in (5.16) can be rewritten to be in the same from as the above integral (recall that the cavity distribution is Gaussian) the Gauss-Hermite scheme is particularly well suited for computing the first three moments of $p_i(x_i)$. The abissca points and weights can be computed in advance (or found in a table) and the Gauss-Hermite quadrature then has complexity $\mathcal{O}(c_f k)$, where c_f the complexity associated with the function evaluations of $f(\cdot)$ (usually taken to be constant, i.e. $\mathcal{O}(1)$).

By considering each step of Alg. 5.2 we see that this EP implementation is dominated by the Cholesky factorization of $\boldsymbol{Q}$ and solving for $\text{diag}(\boldsymbol{\Sigma})$ using $\boldsymbol{L}$ by the method from Sec. 2.4. For a dense precision matrix $\boldsymbol{Q}$ both tasks have complexity $\mathcal{O}(n^3)$, though assuming that $\boldsymbol{x}$ is a GMRF can reduce this cost quite a lot. For instance if $\boldsymbol{Q}$ has bandwidth b_w the complexity of the EP algorithm is $\mathcal{O}(b_w^2 n)$ as this is the cost of both factorizing $\boldsymbol{Q}$ and solving for $\text{diag}(\boldsymbol{\Sigma})$.

6 Analysis and Results

The analysis part in this text has been conducted with the aim of investigating two issues: comparison of the analytical approach and the expectation propagation algorithm discussed in Sec. 5 with regards to time required for computing $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$, and the accuracy in the approximations. For testing computational time use the two methods are used on simulated data for models where the latent field have a band precision matrix. The linear algebra routines can then be implemented using the algorithms presented in Sec. 2.3.3–2.4. For comparison of accuracy in approximating $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ the models were tested on a real data set where it is known that the analytical approach does not give sufficiently good approximations. In this case the expectation propagation approximation should be an improvement.

Algorithm 5.2 Expectation propagation algorithm (precision matrix version)

input $\boldsymbol{W}$ (precision matrix), $\boldsymbol{y}$ (data), $\pi(\boldsymbol{y}|\boldsymbol{x})$ (data distribution)
$\tilde{\boldsymbol{Q}} = \boldsymbol{0}$, $\tilde{\boldsymbol{\mu}} = \boldsymbol{0}$.
repeat
  $\boldsymbol{L} = \text{chol}(\boldsymbol{Q})$
  $\boldsymbol{\mu} = \boldsymbol{L}^{\mathrm{T}} \backslash \boldsymbol{L} \backslash \tilde{\boldsymbol{Q}} \tilde{\boldsymbol{\mu}}$
  Compute $\text{diag}(\boldsymbol{\Sigma}) = (\sigma_1^2, \ldots, \sigma_n^2)^{\mathrm{T}}$ from $\boldsymbol{L}$
  for $i := 1$ to n **do**
    Compute μ_{-i} and σ_{-i}^2 from (5.15)
    Compute $\hat{Z}_i$, $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ from (5.16)
    Compute $\tilde{Z}_i$, $\tilde{\mu}_i$ and $\tilde{\sigma}_i^2$ from (5.17)
  end for
  $\boldsymbol{Q} = \boldsymbol{W} + \tilde{\boldsymbol{Q}}$
until convergence
$\boldsymbol{L} = \text{chol}(\boldsymbol{Q})$
$\boldsymbol{\mu} = \boldsymbol{L}^{\mathrm{T}} \backslash \boldsymbol{L} \backslash \tilde{\boldsymbol{Q}} \tilde{\boldsymbol{\mu}}$
Compute $\log \tilde{Z}_{\boldsymbol{y}|\boldsymbol{\theta}}$ from (5.13)
return $\boldsymbol{Q}$ (or $\boldsymbol{L}$), $\boldsymbol{\mu}$

6.1 Simulated data – Test of efficiency

The first part of the analysis has been done by studying simulating data from two different models. The first is a stochastic volatility model with an AR(1) latent field, and the second model is a binary logistic likelihood model with an AR(2) latent field. These examples are slightly modified versions of two examples studied by Rue et al. (2009, Sec. 5), and are deigned so that the precision matrix will have a band structure. In particular this means that there cannot be any common mean element in the latent field as this ruins the band structure.

6.1.1 Stochastic volatility model

Stochastic volatility models are commonly used with financial time series. In Rue et al. (2009, Ex. 5.3) such a model was applied with data consisting of the daily difference of a currency exchange rate over a certain period of time.

The model used here is formally given as

$$\begin{aligned} y_t \mid \boldsymbol{x} &\overset{\text{iid}}{\sim} \mathcal{N}(0, \exp(x_t)), \quad t = 1, \ldots, n, \\ x_1 \mid \phi', \tau &\sim \mathcal{N}\big(0, 1/((1-\phi^2)\tau)\big), \\ x_t \mid \phi', \tau, x_1, \ldots, x_{t-1} &\sim \mathcal{N}(\phi x_{t-1}, \tau^{-1}), \quad t = 2, \ldots, n, \\ \tau &\sim \mathit{Gamma}(0.001, 0.001), \\ \phi' &\sim \mathcal{N}(0, 3), \end{aligned} \tag{6.1}$$

where $\phi' = \log((\phi+1)/(\phi-1))$ (i.e. $|\phi| < 1$ means $\phi' \in \mathbb{R}$) and the Gamma distribution is parameterized so that $\mathrm{E}(\tau) = 1$ and $\text{var}(\tau) = 1000$. The hyperparameters are

collected in a vector $\boldsymbol{\theta} = (\phi', \tau)^{\mathrm{T}}$ and the elements are taken to be independent, i.e. $\pi(\boldsymbol{\theta}) = \pi(\phi')\pi(\tau)$. It was briefly mentioned in Sec. 2.2 that an AR(1) process such as then one given in (6.1) is a GMRF with respect to the graph in Fig. 2.2, which in turn means that the precision matrix, $\text{prec}(\boldsymbol{x}|\boldsymbol{\theta}) = \boldsymbol{W}_1(\boldsymbol{\theta})$, has bandwidth 1. The prior for the first latent field variable has been chosen in way that makes the precision matrix invariant when reversing the order of the nodes. Explicitly computing the precision matrix of $\boldsymbol{x}$ can be done by the method outlined in Example 2.2, and we find that

$$\boldsymbol{W}_1(\boldsymbol{\theta}) = \tau \begin{pmatrix} 1 & -\phi & & & \\ -\phi & 1+\phi^2 & -\phi & & \\ & \ddots & \ddots & \ddots & \\ & & -\phi & 1+\phi^2 & -\phi \\ & & & -\phi & 1 \end{pmatrix}. \tag{6.2}$$

The the simulated data used in this analysis is shown in Figure 6.1. As we can see the observations fluctuate around a zero mean with a deviation depending on the level of the latent field.

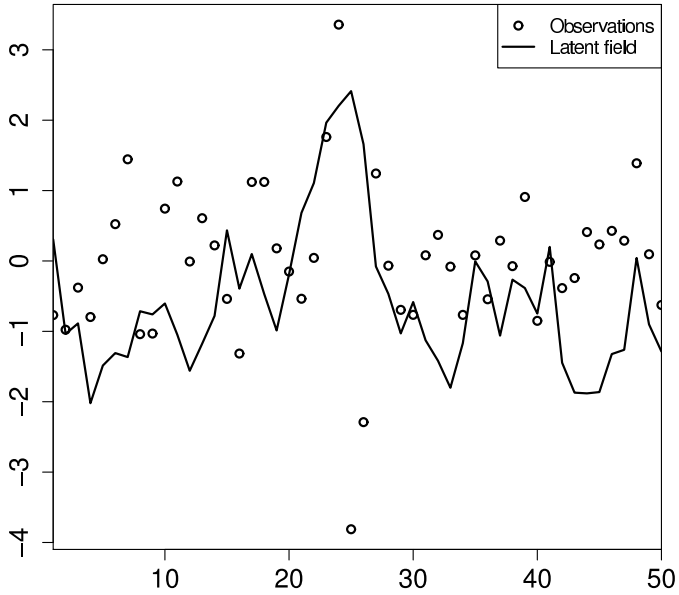
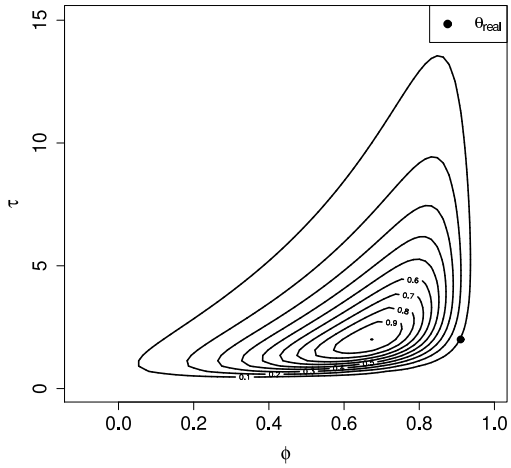


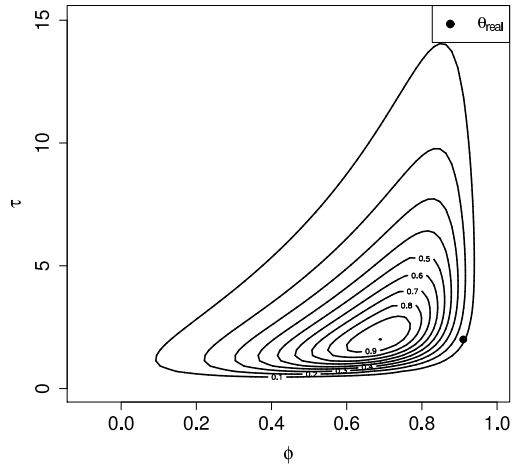
Figure 6.1: Simulated data and true latent field for the stochastich volatility model. The number of observations is $n = 50$ and the true parameter values are $\phi = 0.91$ and $\tau = 2$.

The approximated joint posterior of the hyperparameters was computed with both the analytical approach and the expectation propagation algorithm from Sec. 5, and the results are shown in Figures 6.2a and 6.2b. By using deterministic numerical integration, as discussed in Sec. 4, approximations to the posterior marginal distributions were also found and are shown in Fig. 6.2c and 6.2d. As we can see from these figures the analytical approach and the expectation propagation algorithm yields similar approximations for

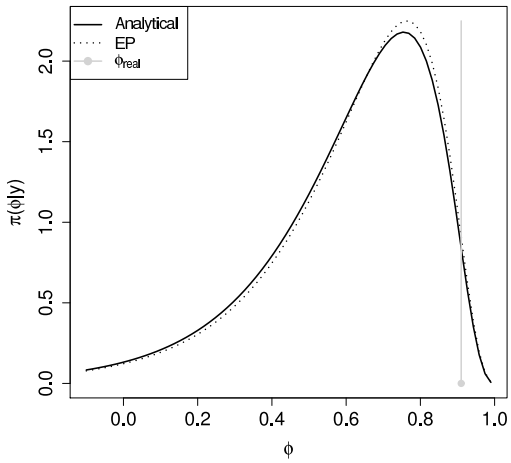
this data set. It is hard to tell whether or not these approximations provides a reasonable fit to the true posterior distributions, and ideally the approximations should have been compared with results from MCMC simulations.



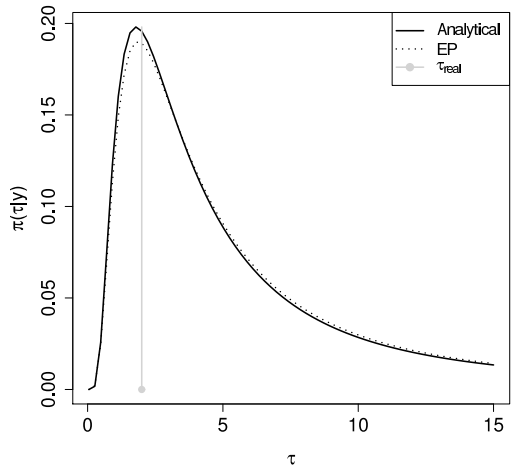
(a) Approximated joint posterior distribution $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ computed using the analytical approach.



(b) Approximated joint posterior distribution $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ computed using EP.



(c) Approximated marginal posterior distribution for ϕ, $\tilde{\pi}(\phi|\boldsymbol{y})$.



(d) Approximated marginal posterior distribution for τ, $\tilde{\pi}(\tau|\boldsymbol{y})$.

Figure 6.2: Approximate Bayesian analysis of the hyperparameters in the stochastic volatility model, Eq. (6.1).

6.1.2 Logistic model

The second model in this study is the logistic model for binary data, often used for classification problems, see Rasmussen and Williams (2006). This model is given as

$$\begin{aligned} y_i \mid x_i &\overset{\text{iid}}{\sim} (\text{logit}^{-1}(x_i))^{y_i}(1-\text{logit}^{-1}(x_i))^{1-y_i}, \\ \boldsymbol{x} \mid \kappa &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{W}_2^{-1}(\kappa)), \\ \kappa &\sim \mathit{Gamma}(0.001, 0.001), \end{aligned} \tag{6.3}$$

where $\text{logit}^{-1}(x_i) = 1/(1+\exp(-x_i))$. The latent field is assumed to be an AR(2) model

$$x_t|\boldsymbol{x}_{1:(t-1)} \sim \mathcal{N}(\phi_1 x_{t-2} + \phi_2 x_{t-1}, \kappa^{-1}), \quad t = 2, \ldots, n. \tag{6.4}$$

As with the AR(1) model the priors for x_1 and $x_2|x_1$ are Gaussian and chosen so that precision matrix is invariant when reversing the ordering of the nodes. The results is

$$\boldsymbol{W}_2(\kappa) = \kappa \begin{pmatrix} 1 & -\phi_1 & c & & & & \\ -\phi_1 & 1+\phi_1^2 & b & c & & & \\ c & b & a & b & c & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & c & b & a & b & c \\ & & & c & b & 1+\phi_1^2 & -\phi_1 \\ & & & & c & -\phi_1 & 1 \end{pmatrix}, \tag{6.5}$$

where

$$\begin{aligned} a &= 1 + \phi_1^2 + \phi_2^2, \\ b &= -\phi_1 + \phi_1\phi_2, \\ c &= -\phi_2. \end{aligned} \tag{6.6}$$

The weights in the AR(2) model are assumed known and chosen so that the process is stationary, $\phi_1 = 0.7$ and $\phi_2 = 0.2$.

The simulated data for this model are shown in Figure 6.3. Just as with the stochastic volatility data both the analytical approach and the expectation propagation algorithm was used to compute the approximated posterior distribution for the single hyperparameter in this model. The results are displayed in Figure 6.4. As we can see the approximations are quite similar except that the mode of the EP approximation is a bit shifted towards zero and flat out more rapid than the analytical approximation. Still, since MCMC simulations was not run for this case it is hard to determine if the EP algorithm offer much of an improvement compared to the analytical approximation.

6.1.3 Computational time requirement

The main purpose with the analysis of the simulated data is to compare the computational time requirement for the expectation propagation algorithm with that of the

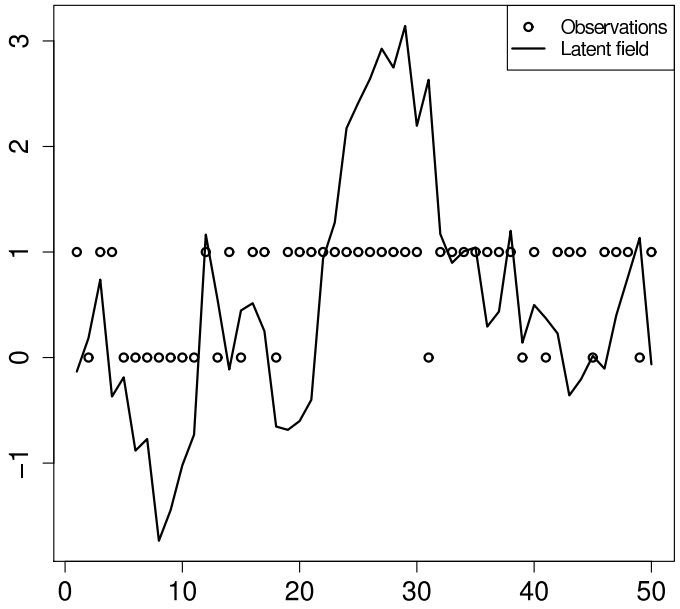


Figure 6.3: Simulated data and true latent field for the binary data logistic model. The number of observations is $n = 50$ and the true parameter values are $\phi_1 = 0.70$, $\phi_2 = .20$ and $\kappa = 2$.

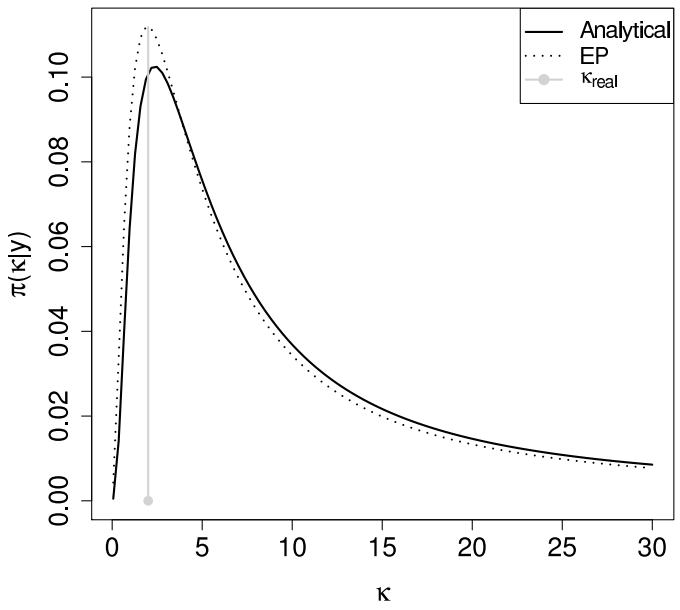


Figure 6.4: Approximate marginal posterior distribution for κ.

analytical approach. Both of these methods have a computational complexity of $\mathcal{O}(b_w^2 n)$ when the precision matrix of the latent field is has bandwidth b_w. Of course, efficiency of each of the algorithms depends on many factors, especially how many time consuming operations are needed per iteration. For instance, every iteration step the expectation propagation algorithm requires the computation of the diagonal of the inverse precision matrix and $3n$ numerical integrations, in addition to computing a Cholesky factorization and solve a system of equations. The Newton-Raphson routine used by the analytical approximation, on the other hand, only computes the Cholesky factorization of the precision matrix and solves for the expectation. To test how fast the two algorithms are compared to each other they were tested using implementations written from scratch in the C programming language. The numerical linear algebra routines needed were also implemented using C, and were used by the both the approximation algorithms. Since the precision matrices for both models, Eq. (6.1) and (6.3), have a band structure the numerical linear algebra routines are implementations of the algorithms presented in Sec. 2.3.3 and 2.4. By using this approach the two algorithms are tested on a more or less equal footing and the results should give an indication of how fast the expectation propagation algorithm can be, compared with the analytical approximation.

All the tests were performed in `R` (R Development Core Team, 2010) and the times were collected using the built-in function `system.time`.[9] The results of the time tests are given in Table 6.1. These results show that the implementations behaves roughly as expected, the analytical approximation is about 15–20 times faster to compute than the EP approximation, and both the implementations have a linear time increase as a function of the latent field dimension n. It is quite clear that computing the expectation propagation approximation requires a substantial amount of extra computational time compared to the analytical approximation, even for quite simple models where the precision matrix has a band structure. The time difference between the two methods would likely be greater if more complex latent field models, like spatial or spatiotemporal fields, were studied. This is because the more general linear algebra methods are much more computationally demanding than the band matrix algorithms. Say if the latent field is a spatial GMRF then the Cholesky factorization of the precision matrix will in general have a complexity of $\mathcal{O}(n^{3/2})$ (Rue and Held, 2005). Partially inverting the precision matrix to obtain the marginal variances also increase in complexity and for the spatial case this is an $\mathcal{O}(n \log(n)^2)$ operation (Rue and Martino, 2007). It is safe to say that the time requirement of EP is a bit inhibiting for this algorithm to be used for fast inference. The approximated posterior distributions were also quite similar, so in these two models the analytical approximation method is preferable to EP for use in the approximate Bayesian inference scheme discussed in Sec. 4.

6.2 Binary data from a longitudinal study – Test of accuracy

In the analysis of the simulated data the expectation propagation algorithm and the analytical approximation gave quite similar approximations to the posterior distribution

[9] The C implementations of the expectation propagation and the analytical algorithm were compiled to make a shared object which was loaded and called from `R` using the `.Call` function.

Table 6.1: Results from the time tests of the C implementations of the expectation propagation and the analytical approximation algorithms. The tests were done by computing $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ for 16 values of $\boldsymbol{\theta}$, and were conducted on a computer with a dual core (2.4GHz) processor running a Linux operating system.

Model	n	Time EP [s]	Time Analytic [s]	EP Analytic time ratio
Stochastic vol.	50	1.740	0.100	17.4
Stochastic vol.	200	7.600	0.560	13.6
Logit	50	1.130	0.070	16.1
Logit	200	4.690	0.260	18.0

of the hyperparameters, $\pi(\boldsymbol{\theta}|\boldsymbol{y})$. In this section we investigate a data set where the analytical approximation does not fit well with the corresponding MCMC simulation, used as a gold standard. As we will see the expectation propagation approximation offers a slight improvement in this particular case.

The data used here is the `toenail` data set found in the `R`-package `glmmAK` (Komarek, 2010). This is data from a longitudinal clinical trial in dermatology conducted for testing the efficacy of two oral treatments of toenail infections. Each patient was given one of the two treatments and the degree of onycholysis, which expresses the degree of separation of the nail plate from the nail-bed, was recorded on between 1 to 7 visits made by the individual patient. There were a total number of $m = 294$ patients participating in this study and a total of $N = 1908$ observations were recorded. The number of visits for patient number i is denoted n_i and the result on visit number j is denoted $y_{ij} \in \{0, 1\}$. The data are coded so that $y_{ij} = 0$ means that the degree of oncholysis was either mild or absent and $y_{ij} = 1$ means that the degree of onycholysis was either moderate or severe.

The goal now is to fit a generalized linear model with the covariates $\texttt{trt}_{ij}$ (treatment received patient i visit j) and $\texttt{time}_{ij}$ (time of visit j for patient i) to these observations, and analyze this model by using the inference methods discussed in Sec. 4. The model suggested in Komarek (2010) is a logistic regression model with the linear predictor given as

$$\text{logit}(\pi(y_{ij} = 1 \mid \boldsymbol{\beta}, \mu_i)) = \beta_0 + \beta_1 \texttt{trt}_{ij} + \beta_2 \texttt{time}_{ij} + \beta_3(\texttt{trt}_{ij} \times \texttt{time}_{ij}) + u_i, \quad (6.7)$$

where $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_3)^{\mathrm{T}}$ are fixed random effects and u_i is random noise associated with patient i. In order to formulate a latent Gaussian model we need to define the latent field and choose Gaussian prior distributions for its elements. The priors used in this model are

$$\begin{aligned} \boldsymbol{\beta} &\sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}), \\ \boldsymbol{u} &\sim \mathcal{N}(\mathbf{0}, \tau^{-1} \boldsymbol{I}), \\ \tau &\sim \Gamma(s, r), \end{aligned} \quad (6.8)$$

thus $\boldsymbol{\beta}$ and $\boldsymbol{u}$ are considered to be latent variables, τ is a hyperparameter, and $\sigma^2 = 10^4$, $s = 0.01$ and $r = 0.01$ are fixed parameters.

In order to use the methods from Section 5 in the approximate analysis we need the dimension of the latent field to be larger than or equal to the number of observations. This is because both the analytical approximation and the expectation propagation algorithm requires the likelihoods to depend on a single latent field variable. We achieve this by adding zero mean noise terms $\varepsilon_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, \kappa^{-1})$ with assumed high (and fixed) precision, e.g. $\kappa = 10^6$, in Eq. (6.7). Due to the high precision this is more or less the same as adding a deterministic term equal to zero, which should not make much difference in the analysis. By defining

$$\text{logit}(\pi(y_{ij} = 1 \mid \boldsymbol{\beta}, \mu_i, \varepsilon_{ij})) = \eta_{ij}, \tag{6.9}$$

the linear predictor term is on the same form as in Eq. (3.1),

$$\begin{aligned} \eta_{ij} = \beta_0 + \beta_1 \texttt{trt}_{ij} + \beta_2 \texttt{time}_{ij} + \beta_3 (\texttt{trt}_{ij} \times \texttt{time}_{ij}) + u_i + \varepsilon_{ij}, \quad & 1 \leq i \leq m, \\ & 1 \leq j \leq n_i. \end{aligned} \tag{6.10}$$

The full latent field in this model is $\boldsymbol{x} = (\boldsymbol{\eta}^{\mathrm{T}}, \boldsymbol{u}^{\mathrm{T}}, \boldsymbol{\beta}^{\mathrm{T}})^{\mathrm{T}}$, with dimension $\dim(\boldsymbol{x}) = N + m + 4 = 2206$. In order to find the distribution of the latent field given τ (the only hyperparameter) we can consider the following factorization

$$\begin{aligned} \pi(\boldsymbol{x} \mid \tau) &= \pi(\boldsymbol{\eta}, \boldsymbol{\beta}, \boldsymbol{u} \mid \tau), \\ &= \pi(\boldsymbol{\eta} \mid \boldsymbol{\beta}, \boldsymbol{u}) \pi(\boldsymbol{\beta}) \pi(\boldsymbol{u} \mid \tau), \\ &\propto \exp\Big\{ -\frac{\kappa}{2} (\boldsymbol{\eta} - \boldsymbol{C}\binom{\boldsymbol{\beta}}{\boldsymbol{u}})^{\mathrm{T}} (\boldsymbol{\eta} - \boldsymbol{C}\binom{\boldsymbol{\beta}}{\boldsymbol{u}}) - \frac{1}{2\sigma^2} \boldsymbol{\beta}^{\mathrm{T}} \boldsymbol{\beta} - \frac{\tau}{2} \boldsymbol{u}^{\mathrm{T}} \boldsymbol{u} \Big\}, \\ &= \exp\Big\{ -\frac{1}{2} \boldsymbol{x}^{\mathrm{T}} \boldsymbol{W}(\tau) \boldsymbol{x} \Big\}. \end{aligned} \tag{6.11}$$

Thus, the prior for the latent field is $\boldsymbol{x} | \tau \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}^{-1}(\tau))$. The matrix $\boldsymbol{C}$ is defined as $\boldsymbol{C} = (\boldsymbol{B}, \boldsymbol{U})$ where

$$\boldsymbol{B} = \begin{pmatrix} 1 & \texttt{trt}_{11} & \texttt{time}_{11} & \texttt{trt}_{11} \times \texttt{time}_{11} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \texttt{trt}_{1n_1} & \texttt{time}_{1n_1} & \texttt{trt}_{1n_1} \times \texttt{time}_{1n_1} \\ 1 & \texttt{trt}_{21} & \texttt{time}_{21} & \texttt{trt}_{21} \times \texttt{time}_{21} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \texttt{trt}_{mn_m} & \texttt{time}_{mn_m} & \texttt{trt}_{mn_m} \times \texttt{time}_{mn_m} \end{pmatrix} \quad \text{and} \quad \boldsymbol{U} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Explicitly computing the precision matrix $\boldsymbol{W}(\tau)$ is a bit cumbersome and the details have been omitted. The result is

$$\boldsymbol{W}(\tau) = \kappa \begin{pmatrix} \boldsymbol{I}_N & -\boldsymbol{U} & -\boldsymbol{B} \\ -\boldsymbol{U}^{\mathrm{T}} & \boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} + \tau\kappa^{-1}\boldsymbol{I}_m & \boldsymbol{U}^{\mathrm{T}}\boldsymbol{B} \\ -\boldsymbol{B}^{\mathrm{T}} & \boldsymbol{B}^{\mathrm{T}}\boldsymbol{U} & \boldsymbol{B}^{\mathrm{T}}\boldsymbol{B} + (\sigma^2\kappa)^{-1}\boldsymbol{I}_4 \end{pmatrix}. \tag{6.12}$$

The model can now be summarized as

$$\begin{aligned} y_{ij} \mid \eta_{ij} &\overset{\text{indep}}{\sim} (\text{logit}^{-1}(\eta_{ij}))^{y_{ij}} (1 - \text{logit}^{-1}(\eta_{ij}))^{1 - y_{ij}}, \\ \boldsymbol{x} \mid \tau &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}^{-1}(\tau)), \\ \tau &\sim \mathit{Gamma}(0.01, 0.01). \end{aligned} \tag{6.13}$$

Even though it has not been explicitly mentioned $\boldsymbol{x}$ is a Gaussian Markov random field with respect to a very sparse graph. Figure 6.5 displays the precision matrix of $\boldsymbol{x}|\tau$ with the non-zero elements indicated by a black dot. The total number of non-zero elements in this matrix is 18422, which means that only 0.4% of the total number of elements are different from zero. Observe, however, that $\boldsymbol{W}(\tau)$ does not have a band structure (the bandwidth is 2205) so to efficiently factorize this matrix we must used the general sparse factorization methods discussed in the end of Sec. 2.3.3.

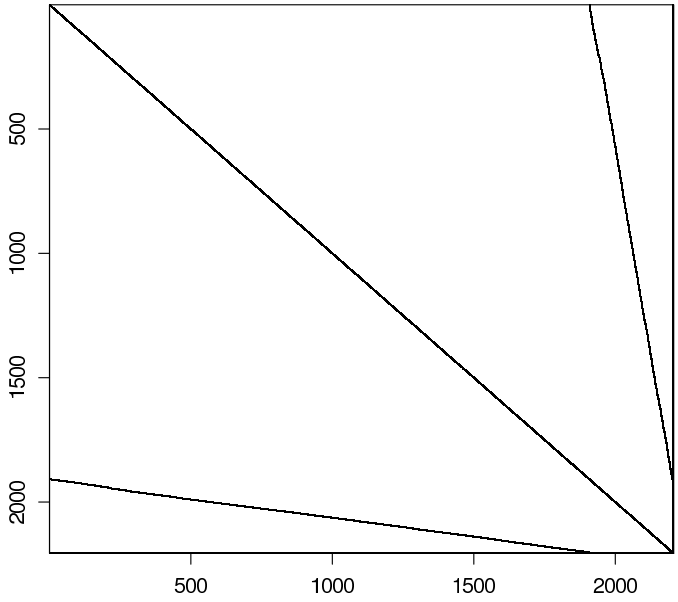


Figure 6.5: The precision matrix $\boldsymbol{W}(\tau)$ in the model given by Eq. (6.13).

The primary task in the analysis of model (6.13) is to compute the approximated posterior distribution for the hyperparameter τ, that is, the precision for the random effects associated with each individual patient. Recall from Sec. 4 that this distribution is approximated using the constant $\tilde{Z}$ in the unnormalized Gaussian approximation $\tilde{\pi}(\boldsymbol{x}, \boldsymbol{y}|\tau) = \tilde{Z}\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{Q}^{-1})$. Both the analytical approximation method (Taylor expansion) and the expectation propagation algorithm was used here for computing $\tilde{\pi}(\boldsymbol{x}, \boldsymbol{y}|\tau)$. For assessing the errors in these approximations a long MCMC simulation, with a total of 3×10^5 samples, was run by using the `glmmAK` package (Komarek, 2010). In addition to the analytical approximation method and EP a third approximation scheme using EP in the Laplace approximation (discussed in Sec. 4.1), was tested. Figure 6.2 displays the result of this analysis. It is clear from this figure that the EP approximation is an improvement to the analytical even though non of the approximations are really optimal. To get a quantitative measure of how good these approximations are, the symmetric Kullback-Leibler divergence, see Appendix A, between the approximations and the MCMC gold standard was also computed. The results are $\text{SKL}(\pi_{\text{MCMC}}||\pi_{\text{LA}}) = 1.865$ for the analytical approximation and $\text{SKL}(\pi_{\text{MCMC}}||\pi_{\text{EP}}) = 0.917$ for the EP approximation. The difference between the two approximation methods is clearly non negligible

for this particular data set, and by halving the symmetric Kullback-Leibler divergence the EP approximation is clearly the better choice.

As mentioned in Sec. 4.1 using the Gaussian approximation to $\pi(\boldsymbol{x}|\boldsymbol{y},\tau)$, computed by EP, in the exact identity (4.10) and evaluating in the mode of $\tilde{\pi}(\boldsymbol{x}|\boldsymbol{y},\tau)$ is not a good idea. This Laplace-EP mixture is shown in Figure 6.2 as the dotted line and clearly this approximation miss the true distribution by quite a bit. The symmetric KL-divergence in this case is a staggering 7.782. Thus, this identity given in Eq. (4.10) should not be used for constructing approximations to the posterior of the hyperparameters. One should rather stick to the approximation obtained from the unnormalized Gaussian $\tilde{\pi}(\boldsymbol{x},\boldsymbol{y}|\tau) = \tilde{Z}\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{Q}^{-1})$, i.e. $\pi(\tau|\boldsymbol{y}) \propto \tilde{Z}\pi(\tau)$.

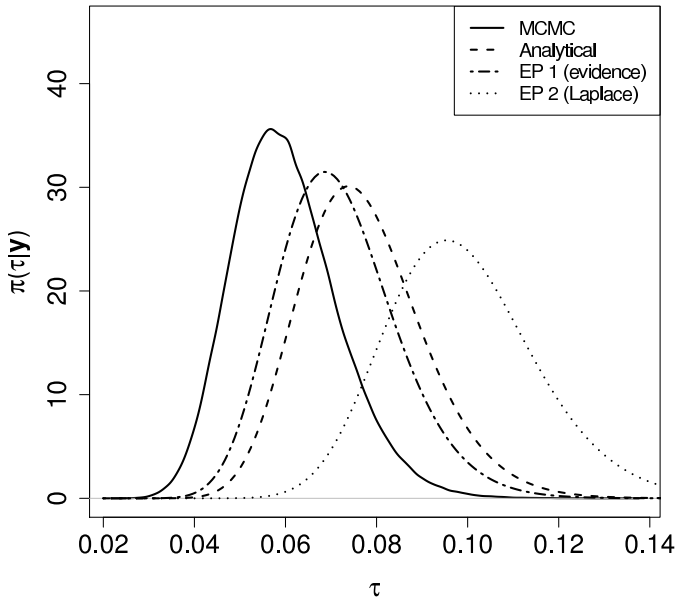


Figure 6.6: The posterior marginal distribution for the hyperparameter τ in model (6.13).

6.2.1 Marginal distributions for the latent field

It was briefly mentioned in the end of Sec. 4 that finding point-wise approximations to $\pi(x_i|\boldsymbol{y},\tau)$ is helpful for computing the posterior marginals of the latent field as the hyperparameter(s) can be integrated out by using numerical integration. One way of obtaining these approximations is simply using the marginal distributions of $\tilde{\pi}(\boldsymbol{x}|\boldsymbol{y},\tau)$, which is easy to obtain as this is a Gaussian distribution with mean $\boldsymbol{\mu}$ and precision $\boldsymbol{Q}$. The approximated marginal is then $\tilde{\pi}(x_i|\boldsymbol{y},\tau) = \mathcal{N}(x_i|\mu_i,\sigma_i^2)$, where σ_i^2 is the ith diagonal element in the inverse of $\boldsymbol{Q}$. Rue et al. (2009) show that the marginals obtained from the Gaussian distribution computed by the analytical approach does not yield good approximations, and they suggest a way to correct these marginals to improve upon the approximations. Here we will see that the marginals obtained from the Gaussian distribution computed by the expectation propagation algorithm also yield

better approximations. It is also possible to correct the EP computed marginals as discussed by Cseke and Heskes (2010). Figure 6.7 shows the approximated marginal distributions for the fixed effects in (3.1) (recall that $\boldsymbol{\beta}$ is part of the latent field), where $\tau = 0.06$ is fixed. To assess the error in these marginals MCMC simulations were run with a $Gamma(1000 \times 0.06, 1000^2)$ prior for τ (i.e $\mathrm{E}(\tau) = 0.06$ and $\mathrm{var}(\tau) = 0.001$) using the `glmmAK` package. The MCMC results are also displayed in Figure 6.7, and as we can see the marginals obtained by EP corresponds quite well with the MCMC results. The marginals obtained from the analytical Gaussian approximation are on the other hand not so good, especially for β_0 and β_2. The symmetric Kullback-Leibler divergence between the approximated marginals and the MCMC gold standard is given in Table 6.2. This shows that the EP computed marginals offer great improvements compared with the analytical approximation marginals. Of course this result is not very surprising since the expectation propagation method works by optimizing sites parameters with respect to the (pseudo-exact) marginals, as was discussed in Sec. 5.2.

The correction methods used by Rue et al. (2009) for improving the analytical approximation marginals were not considered here, nor the corrections of the EP marginals discussed by Cseke and Heskes (2010). Both of these methods would likely give better fit than what we get from just using the marginals obtained from $\tilde{\pi}(\boldsymbol{x}|\boldsymbol{y}, \tau)$.

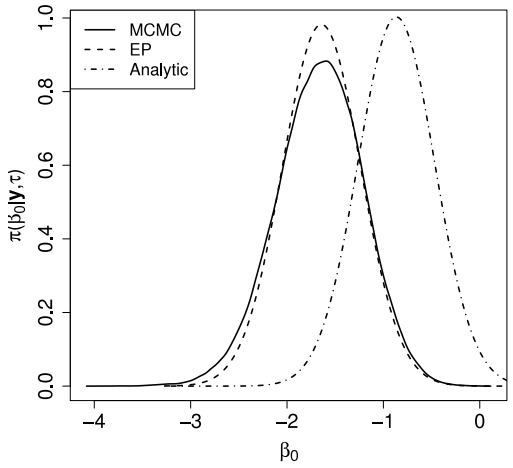
Approximated marginal	$\mathrm{SKL}(\pi_{\mathrm{MCMC}}\|\|\tilde{\pi}_{\mathrm{Analytic}})$	$\mathrm{SKL}(\pi_{\mathrm{MCMC}}\|\|\tilde{\pi}_{\mathrm{EP}})$
$\tilde{\pi}(\beta_0\|\boldsymbol{y}, \tau)$	3.697	0.027
$\tilde{\pi}(\beta_1\|\boldsymbol{y}, \tau)$	0.035	0.005
$\tilde{\pi}(\beta_2\|\boldsymbol{y}, \tau)$	0.913	0.033
$\tilde{\pi}(\beta_3\|\boldsymbol{y}, \tau)$	0.133	0.003

Table 6.2: Symmetric Kullback-Leibler divergence between the approximated and the MCMC computed marginal distributions for β_0, β_1, β_2 and β_3.

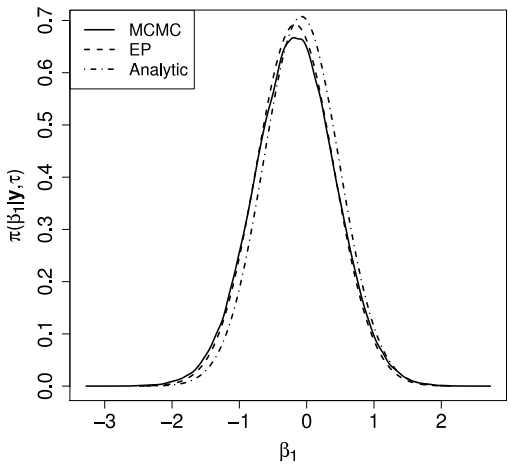
7 Conclusion

The main theme in this text has been to study the expectation propagation algorithm for use in the approximate Bayesian inference scheme for latent Gaussian models, introduced by Rue et al. (2009). The goal has been to compare EP to the analytical approximation used by Rue et al. (2009) for computing an unnormalized Gaussian approximation to the conditional posterior distribution of the latent field. This approximation is an important part of the approximate Bayesian analysis as it is used for approximating posterior distributions for hyperparameters and posterior marginal distributions for the latent field variables. The assumption that the latent field is a Gaussian Markov random field is not particularly common in texts where EP used, e.g. Rasmussen and Williams (2006), and one point that has been studied here is the usage of specialized algorithms for sparse matrices, discussed by Rue and Held (2005), to speed up EP.

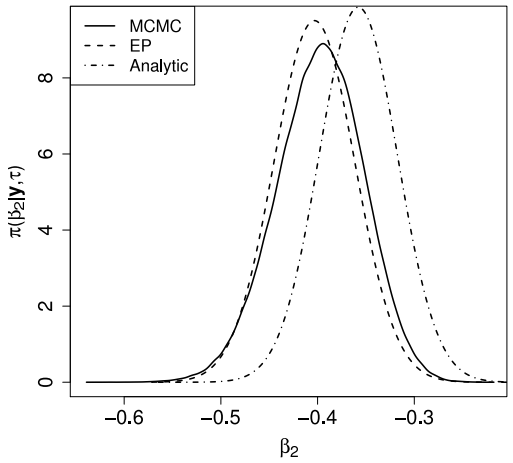
The findings in this analysis are not very surprising nor revolutionary. For the logistic regression case in Sec. 6.2 the expectation propagation algorithm improved



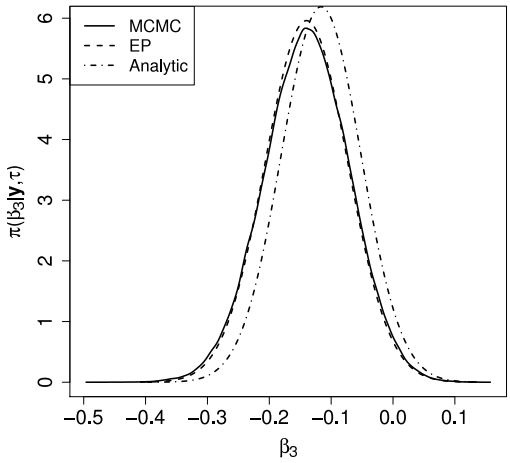
(a) Posterior marginal distribution for the fixed effect β_0



(b) Posterior marginal distribution for the fixed effect β_1



(c) Posterior marginal distribution for the fixed effect β_2



(d) Posterior marginal distribution for the fixed effect β_3

Figure 6.7: The posterior marginal distributions for the fixed effects (6.13). Note that the hyperparameter τ has not been integrated out.

the approximations to $\pi(\tau|\boldsymbol{y})$ slightly compared to the approximation obtained by the analytical approach. However, the EP approximation was still a bit off compared to the long MCMC simulation gold standard. On the other hand, the marginal distributions obtained from the EP Gaussian approximation $\tilde{\pi}(\boldsymbol{x}|\boldsymbol{y},\tau)$ was a great improvement to the corresponding marginals obtained from the analytical approximation. By correcting these approximations further, for instance by the methods suggested by Cseke and Heskes (2010), the EP computed marginals can probably be made quite exact. Though, the *integrated nested Laplace* approximations used by Rue et al. (2009) already does this job quite well and at a lower computational cost.

The cost factor is really the essential draw back with EP compared to the analytical method. The test of computational time use for both algorithms on simple (almost trivial) problems suggests that EP is *at least* a factor of ten times slower than the analytical method. Of course, there are probably numerous ways to speed up both algorithms so the results are not definite. However, this indicates that unless the analytical approach is known to perform poorly for a certain model, this method is preferable to EP, at least if time is an issue.

7.1 Suggestions for further studies

As discussed throughout this text the main issue with the expectation propagation algorithm is speed. One option for making improved approximations by using EP is to first compute the analytical approximation and use this as an initial condition for EP. By only performing a small number of iteration steps in the EP algorithm, the approximation will be improved and the computational time use is likely to be reasonably low. This has been suggested by Cseke and Heskes (2010) as an alternative to the sole use of the analytical approximation in the approximate Bayesian inference scheme. It is also interesting to look at, as has been done in Cseke and Heskes (2010), corrected approximated posterior marginals of elements of the latent field and to consider models where the analytical approximation fails (e.g. models with Laplace likelihoods). The expectation propagation algorithm is a valuable contribution to the field of approximate Bayesian inference and further studies of EP for use in the analysis methods introduced by Rue et al. (2009) should probably use the recent work of Cseke and Heskes (2010) as a starting point.

A Kullback-Leibler divergence

The Kullback-Leibler divergence is a pseudo-metric for probability distributions. If $\pi_1(\boldsymbol{x})$ and $\pi_2(\boldsymbol{x})$ are two probability density functions, the Kullback-Leibler divergence $\mathrm{KL}(\pi_1\|\pi_2)$ is defined as

$$\mathrm{KL}(\pi_1\|\pi_2) = \mathrm{E}_{\pi_1}\left[\log\frac{\pi_1(\boldsymbol{x})}{\pi_2(\boldsymbol{x})}\right] = \int \log\frac{\pi_1(\boldsymbol{x})}{\pi_2(\boldsymbol{x})}\pi_1(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \tag{A.1}$$

The KL-divergence share some of the characteristics of a metric since

$$\mathrm{KL}(\pi_1\|\pi_2) \geq 0, \quad \mathrm{KL}(\pi_1\|\pi_2) = 0 \iff \pi_1 = \pi_2 \text{ almost everywhere.}^{10}$$

The symmetry condition, however, do not hold as $\mathrm{KL}(\pi_1\|\pi_2) \neq \mathrm{KL}(\pi_2\|\pi_1)$, clearly from equation (A.1). Also the triangle inequality is not satisfied for this measure. To remedy the symmetry issue it is possible to define the symmetric KL-divergence between π_1 and π_2 as

$$\mathrm{SKL}(\pi_1\|\pi_2) = \mathrm{KL}(\pi_1\|\pi_2) + \mathrm{KL}(\pi_2\|\pi_1),$$

which is used by Rue et al. (2009) as a measure of agreement between distributions.

A useful result with the KL-divergence, also mentioned by Rasmussen and Williams (2006), is that if π_1 is a general distribution and π_2 is a $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ distribution, the minimal Kullback-Leibler divergence $\mathrm{KL}(\pi_1\|\pi_2)$ is achieved when $\boldsymbol{\mu} = \mathrm{E}_{\pi_1}(\boldsymbol{x})$ and $\boldsymbol{\Sigma} = \mathrm{cov}_{\pi_1}(\boldsymbol{x})$, i.e. when the first and second order moments of π_2 equals those of π_1.

References

Abramowitz, M. and Stegun, I. A. (1964), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, ninth dover printing, tenth gpo printing edn, Dover, New York.

Cseke, B. and Heskes, T. (2010), Improving posterior marginal approximations in latent Gaussian models, *in* Y. W. Teh and M. Titterington, eds, 'Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics', Vol. 9, pp. 121–128.

Gamerman, D. and Lopes, H. F. (2006), *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*, second edn, Chapman & Hall, Boca Raton, FL.

Golub, G. H. and Van Loan, C. F. (1996), *Matrix Computations*, third edn, The Johns Hopkins University Press, Baltimore, MD.

Johnson, R. A. and Wichern, D. W. (2007), *Applied Multivariate Statistical Analysis*, sixth edn, Pearson Prentice Hall, Upper Saddle River, NJ.

Kingman, J. and Taylor, S. (2008), *Introduction to Measure and Probability*, Cambridge University Press, Cambridge.

Komarek, A. (2010), *glmmAK: Generalized Linear Mixed Models*. R package version 1.4. Available from `http://CRAN.R-project.org/package=glmmAK`

MacKay, D. J. (2003), *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, Cambridge.

[10] To prove this use Jensens inequality: $\mathrm{E}\, g(u) \geq g(\mathrm{E}\, u)$ for convex function $g(\cdot)$. Let $g(u) = 1/\log u$, $u = \pi_2(\boldsymbol{x})/\pi_1(\boldsymbol{x})$ and use expecatation with respect to the probability measure defined by π_1. See MacKay (2003).

Martin, R. S., Peters, G. and Wilkinson, J. H. (1965), 'Symmetric decomposition of a positive definite matrix', *Numerische Mathematik* **7**(5), 362–383.

Minka, T. P. (2001), A family of algorithms for approximate Bayesian inference, PhD thesis, Massachusetts Institute of Technology.

R Development Core Team (2010), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. Available from `http://www.R-project.org`

Rasmussen, C. E. and Williams, C. K. I. (2006), *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA.

Rue, H. and Held, L. (2005), *Gaussian Markov Random Fields: Theory and Applications*, Vol. 104 of *Monographs on Statistics and Applied Probability*, Chapman & Hall, London.

Rue, H. and Martino, S. (2007), 'Approximate Bayesian inference for hierarchical Gaussian Markov random field models', *J. Statist. Plann. Inference* **137**(10), 3177–3192.

Rue, H., Martino, S. and Chopin, N. (2009), 'Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations', *J. R. Statist. Soc. B* **71**(2), 319–392.

Saad, Y. (2003), *Iterative Methods for Sparse Linear Systems*, second edn, SIAM, Philadelphia, PA.

Strang, G. (2006), *Linear Algebra and its Applications*, Thomson Brooks/Cole, Belmont, CA.

Tierney, L. and Kadane, J. B. (1986), 'Accurate approximations for posterior moments and marginal densities', *J. Am. Statist. Ass.* **81**(393), 82–86.