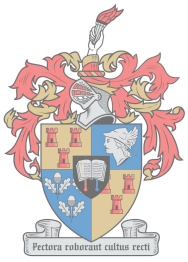
Regularised Gaussian Belief Propagation

by

Francois Kamper



Dissertation presented for the degree of Doctor of Philosophy in Mathematical Statistics in the Faculty of Economic and Management Sciences at Stellenbosch University

Supervisor: Prof. S.J. Steel

Co-supervisor: Prof. J.A. du Preez

December 2018

The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NRF.

Declaration

By submitting this dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: December 2018

Abstract

Regularised Gaussian Belief Propagation

F. Kamper

Dissertation: PhD

December 2018

Belief propagation (BP) has been applied as an approximation tool in a variety of inference problems. BP does not necessarily converge in loopy graphs and, even if it does, is not guaranteed to provide exact inference. Even so, BP is useful in many applications due to its computational tractability. On a high level this dissertation is concerned with addressing the issues of BP when applied to loopy graphs. To address these issues we formulate the principle of node regularisation on Markov graphs (MGs) within the context of BP. The main contribution of this dissertation is to provide mathematical and empirical evidence that the principle of node regularisation can achieve convergence and good inference quality when applied to a MG constructed from a Gaussian distribution in canonical parameterisation. There is a rich literature surrounding BP on Gaussian MGs (labelled Gaussian belief propagation or GaBP), and this is known to suffer from the same problems as general BP on graphs. GaBP is known to provide the correct marginal means if it converges (this is not guaranteed), but it does not provide the exact marginal precisions. We show that our regularised BP will, with sufficient tuning, always converge while maintaining the exact marginal means. This is true for a graph where nodes are allowed to have any number of variables. The selection of the degree of regularisation is addressed through the use of heuristics. Our variant of GaBP is tested empirically in a variety of settings. We show that our method outperforms other variants of GaBP available in the literature, both in terms of convergence speed and quality of inference. These improvements suggest that the principle of node regularisation in BP should be investigated in other inference problems. A by-product of GaBP is that it can be used to solve linear systems of equations; the same is true for our variant and in this context we make an empirical comparison with the conjugate gradient (CG) method.

Uittreksel

Geregulariseerde Gaussiese Geloofspropagasie

("Regularised Gaussian Belief Propagation")

F. Kamper

Proefskrif: PhD

Desember 2018

Geloofspropagasie word in 'n verskeidenheid van inferensie-probleme as 'n benaderings-gereedskap toegepas. Geloofspropagasie konvergeer nie noodwendig in grafieke met sirkelroetes nie, en selfs al doen dit, is dit nie gewaarborg om korrekte inferensie te verskaf nie. Ten spyte hiervan is geloofspropagasie nuttig in baie toepassings as gevolg van die goeie berekenings-aspekte daarvan. Op 'n hoë vlak is hierdie tesis gefokus op die verbetering van geloofspropagasie wanneer dit op grafieke met sirkelroetes toegepas word. Om hierdie verbeteringe te bereik, word die beginsel van nodusregulering in die konteks van Markov-grafieke (MG'e) geformuleer. Die hoof bydrae van hierdie tesis is om wiskundige en empiriese bewyse te verskaf dat die beginsel van nodusregulering konvergensie kan bewerkstellig en goeie inferensie-kwaliteit kan verskaf wanneer die MG saamgestel is uit 'n Gaussiese verdeling in kanoniese vorm. Daar is 'n ryk literatuur rondom hierdie tipe geloofspropagasie (Gaussiese geloofspropagasie of GaBP) en dit is bekend dat GaBP dieselfde probleme as algemene geloofspropagasie ervaar. GaBP verskaf die korrekte marginale gemiddeldes by konvergensie (konvergensie is nie gewaarborg nie), maar verskaf nie noodwendig die korrekte marginale presisies nie. Ons wys dat ons aangepaste geloofspropagasie altyd konvergeer, gegewe voldoende regulering, terwyl dit steeds die korrekte marginale gemiddeldes verskaf. Hierdie is waar vir enige grafiek, ongeag die aantal veranderlikes per nodus. Die keuse van die graad van regulering word deur heuristieke metodes aangespreek. Ons GaBP metode word in 'n verskeidenheid van situasies empiries getoets. In hierdie situasies vaar ons metode beter as ander metodes in die literatuur, beide in terme van konvergensie-spoed en die kwaliteit van die inferensie. Hierdie verbeteringe stel voor dat die beginsel van nodusregulering in geloofspropagasie in ander inferensie-probleme ondersoek moet word. 'n Byproduk van GaBP is dat dit gebruik kan word om stelsels van lineêre vergelykings op te los; dieselfde is waar

in die geval van ons metode en ons maak binne hierdie konteks 'n empiriese vergelyking met die toegevoegde gradiënt (CG)-metode.

Acknowledgements

Ek wil begin deur professor S.J. Steel te bedank vir sy studieleiding en mentorskap gedurende my lewe as 'n student. Ek waardeer al die tyd en moeite wat hy aan my ontwikkeling spandeer het, beide as 'n mens en 'n statistikus. Die vreugde wat hy kry deur nuwe en opwindende dinge te leer is 'n aansteeklike bron van inspirasie vir al sy studente. Sy leiding, sagte geaardheid en ondersteuning oor al die jare word opreg waardeer. Ek bedank professor J.A. du Preez vir die voorreg om hom oor die afgelope paar jare te leer ken het. Ek is veral dankbaar dat hy my voorgestel het aan die opwindende veld van probabilistiese grafiese modelle. Sy hulp en insette gedurende my PhD-studies kan nie oorskat word nie.

Ek wil professor S. Wagner bedank vir die bydrae wat hy gelewer het tot die wiskundige afleidings in hierdie werkstuk. Sy insigte in verband met die gedrag van eiewaardes het die hoeksteen vir die konvergensie-bewyse in hierdie werkstuk gevorm.

I would like to thank my dissertation examiners Dr S. Kroon, Dr M. Burke and Prof F. Lombard for their helpful and constructive criticism. Their contribution to the improvement of this dissertation cannot be overestimated.

I am grateful to professor N. Sebe and his team, for the time I was able to spend at the university of Trento. Under his supervision I was able to investigate practical applications of the algorithms presented in this dissertation.

Ek wil dankie sê vir al my vriende wat my deur my studies gehelp het. Aan Sven, vir die lekker chats oor stats en dat jy my altyd laat lag. Aan Zander, vir al die kuiers en gesamentlike ondersteuning van die beste klub in die wêreld (Liverpool). Aan Rassie, vir 'n lekker vriendskap, goeie gesprekke, Skype chats en die beste kuiers. Aan Simon, vir 'n vriendskap wat so vinnig gegroei het. En aan al die DSP-dwellers, wat my vinnig laat tuis voel het.

I would like to thank *The Doors* for their music, which helped me to stay calm during times of stress.

Ek wil graag afsluit deur my familie te bedank vir al hul liefde, hulp en bystand. Vir mams en paps, wat altyd beskikbaar was wanneer ek iemand nodig gehad het om mee te praat. Vir Herman, wat altyd daar was vir raad gee. Vir Femia, wat elke aand vir my boodskappe skryf. Vir Marina, wie se glimlag vir my so baie beteken. Vir Helena, wat so 'n fantastiese vrou vir my boetie is. En die belangrikste - vir my vrou, Andrea, vir wie ek ontsaglik lief

is, dat sy elke dag daar was vir my, liefde gegee het en nooit in my getwyfel het nie.

"Great are the works of the LORD, studied by all who delight in them." Thank you for the opportunity to delight in your great works.

Aan Andrea, my inspirasie vir die lewe.

Contents

List of Figures

List of Tables

List of Abbreviations

BP belief propagation. ii, 1, 4–8, 12, 16, 28, 32, 41, 76, 100, 102–105

CFGaBP convergence fix Gaussian belief propagation. xi, 14–16, 40–43, 92–96

CG conjugate gradient. ii, iv, xi, 22, 35, 42–47, 64, 72–76, 102, 103

FGaBP feedback Gaussian belief propagation. 14, 15

FMP feedback message passing. 14, 15

GaBP Gaussian belief propagation. ii, iii, viii, x, 6–19, 21, 22, 25, 29, 32, 33, 35, 36, 38, 40–42, 46, 48, 50, 51, 63–65, 69–76, 78, 79, 88, 93, 94, 96, 100–105, 114, 117

GDH gradient descent heuristic. xi, 30–33, 45

KL Kullback-Leibler. 29, 35–38, 40, 41, 43, 44, 46, 94, 95, 103, 104

MG Markov graph. ii, iii, xi, 1–5, 7, 8, 12, 13, 46, 48, 49, 51–54, 56, 70, 72, 77, 100, 103, 127

PCG preconditioned conjugate gradient. 72–74, 102, 103

PGM probabilistic graphical model. 1, 4

ReGaBP reweighted Gaussian belief propagation. 14

RGaBP relaxed Gaussian belief propagation. xi, 15, 16, 36–40, 45–47, 87, 92–98

RIP running intersection property. 104

sGaBP slow Gaussian belief propagation. x, xi, 16–18, 20–25, 27–32, 34–47, 77–79, 83, 85, 87–90, 92–105, 114, 127

SH search heuristic. xi, 30–33

UWT unwrapped trees. ix, xi, 46, 48, 51–56, 60, 64–66, 68, 70, 71, 76, 78–80, 88–91, 116–118, 120, 127–130

Chapter 1

Introduction

The main goal of this chapter is to provide the reader with a high-level understanding of the topics covered in this dissertation before delving into specifics in the subsequent chapters. The scope can be narrowed to the application of belief propagation (BP) to probabilistic graphical models (PGMs). We start with a discussion of these fundamentals.

1.1 Probabilistic Graphical Models

PGMs are used to represent the conditional dependence between random variables in the form of a graph. PGMs can be used to encode complex probability distributions compactly and allow the user to exploit the conditional dependence structure of a random vector to perform tasks efficiently (even in some otherwise intractable tasks). Consider for instance the task of assigning values to a vector of binary random variables $\mathbf{X} : k \times 1$ given evidence E. The maximum a posteriori (MAP) assignment is

$$\mathbf{x}_{\text{MAP}} = \underset{\mathbf{x}}{\text{argmax}}\{\text{Prob}(\mathbf{X} = \mathbf{x}|E)\}, \tag{1.1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_k)'$ is such that $x_i \in \{0, 1\}$. The brute-force approach would be to evaluate $\text{Prob}(\mathbf{X} = \mathbf{x}|E)$ for all 2^k possible $\mathbf{x}$, a computationally intractable task even for moderate k. A PGM would attempt to exploit conditional independence characteristics of $\text{Prob}(\mathbf{X} = \mathbf{x}|E)$ to perform the inference given in Equation (1.1) in a tractable way, using one of a variety of algorithms.

In this dissertation we focus on Markov graphs (MGs), also known as undirected graphs or Markov random fields. A MG consists of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. Each node in $\mathcal{V}$ is represented by a random variable, while the set of edges $\mathcal{E}$ defines which nodes are linked in the graph. Consider two nodes i, j, where $i \neq j$, and let $m = \min(i, j)$ and $n = \max(i, j)$. Nodes i and j are linked if and only $(m, n) \in \mathcal{E}$. Figure 1.1 contains two examples of a MG for the random vector $\mathbf{X} = (X_1, X_2, X_3, X_4, X_5, X_6)'$. The set of edges for the sparsely

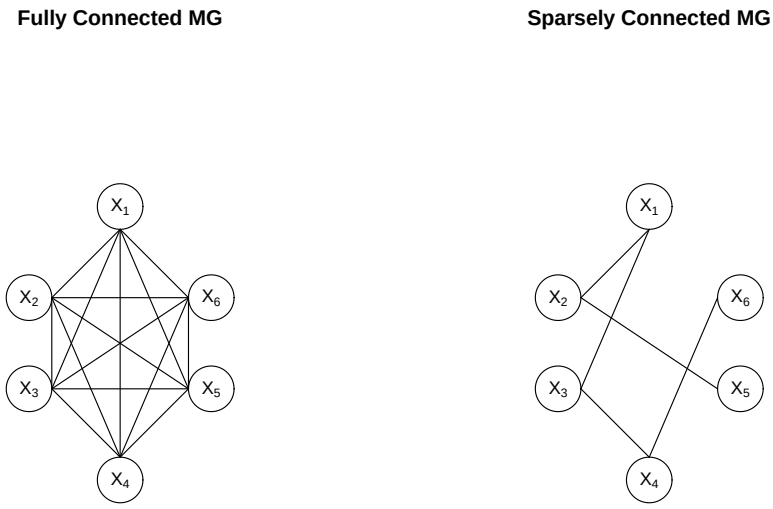


Figure 1.1: Illustration of two MGs for the random vector $\mathbf{X} = (X_1, X_2, X_3, X_4, X_5, X_6)'$. The left side shows a fully connected MG, while the right side shows a sparsely connected graph.

connected graph in Figure 1.1 is $\mathcal{E} = \{(1,2);(1,3);(2,5);(3,4);(4,6)\}$. The neighbourhood of node i is denoted by $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\} \cup \{j : (j,i) \in \mathcal{E}\}$. For the fully connected graph in Figure 1.1, we have $\mathcal{N}_3 = \{1,2,4,5,6\}$, while for the sparsely connected graph, $\mathcal{N}_3 = \{1,4\}$.

The structure of a MG can be used to infer certain conditional independencies of the random vector it represents. Consider a MG, $\mathcal{G}$, for the random vector $\mathbf{X} = (X_1, X_2, \ldots, X_k)'$. The univariate variables X_i, X_j, where $i < j$, are conditionally independent given the other variables if $(i,j) \notin \mathcal{E}$ (these are called the pairwise independencies of the graph). In the sparsely connected graph of Figure 1.1, we see that the random variables X_1 and X_6 are conditionally independent given the other random variables. Let $\mathbf{X}_1$, $\mathbf{X}_2$ and $\mathbf{X}_3$ be non-overlapping subvectors of $\mathbf{X}$. The vectors $\mathbf{X}_1$ and $\mathbf{X}_2$ are conditionally independent given $\mathbf{X}_3$ if every path between the nodes representing $\mathbf{X}_1$ and the nodes representing $\mathbf{X}_2$ intersect a node representing a variable in $\mathbf{X}_3$. These are called the global Markov independencies of $\mathcal{G}$. For instance, in the sparsely connected graph of Figure 1.1, we see that the random variable X_5 and the random vector $(X_3, X_4)'$ are conditionally independent given $(X_1, X_2)'$.

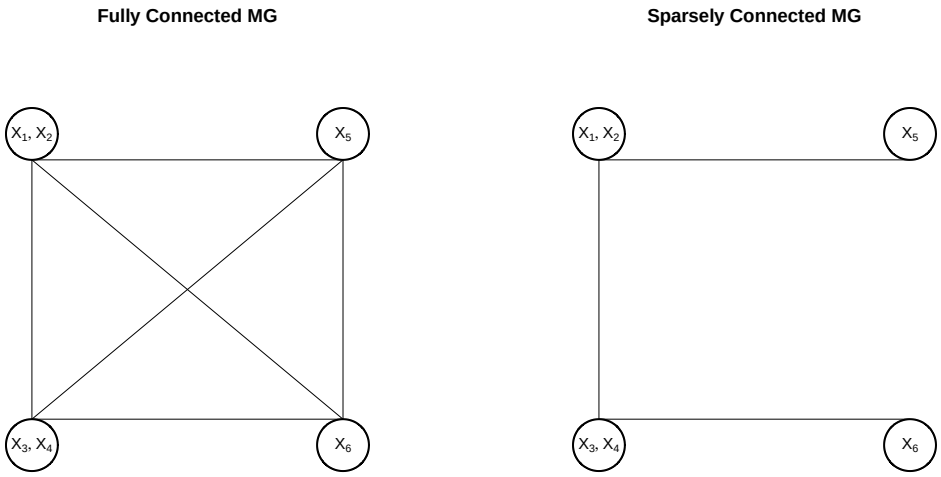


Figure 1.2: Illustration of clustering applied to the two MGs displayed in Figure 1.1. The clusters are $(X_1, X_2)'; (X_3, X_4)'; X_5$ and X_6.

We also allow for MGs where nodes can receive more than one variable. Consider the partition $\mathbf{X} = (\mathbf{X}_1', \mathbf{X}_2', \ldots, \mathbf{X}_p')'$, where $\mathbf{X}_t : t = 1, 2, \ldots, p$ are non-overlapping subvectors of $\mathbf{X}$. Suppose we are interested in the pairwise independencies of these subvectors, i.e. are $\mathbf{X}_t$ and $\mathbf{X}_s$ conditionally independent, given the other subvectors? From the global Markov independencies we can define a new graph, $\tilde{\mathcal{G}}$, with nodes $\tilde{\mathcal{V}} = \{1, 2, \ldots, p\}$ and $(s, t) \in \tilde{\mathcal{E}}$ if, and only if, no variable in $\mathbf{X}_s$ has a edge with a variable in $\mathbf{X}_t$ in $\mathcal{G}$. The pairwise independencies of the partition can be inferred from $\tilde{\mathcal{G}}$ in the same way we inferred the pairwise independencies of the univariate random variables from $\mathcal{G}$. We refer to the process of creating a partition as clustering. When confronted with clustering, the term MG refers to $\tilde{\mathcal{G}}$. The MGs for the graphs displayed in Figure 1.1, with the clusters $(X_1, X_2)'; (X_3, X_4)'; X_5$ and X_6, are given in Figure 1.2. For the sparsely connected graph, we see that (X_1, X_2) and X_6 are conditionally independent given the other variables. This figure also displays the assertion that the random variable X_5 and the random vector $(X_3, X_4)'$ are conditionally independent given $(X_1, X_2)'$ more clearly.

When performing clustering, we denote the indices of the variables represented by cluster i as $\mathcal{C}_i$. For the clustering done in Figure 1.2 we see that $\mathcal{C}_1 = \{1, 2\}, \mathcal{C}_2 = \{3, 4\}, \mathcal{C}_3 = \{5\}$ and $\mathcal{C}_4 = \{6\}$. Let $\text{vec}(\mathcal{C}_i)$ be the vector of the indices in $\mathcal{C}_i$. We assume, without loss of generality, that

$(\text{vec}(\mathcal{C}_1)', \text{vec}(\mathcal{C}_2)', \ldots, \text{vec}(\mathcal{C}_p)')' = (1, 2, \ldots, k)'$ (if this does not hold the variables can be reordered to do so).

In the next section we discuss a certain type of BP that exploits the pairwise and global independencies of MGs to obtain computational advantages for the task of performing inference.

1.2 Belief Propagation

Belief propagation (BP) is one of the algorithms that can be used to perform inference on PGMs. Informally speaking, BP is a message-passing algorithm that communicates between the nodes of a PGM. The critical advantage is that direct communication between unlinked nodes is not necessary. Hence the more sparse the graph, the more efficient the BP. The BP algorithm iteratively constructs messages between linked nodes until the messages converge. At this point a node collects all incoming messages and uses these to create a potential. This potential is used to perform inference.

In the context of error-correcting codes, the roots of BP can be traced back to the development of the sum-product algorithm as a decoding algorithm for LDPC codes (Gallager, 1963). Belief propagation (probability propagation) for Bayesian networks was introduced by Pearl (1988); Shachter (1988); Shafer and Shenoy (1990); and Lauritzen and Spiegelhalter (1988). BP was found (at a later stage) to be equivalent to the sum-product algorithm (Frey and Kschischang, 1996; Aji and McEliece, 2000). Perhaps the most successful application of BP can be found in the development of turbo codes, with performances in terms of bit error rate (BER) approaching that of the Shannon limit (Murphy *et al.*, 1999).

BP on tree-structured graphs is known to converge and provide exact inference at convergence (Pearl, 1988). The inference quality of BP can be severely compromised when it is applied to loopy graphs (Weiss and Freeman, 2001). A graph is said to contain a loop if there exists a path in the graph that connects a node to itself. When a graph has loops, BP does not necessarily converge, and even if it does, is not guaranteed to provide exact inference. Even so, BP is useful in many applications due to its computational tractability, particularly in cases where the inference quality of BP is acceptable (as in the case of turbo codes).

In this dissertation we focus on BP applied to MGs where the underlying density function has a specific factorisation. We discuss this factorisation in the next section.

1.2.1 Formulation

Let us consider a random vector $\mathbf{X} : k \times 1$ with density function $f(\mathbf{x})$. Our task is to perform inference on $f(\mathbf{x})$ where it is assumed that,

$$f(\mathbf{x}) = \frac{1}{Z} \prod_{i \in \mathcal{V}} \psi_{ii}(\mathbf{x}_i) \prod_{(i,j) \in \mathcal{E}} \psi_{ij}(\mathbf{x}_i, \mathbf{x}_j), \tag{1.2}$$

where Z is a normalising constant, the $\mathbf{x}_i$'s are mutually exclusive and exhaustive subvectors of $\mathbf{x} = (\mathbf{x}_1', \mathbf{x}_2', \ldots, \mathbf{x}_p')'$ and $\mathcal{V} = \{1, 2, \ldots, p\}$ where $p \leq k$. Consider the MG obtained by partitioning $\mathbf{X}$ according to the partition $\mathbf{x} = (\mathbf{x}_1', \mathbf{x}_2', \ldots, \mathbf{x}_p')'$. A consequence of the Hammersley-Clifford theorem (Hammersley and Clifford, 1971; Clifford, 1990) is that the MG for the partition will have nodes $\mathcal{V}$ and edges $\mathcal{E}$. Note that if $(i, j) \in \mathcal{E}$ we implicitly assume that $i < j$.

When the nodes are one-dimensional ($p = k$), the factorisation in Equation (1.2) yields a pairwise MG. There are many ways of formulating BP on the factorisation given in Equation (1.2) (factor graphs or cluster graphs, for example). In this dissertation we focus on a multivariate extension of BP on pairwise MGs.

Let $\mathcal{N}_i \setminus j$ be the set $\mathcal{N}_i$ with element j removed (the assumption that $j \in \mathcal{N}_i$ is made implicitly). BP can be formulated using the sum-product or max-sum rule. For the sum-product rule, BP seeks to perform iterative updates to obtain the stationary conditions,

$$m_{ij}(\mathbf{x}_j) = \int \psi_{ii}(\mathbf{x}_i) \psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) \prod_{t \in \mathcal{N}_i \setminus j} m_{ti}(\mathbf{x}_i) d\mathbf{x}_i, \tag{1.3}$$

for all $i \in \mathcal{V}$ and all $j \in \mathcal{N}_i$. For the max-sum rule, this changes to

$$m_{ij}(\mathbf{x}_j) = \max_{\mathbf{x}_i} \left\{ \log(\psi_{ii}(\mathbf{x}_i)) + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) + \sum_{t \in \mathcal{N}_i \setminus j} m_{ti}(\mathbf{x}_i) \right\}, \tag{1.4}$$

for all $i \in \mathcal{V}$ and all $j \in \mathcal{N}_i$. Note that the max-sum formulation given in Equation (1.4) can easily be changed to a minimisation problem. In both Equation (1.3) and Equation (1.4) we have $\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \psi_{ji}(\mathbf{x}_j, \mathbf{x}_i)$ when $i > j$.

1.2.2 Message Updates

Bickson (2008) describes two conventional types of message-update rules that can be used for the purpose of iterating equations to achieve the stationary conditions given in Equation (1.3) or Equation (1.4). In synchronous message passing, new messages are formed using messages from the previous round only

and are therefore not influenced by the message scheduling. This is in contrast to the asynchronous case, where messages updated in the current round are used to compute new messages. Although asynchronous updates tend to outperform the synchronous approach (Koller and Friedman, 2009), our main focus is on the synchronous case. We do this since one of the more attractive properties of GaBP is its application in distributive settings, which is far more compatible with synchronous message updates. Synchronous implementation also allows us to compare different GaBP algorithms without considering the effects of different message schedulings. For the sum-product formulation, the synchronous message updates will be,

$$m_{ij}^{(n+1)}(\mathbf{x}_j) = \int \psi_{ii}(\mathbf{x}_i)\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) \prod_{t \in \mathcal{N}_i \setminus j} m_{ti}^{(n)}(\mathbf{x}_i) d\mathbf{x}_i, \tag{1.5}$$

while for the max-sum we have:

$$m_{ij}^{(n+1)}(\mathbf{x}_j) = \max_{\mathbf{x}_i} \left\{ \log(\psi_{ii}(\mathbf{x}_i)) + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) + \sum_{t \in \mathcal{N}_i \setminus j} m_{ti}^{(n)}(\mathbf{x}_i) \right\}, \tag{1.6}$$

for all $i \in \mathcal{V}$ and all $j \in \mathcal{N}_i$. The sum-product and max-sum formulation of BP are related in the sense that they can both be used to perform approximate inference, although they may provide different approximations. We note that if the density function given in Equation (1.2) is a Gaussian density function, then the message updates given in Equations (1.5) and (1.6) are equivalent.

1.2.3 Posterior Distributions

We associate with each $i \in \mathcal{V}$ the prior distribution $\psi_{ii}(\mathbf{x}_i)$. After iteration n we have messages $m_{ij}^{(n)}(\mathbf{x}_j)$ for all $i \in \mathcal{V}$ and all $j \in \mathcal{N}_i$. Within the context of BP, we can construct a posterior distribution for node i at iteration n using

$$\text{post}_i^{(n)}(\mathbf{x}_i) = \psi_{ii}(\mathbf{x}_i) \prod_{t \in \mathcal{N}_i} m_{ti}^{(n)}(\mathbf{x}_i) \tag{1.7}$$

and

$$\text{post}_i^{(n)}(\mathbf{x}_i) = \psi_{ii}(\mathbf{x}_i) \prod_{t \in \mathcal{N}_i} \exp\left\{ m_{ti}^{(n)}(\mathbf{x}_i) \right\}. \tag{1.8}$$

for the sum-product and max-sum formulations respectively. Essentially, the distribution given in Equation (1.7) (or Equation (1.8)), after appropriate normalisation, provides an approximation for the true marginal density function of the variables in $\mathbf{x}_i$. Consider the max-sum formulation and let

$$\text{post}_{i\setminus j}^{(n)}(\mathbf{x}_i) = \psi_{ii}(\mathbf{x}_i) \prod_{t \in \mathcal{N}_i \setminus j} \exp\left\{ m_{ti}^{(n)}(\mathbf{x}_i) \right\}, \tag{1.9}$$

then

$$m_{ij}^{(n+1)}(\mathbf{x}_j) = \max_{\mathbf{x}_i} \left\{ \log(\text{post}_{i\setminus j}^{(n)}(\mathbf{x}_i)) + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) \right\}. \tag{1.10}$$

Equation (1.10) provides a good basis for motivating our high-level approach, which we discuss in the next section.

1.2.4 High-level Approach of this Dissertation

The focus of this dissertation can be (partially) narrowed to addressing the convergence shortcomings of BP when applied to loopy MGs. The idea is to develop an adjusted BP variant that is able to perform satisfactory inference on loopy MGs with an arbitrary conditional independence structure. To this end we propose a regularised variant of Equation (1.6) (we give a sum-product variant in Section 1.4.3):

$$\begin{aligned} m_{ij}^{(n+1)}(\mathbf{x}_j) &= \max_{\mathbf{x}_i} \left\{ \log(\psi_{ii}(\mathbf{x}_i)) + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) + \sum_{t \in \mathcal{N}_i \setminus j} m_{ti}^{(n)}(\mathbf{x}_i) \right. \\ &\quad \left. - \frac{\lambda}{q} ||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_q^q \right\} \\ &= \max_{\mathbf{x}_i} \left\{ \log(\text{post}_{i\setminus j}^{(n)}(\mathbf{x}_i)) - \frac{\lambda}{q} ||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_q^q + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) \right\} \end{aligned} \tag{1.11}$$

The value $\boldsymbol{\theta}_i^{(n-1)}$ in Equation (1.11) is selected at iteration $n-1$, and λ is a regularisation parameter. Essentially we are encouraging $\mathbf{x}_i$ to take values closer to $\boldsymbol{\theta}_i^{(n-1)}$ at iteration n for the purpose of constructing the messages to neighbouring nodes at iteration $n+1$. The purpose of the max-sum variant given in Equation (1.11) is to provide a method that will converge for any MG, while providing satisfactory inference quality. In cases where basic BP converges, this method should provide comparable results (hopefully improving), both in terms of convergence speed and inference quality. This dissertation focuses on discussing these aspects for a special MG, i.e. one induced by a multivariate Gaussian distribution. The Gaussian assumption simplifies certain technical aspects (such as conjugacy of the messages) and provides a convenient starting point for testing our high-level approach. It is our viewpoint that, if the propagation given in Equation (1.11) proves successful for the Gaussian case, there is no reason why it should not be able to generalise to other MGs and indeed general BP on graphs.

BP on a MG induced by a multivariate Gaussian distribution is often referred to as Gaussian belief propagation (GaBP). On a high level, this dissertation is concerned with addressing the problems underlying the application of BP to

general loopy MGs using the Gaussian assumption as an example. It should be noted that studying GaBP is worthwhile in its own right and there is significant literature available on this subject. This is reviewed in the next section. Although this dissertation is focused heavily on GaBP, we do not want the reader to forget the high-level approach and the link to BP in general. We highlight some focus points of this dissertation:

- Providing mathematical details and proofs for validating our high-level approach (such as proving convergence).
- Providing empirical evidence for statements not explicitly proven.
- Empirical comparisons of our GaBP variant with other variants from the literature.
- Proposing novel ways of utilising the GaBP algorithm.

In the next section we discuss aspects of GaBP and conduct a literature review of the topic.

1.3 Gaussian Belief Propagation

GaBP is a specialised application of BP on a Markov graph assuming a multivariate Gaussian random variable in canonical parameterisation. From this point onwards we denote the precision matrix of a $k \times 1$ Gaussian random variable by $\mathbf{S} : k \times k$ and the potential vector by $\boldsymbol{b} : k \times 1$. Consider the set of nodes $\mathcal{V}$ where $|\mathcal{V}| = p \leq k$. Each node i in $\mathcal{V}$ is associated with a cluster of variables $\mathcal{C}_i \subseteq \{1, 2, \ldots, k\}$, such that the $\mathcal{C}_i$'s are mutually exclusive and $\cup_{i \in \mathcal{V}} \mathcal{C}_i = \{1, 2, \ldots, k\}$.

Notation 1 *Consider the precision matrix* $\mathbf{S}$ *and the clusters* $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_p$ *assigned to nodes. Set* $d_i = |\mathcal{C}_i|$. *By* $\mathbf{S}_{ij} : d_i \times d_j$ *we mean the submatrix of* $\mathbf{S}$ *corresponding to the variables in* $\mathcal{C}_i$ *for the rows and* $\mathcal{C}_j$ *for the columns. The vector* $\boldsymbol{b}_i : d_i \times 1$ *is the subvector of* $\boldsymbol{b}$ *corresponding to the variables in* $\mathcal{C}_i$.

Notation 2 *For a Gaussian distribution with precision matrix* $\mathbf{S}$, *the set of edges is defined to be* $\mathcal{E} = \{(i, j) : i < j \text{ and } \mathbf{S}_{ij} \neq \mathbf{0}\}$. *The neighbourhood of node* i *simplifies to* $\mathcal{N}_i = \{j \neq i : \mathbf{S}_{ij} \neq \mathbf{0}\}$.

In terms of Equation (1.2) we can write:

$$\begin{aligned} \psi_{ii}(\mathbf{x}_i) &= \exp\left[-\frac{1}{2}\mathbf{x}_i'\mathbf{S}_{ii}\mathbf{x}_i + \mathbf{x}_i'\boldsymbol{b}_i \right] : i \in \mathcal{V} \\ \psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) &= \exp\left[-\mathbf{x}_i'\mathbf{S}_{ij}\mathbf{x}_j \right] : (i, j) \in \mathcal{E}. \end{aligned} \tag{1.12}$$

We can now apply the max-sum rule to formulate the message updates for the GaBP algorithm. The max-sum and sum-product rules are equivalent in the Gaussian case. See, for instance, Bickson (2008) for the equivalence of sum-product GaBP and max-product GaBP (identical to max-sum) where nodes are one-dimensional. A proof for this equivalence in the multivariate case is given implicitly in Section 1.4.3 (the proof can be obtained by setting $\lambda = 0$ in the derivation).

1.3.1 Message Updates

We proceed under the assumption that $m_{ij}^{(n)}(\mathbf{x}_j) = K_{ij}^{(n)} - \frac{1}{2}\mathbf{x}_j'\mathbf{Q}_{ij}^{(n)}\mathbf{x}_j + \mathbf{x}_j'\mathbf{v}_{ij}^{(n)}$, $K_{ij}^{(n)}$ is a constant and $\mathbf{Q}_{ij}^{(n)} : d_j \times d_j$ is symmetric for all nodes i and all $j \in \mathcal{N}_i$. We have that

$$\begin{aligned}
\log(\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)) &= -\frac{1}{2}\mathbf{x}_i'\mathbf{S}_{ii}\mathbf{x}_i + \mathbf{x}_i'\boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\backslash j} K_{ti}^{(n)} - \frac{1}{2}\sum_{t\in\mathcal{N}_i\backslash j}\mathbf{x}_i'\mathbf{Q}_{ti}^{(n)}\mathbf{x}_i \\
&\quad + \sum_{t\in\mathcal{N}_i\backslash j}\mathbf{x}_i'\mathbf{v}_{ti}^{(n)} \\
&= -\frac{1}{2}\mathbf{x}_i'\left[\mathbf{S}_{ii} + \sum_{t\in\mathcal{N}_i\backslash j}\mathbf{Q}_{ti}^{(n)}\right]\mathbf{x}_i + \mathbf{x}_i'\left[\boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\backslash j}\mathbf{v}_{ti}^{(n)}\right] + \tilde{K}_{ij}^{(n)} \\
&= -\frac{1}{2}\mathbf{x}_i'\mathbf{P}_{ij}^{(n)}\mathbf{x}_i + \mathbf{x}_i'\mathbf{z}_{ij}^{(n)} + \tilde{K}_{ij}^{(n)}, \qquad (1.13)
\end{aligned}$$

where $\mathbf{P}_{ij}^{(n)} = \mathbf{S}_{ii} + \sum_{t\in\mathcal{N}_i\backslash j}\mathbf{Q}_{ti}^{(n)}$, $\mathbf{z}_{ij}^{(n)} = \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\backslash j}\mathbf{v}_{ti}^{(n)}$ and $\tilde{K}_{ij}^{(n)} = \sum_{t\in\mathcal{N}_i\backslash j} K_{ti}^{(n)}$. We make the additional assumption that $\mathbf{P}_{ij}^{(n)}$ is positive definite and note that $\mathbf{P}_{ij}^{(n)}$ is symmetric by the symmetry of the $\mathbf{Q}_{ti}^{(n)}$'s. We see that:

$$\begin{aligned}
m_{ij}^{(n+1)}(\mathbf{x}_j) &= \tilde{K}_{ij}^{(n)} + \max_{\mathbf{x}_i}\left\{ -\frac{1}{2}\mathbf{x}_i'\mathbf{P}_{ij}^{(n)}\mathbf{x}_i + \mathbf{x}_i'(\mathbf{z}_{ij}^{(n)} - \mathbf{S}_{ij}\mathbf{x}_j)\right\} \\
&= \tilde{K}_{ij}^{(n)} + \frac{1}{2}(\mathbf{S}_{ij}\mathbf{x}_j - \mathbf{z}_{ij}^{(n)})'[\mathbf{P}_{ij}^{(n)}]^{-1}(\mathbf{S}_{ij}\mathbf{x}_j - \mathbf{z}_{ij}^{(n)}) \\
&= \tilde{K}_{ij}^{(n)} + \frac{1}{2}[\mathbf{z}_{ij}^{(n)}]'[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{z}_{ij}^{(n)} + \frac{1}{2}\mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{S}_{ij}\mathbf{x}_j \\
&\quad - \mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{z}_{ij}^{(n)} \\
&= K_{ij}^{(n+1)} - \frac{1}{2}\mathbf{x}_j'\mathbf{Q}_{ij}^{(n+1)}\mathbf{x}_j + \mathbf{x}_j'\mathbf{v}_{ij}^{(n+1)}, \qquad (1.14)
\end{aligned}$$

where:

$$\begin{aligned}
K_{ij}^{(n+1)} &= \tilde{K}_{ij}^{(n)} + \frac{1}{2}[\mathbf{z}_{ij}^{(n)}]'[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{z}_{ij}^{(n)} \\
\mathbf{Q}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{S}_{ij} \qquad (1.15) \\
\mathbf{v}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}]^{-1}\mathbf{z}_{ij}^{(n)}. \qquad (1.16)
\end{aligned}$$

We see that conjugate messages can be obtained through appropriate initialisation and we use $\mathbf{Q}_{ij}^{(0)} = \mathbf{0}$ and $\mathbf{v}_{ij}^{(0)} = \mathbf{0}$ for all nodes i and all $j \in \mathcal{N}_i$.

1.3.2 Processing the Posteriors

After performing one pass of synchronous message updates we need to construct marginals associated with the updated messages. At iteration n we have:

$$\begin{aligned}
\text{post}_i^{(n)}(\mathbf{x}_i) &= \psi_{ii}(\mathbf{x}_i) \prod_{t \in \mathcal{N}_i} \exp\left\{ m_{ti}^{(n)}(\mathbf{x}_i) \right\} \\
&= \exp\left[-\frac{1}{2}\mathbf{x}_i' \mathbf{S}_{ii} \mathbf{x}_i + \mathbf{x}_i' \boldsymbol{b}_i \right] \prod_{t \in \mathcal{N}_i} \exp\left[K_{ti}^{(n)} - \frac{1}{2}\mathbf{x}_i' \mathbf{Q}_{ti}^{(n)} \mathbf{x}_i + \mathbf{x}_i' \mathbf{v}_{ti}^{(n)} \right] \\
&= \exp\left[\sum_{t \in \mathcal{N}_i} K_{ti}^{(n)} \right] \exp\left[-\frac{1}{2}\mathbf{x}_i' \mathbf{P}_i^{(n)} \mathbf{x}_i + \mathbf{x}_i' \mathbf{z}_i^{(n)} \right] \\
&\propto \exp\left[-\frac{1}{2}\mathbf{x}_i' \mathbf{P}_i^{(n)} \mathbf{x}_i + \mathbf{x}_i' \mathbf{z}_i^{(n)} \right],
\end{aligned}$$

where $\mathbf{P}_i^{(n)} = \mathbf{S}_{ii} + \sum_{t \in \mathcal{N}_i} \mathbf{Q}_{ti}^{(n)}$ and $\mathbf{z}_i^{(n)} = \boldsymbol{b}_i + \sum_{t \in \mathcal{N}_i} \mathbf{v}_{ti}^{(n)}$. Notice that the constant $K_{ij}^{(n)}$ plays no role in determining the marginal after the posterior distribution has been normalised. Hence the update and tracking of the constant component of the messages is ignored in the GaBP algorithm. Note that the posterior distribution estimates the marginal as a Gaussian distribution, with precision matrix and potential vector $\mathbf{P}_i^{(n)}$ and $\mathbf{z}_i^{(n)}$ respectively. This gives an estimate for the marginal mean, i.e. $\boldsymbol{\mu}_i^{(n)} = [\mathbf{P}_i^{(n)}]^{-1}\mathbf{z}_i^{(n)}$, and therefore we can use the GaBP algorithm to convert a multivariate Gaussian distribution in canonical form to approximate marginals in the mean-covariance parameterisation. This is an important observation, since it links GaBP to linear algebra. We introduce some terminology:

Terminology 1 *The posterior distribution for node $i \in \mathcal{V}$ resulting from the GaBP algorithm at iteration n is characterised by the posterior precision, $\mathbf{P}_i^{(n)}$, and the posterior mean, $\boldsymbol{\mu}_i^{(n)}$.*

The synchronous GaBP procedure in Algorithm 1 uses the following computational shortcuts:

$$\begin{aligned}
\mathbf{P}_{ij}^{(n)} &= \mathbf{P}_i^{(n)} - \mathbf{Q}_{ji}^{(n)} \\
\mathbf{z}_{ij}^{(n)} &= \mathbf{z}_i^{(n)} - \mathbf{v}_{ji}^{(n)}.
\end{aligned}$$

Algorithm 1 is terminated when $\text{Err} \leq \epsilon$, where $\boldsymbol{e}_i^{(n)} = \sum_j \mathbf{S}_{ij}\boldsymbol{\mu}_j^{(n)} - \boldsymbol{b}_i$ and $\text{Err} = \max_i\{||\boldsymbol{e}_i^{(n)}||_\infty\}$, or if the maximum number of iterations is reached. The

Algorithm 1 Synchronous GaBP

1. Provide a precision matrix $\mathbf{S} : k \times k$, a potential vector $\boldsymbol{b} : k \times 1$ and clusters $\mathcal{C}_i : i = 1, 2, \ldots, p$ as inputs.
2. Specify a tolerance ϵ and a maximum number of iterations m.
3. Initialise $\mathbf{Q}_{ij}^{(0)} = \mathbf{0} : d_j \times d_j$ and $\mathbf{v}_{ij}^{(0)} = \mathbf{0} : d_j \times 1$ for all i and all $j \in \mathcal{N}_i$.
4. Set Err = Inf and $n = 0$.
5. While Err $> \epsilon$
   a) Compute $\mathbf{P}_i^{(n)} = \mathbf{S}_{ii} + \sum_{j \in \mathcal{N}_i} \mathbf{Q}_{ji}^{(n)}$ and $\mathbf{z}_i^{(n)} = \boldsymbol{b}_i + \sum_{j \in \mathcal{N}_i} \mathbf{v}_{ji}^{(n)}$ for $i = 1, 2, \ldots, p$.
   b) Set $\boldsymbol{\mu}_i^{(n)} = [\mathbf{P}_i^{(n)}]^{-1}\mathbf{z}_i^{(n)}$, $\boldsymbol{e}_i^{(n)} = \sum_j \mathbf{S}_{ij}\boldsymbol{\mu}_j^{(n)} - \boldsymbol{b}_i$ and Err $= \max_i\{||\boldsymbol{e}_i^{(n)}||_\infty\}$.
   c) If Err $> \epsilon$, do for all $i \in \{1, 2, \ldots, p\}$ and all $j \in \mathcal{N}_i$:
   $\mathbf{Q}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)} - \mathbf{Q}_{ji}^{(n)}]^{-1}\mathbf{S}_{ij}$ and
   $\mathbf{v}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)} - \mathbf{Q}_{ji}^{(n)}]^{-1}[\mathbf{z}_i^{(n)} - \mathbf{v}_{ji}^{(n)}]$.
   d) Increment n.
   e) If $n = m$, break.
6. End.

reason for this is that convergence of the posterior means usually indicates that the remaining components of the algorithm have converged, since the posterior means are functions of all the components iterated in Algorithm 1. In the next section we discuss the convergence and inference quality of the GaBP algorithm.

1.3.3 Convergence Behaviour and Inference Quality

Important work on GaBP can be found in Weiss and Freeman (2001). Here it is shown that, if GaBP converges, the marginal means supplied by this algorithm are the correct ones. Furthermore, it is shown that, if the precision matrix is strictly diagonally dominant, GaBP is guaranteed to converge.

Definition 1 *A matrix* $\mathbf{S} = [s_{ij}]$ *is defined to be strictly diagonally dominant (sdd) if* $s_{ii} > \sum_{j \neq i} |s_{ji}|$ *for all* i.

Definition 2 *A matrix* $\mathbf{S} = [s_{ij}]$ *is defined to be weakly diagonally dominant (wdd) if* $s_{ii} \geq \sum_{j \neq i} |s_{ji}|$ *for all* i.

Weiss and Freeman (2001) also interpret GaBP on loopy graphs as BP on a tree-structured Gaussian MG (known as unwrapped trees). This provides a more convenient typology for the convergence analysis of GaBP, something that will be revisited in this dissertation.

Unfortunately, GaBP does not necessarily provide the correct marginal precisions at convergence, but can still provide reasonable estimates for these quantities.

The exact convergence properties of GaBP remains an open area of research. The class of precision matrices for which GaBP converges has been expanded to include precision matrices that are walk-summable (Malioutov *et al.*, 2006). We introduce some new notation and definitions.

Notation 3 *Consider a matrix* $\mathbf{A} : m \times n$. *By* $|\mathbf{A}|$ *we mean the matrix* $\mathbf{A}$ *in which all elements are replaced by their corresponding absolute values. The determinant of a square matrix* $\mathbf{A}$ *is denoted by* $det(\mathbf{A})$.

Definition 3 *Consider a matrix* $\mathbf{A} : m \times m$ *with eigenvalues (possibly complex)* $\lambda_1, \lambda_2, \ldots, \lambda_m$. *Let* $|.|$ *be the modulus of a complex number. The spectral radius of* $\mathbf{A}$ *is defined as* $max_i\{|\lambda_i|\}$ *and is denoted by* $\rho(\mathbf{A})$.

Definition 4 *The zero-diagonal spectral radius of a matrix* $\mathbf{A} : m \times m = [a_{ij}]$ *is defined to be the spectral radius of* $diag(a_{11}, a_{22}, \ldots, a_{mm}) - \mathbf{A}$, *and is denoted by* $\tilde{\rho}(\mathbf{A})$.

Definition 5 *Consider a positive definite matrix* $\mathbf{S} : p \times p = [s_{ij}]$ *and let* $\mathbf{D} = diag(\frac{1}{\sqrt{s_{11}}}, \frac{1}{\sqrt{s_{22}}}, \ldots, \frac{1}{\sqrt{s_{pp}}})$. *The matrix* $\mathbf{S}$ *is defined to be walk-summable if* $\tilde{\rho}(|\mathbf{DSD}|) < 1$.

The proof for convergence of the GaBP algorithm for walk-summable precision matrices provided by Malioutov *et al.* (2006) is for univariate nodes. In Chapter 3 we show that the walk-summability convergence result also extends to the multivariate case. We will also define a new, preconditioned variant of walk-summability that shows that GaBP with higher-dimensional nodes could converge in cases where the univariate version does not. A recent work provides necessary and sufficient conditions for the convergence of univariate synchronous GaBP, under a specified initialisation set (Su and Wu, 2015). Furthermore, necessary and sufficient convergence conditions are established for damped univariate synchronous GaBP and these include the allowable range for the damping factor. They also provide theoretical confirmation that damping can improve the convergence behaviour of GaBP. The problem with these contributions is that their conditions for convergence may be difficult to verify. For instance, verifying walk-summability requires finding the maximum absolute eigenvalue of a $p \times p$ matrix, while the conditions from Su and

Wu (2015) require solving a semi-definite program (SDP) and evaluating the spectral radius of an infinite dimensional matrix (Sui *et al.*, 2015).

1.3.4 Links to Linear Algebra

GaBP can be connected to the area of linear algebra (Shental *et al.*, 2008). As discussed previously, GaBP can be used to convert a multivariate Gaussian distribution in canonical form to approximate marginals for each set of variables $\mathcal{C}_i : i = 1, 2, \ldots, p$. Suppose that GaBP has converged, then the converged posterior marginals provide the exact marginal means, i.e. $\boldsymbol{\mu}_i^{(\infty)} = \boldsymbol{\mu}_i$, where $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$, and hence GaBP implicitly solves this linear system of equations. The use of GaBP as a solver of linear systems has been proposed in the literature (Shental *et al.*, 2008; Bickson, 2008; El-Kurdi *et al.*, 2012*b*). GaBP is attractive as a solver of linear systems due to the ease of distributed implementation and the exploitation of sparsity in the precision matrix (our high-level approach retains these characteristics).

One of the major restrictions in using GaBP as a solver of linear systems is that it can only be applied on a sub-class of symmetric and positive definite matrices. In this dissertation, we will show that our high-level approach yields a GaBP variant capable of converging for any symmetric and positive definite matrix, given sufficient regularisation. Hence, our approach expands the application domain of GaBP in the context of linear systems.

GaBP has been compared favourably to other solvers of linear systems in the literature, such as the Jacobi and Gauss-Seidel methods (Bickson, 2008). In our experience, the GaBP algorithm can struggle to compete with methods such as the conjugate gradient (CG) and preconditioned conjugate gradient (PCG) methods, particularly in loopy MGs with ill-conditioned precision matrices. Apart from inducing convergence, our high-level approach can also accelerate GaBP in cases where this algorithm converges without the aid of regularisation. A portion of this dissertation is dedicated to comparing our GaBP variant with the CG and PCG methods for the purpose of solving linear systems. In Chapter 3 we show that the basic GaBP algorithm performs a certain type of preconditioning automatically. The fact that this preconditioning can significantly lower the condition number of the precision matrix is an advantage of GaBP over CG and PCG.

From a marginalisation viewpoint, GaBP can be used to estimate the marginal precision associated with each set of variables assigned to nodes. The direct method of determining the marginal precisions is to invert the precision matrix, extract the appropriate blocks and invert these block matrices. Therefore, GaBP can be used for the purpose of estimating certain diagonal blocks of the inverse of symmetric and positive definite matrices without the use of direct

matrix inversion. Within the context of our high-level approach, we show, empirically, that its application in GaBP can often improve the accuracy of these approximations.

1.3.5 Competing Algorithms

This section is dedicated to describing some of the competing algorithms to our approach.

- *Convergence Fix GaBP - CFGaBP* (Johnson *et al.*, 2009). This method uses basic GaBP to solve a sequence of linear systems defined by $(\mathbf{S} + \lambda\mathbf{I})\boldsymbol{\mu}^{(n)} = \boldsymbol{b} + \lambda\boldsymbol{\mu}^{(n-1)}$. If λ is chosen sufficiently large (for instance, large enough for $\mathbf{S} + \lambda\mathbf{I}$ to be walk-summable), each application of basic GaBP will converge and the limit of $\boldsymbol{\mu}^{(n)}$ will solve the system of equations $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$.

  One problem with CFGaBP is that convergence can be slow due to the double-loop nature of the algorithm (inner loop applies GaBP and outer loop updates the linear system). Also, since CFGaBP was designed specifically to solve linear systems, it does not address the problem of improving the precision estimates.

- *Reweighted GaBP - ReGaBP* (Ruozzi and Tatikonda, 2013). This approach uses a parameterised generalisation of the max-sum algorithm to improve on the convergence of GaBP. For ReGaBP there is always a set of parameters that gives computation trees that are positive definite – this is sufficient for the precision components of messages to converge.

  The main problem with ReGaBP is that it may require additional damping factors for the mean components to converge. As far as we are aware, there are no theoretical results establishing that there will always be sufficient damping to guarantee convergence (although we were always able to find sufficient damping in our simulations). In our opinion, the convergence behaviour of ReGaBP can be erratic in terms of the selection of the reweighting, and this is further complicated by the need for additional damping. The effect of the reweighting of the messages on the accuracy of the posterior precisions as approximations for the marginal precisions is a concern.

- *Feedback GaBP - FGaBP* (Liu, 2010). A set of nodes is called a feedback set if their removal results in a cycle-free graph. The individual nodes in such a set are called feedback nodes. The idea behind a feedback message passing algorithm (FMP) is to treat the feedback and non-feedback nodes separately. Informally speaking, a FMP starts with message passing among the non-feedback nodes. The non-feedback nodes

send messages to the feedback nodes, which they use to compute their exact marginals. The next step is for the feedback nodes to pass messages, based on their exact marginals, back to the non-feedback nodes. The non-feedback nodes use these messages to perform a final round of message-passing, yielding their exact marginals.

For the Gaussian case, the inference provided by a FMP is exact and completes after $\mathcal{O}(k^2p)$ computations, with k being the number of feedback nodes and p the total number of nodes. The consequence is that we would like the feedback set to be as small as possible.

The main problem with FGaBP is that it is less applicable in graphs with a low degree of sparsity. When a graph has a low degree of sparsity, the size of the smallest feedback set will be large, resulting in a higher number of computations. Furthermore, even in cases of moderate sparsity, finding the smallest feedback set can be computationally intractable. One way of dealing with these issues is to use a pseudo-feedback set, that is a set of nodes the removal of which removes the most significant cycles in the graph. The advantage is that the pseudo-feedback set can be chosen to be of any size. The disadvantages are that inference becomes approximate and one is still left with the task of selecting a good pseudo-feedback set.

- *Relaxed GaBP - RGaBP* (El-Kurdi *et al.*, 2012*b*). RGaBP uses relaxation factors (algorithm can easily be adjusted to use damping factors) to accelerate the convergence of basic GaBP (Su and Wu, 2015).

  The main problem with RGaBP is that it uses the same updates as basic GaBP for the precision components of messages. The consequence is that RGaBP cannot be applied to arbitrary precision matrices.

- Johnson *et al.* (2009) make a brief reference to a compressed inner-loop version of CFGaBP, in which each application of GaBP is limited to one iteration. They report that compressed inner-loop CFGaBP can be more efficient than the original method, but may require damping for the adjustment of the potential vector.

  The similarity of the compressed CFGaBP to our algorithm depends heavily on the interpretation of the description in the literature. We could not find any reference to compressed CFGaBP in the source code provided by Bickson (2008). If we consider the source code provided for the original CFGaBP method, and apply the description provided by Johnson *et al.* (2009), we see differences in the application of the two methods.

In contrast to the method provided in this dissertation, compressed inner-loop CFGaBP is not a full marginalisation algorithm, requires additional damping factors to be specified, is not theoretically proved to converge for all precision matrices, is not generalised to higher-dimensional nodes, and does not provide a principle that can be applied in general BP.

Each algorithm mentioned above suffers from at least one of the following problems:

1. Does not provide a principle that can be generalised to other applications of BP.
2. Does not converge for arbitrary precision matrices.
3. Convergence can be slow due to implementation method.
4. Estimation of the precisions is ignored.
5. No theoretical proof of guaranteed convergence.
6. Not applicable in dense graphs.
7. Focus is on one-dimensional nodes only.

A large portion of this dissertation will be dedicated to show how our algorithm deals with these issues. In the empirical sections of Chapter 2 and Chapter 4, we provide a comparison of our variant with some of these algorithms. The next section describes the application of our high-level approach in GaBP.

1.4 High-level Application in GaBP

We refer to the application of our high-level approach in GaBP as "regularised" - or "slow" - Gaussian belief propagation. The "slow" relates the regularisation used in our approach to the principle of slow learning. We label our method sGaBP as opposed to rGaBP to avoid confusion with relaxed Gaussian belief propagation (RGaBP).

1.4.1 Message Updates

A natural selection for the L_q-norm in Equation (1.11) is $q = 2$, since this will preserve the conjugacy. As for GaBP, we assume that $m_{ij}^{(n)}(\mathbf{x}_j) = K_{ij}^{(n)} - \frac{1}{2}\mathbf{x}_j'\mathbf{Q}_{ij}^{(n)}\mathbf{x}_j + \mathbf{x}_j'\mathbf{v}_{ij}^{(n)}$, where the components will depend on λ. We will not express the dependence on λ explicitly. Again, $K_{ij}^{(n)}$ is a constant and $\mathbf{Q}_{ij}^{(n)}$ is symmetric. In an identical manner to Equation (1.13), we obtain:

$$\log(\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)) = -\frac{1}{2}\mathbf{x}_i'\mathbf{P}_{ij}^{(n)}\mathbf{x}_i + \mathbf{x}_i'\mathbf{z}_{ij}^{(n)} + \tilde{K}_{ij}^{(n)}. \tag{1.17}$$

The application of Equation (1.11) yields:

$$\begin{aligned}
m_{ij}^{(n+1)}(\mathbf{x}_j) &= \tilde{K}_{ij}^{(n)} + \max_{\mathbf{x}_i}\Big\{ -\frac{1}{2}\mathbf{x}_i'\mathbf{P}_{ij}^{(n)}\mathbf{x}_i + \mathbf{x}_i'(\mathbf{z}_{ij}^{(n)} - \mathbf{S}_{ij}\mathbf{x}_j) \\
&\quad - \frac{\lambda}{2}(\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)})'(\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)})\Big\} \\
&= \tilde{K}_{ij}^{(n)} - \frac{\lambda}{2}[\boldsymbol{\theta}_i^{(n-1)}]'\boldsymbol{\theta}_i^{(n-1)} \\
&\quad + \max_{\mathbf{x}_i}\Big\{ -\frac{1}{2}\mathbf{x}_i'(\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i})\mathbf{x}_i + \mathbf{x}_i'(\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)} - \mathbf{S}_{ij}\mathbf{x}_j)\Big\} \\
&= \tilde{\tilde{K}}_{ij}^{(n)} \\
&\quad + \frac{1}{2}(\mathbf{S}_{ij}\mathbf{x}_j - \mathbf{z}_{ij}^{(n)} - \lambda\boldsymbol{\theta}_i^{(n-1)})'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}(\mathbf{S}_{ij}\mathbf{x}_j - \mathbf{z}_{ij}^{(n)} - \lambda\boldsymbol{\theta}_i^{(n-1)}) \\
&= \tilde{\tilde{K}}_{ij}^{(n)} + \frac{1}{2}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}]'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}] \\
&\quad + \frac{1}{2}\mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}\mathbf{S}_{ij}\mathbf{x}_j - \mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}] \\
&= K_{ij}^{(n+1)} - \frac{1}{2}\mathbf{x}_j'\mathbf{Q}_{ij}^{(n+1)}\mathbf{x}_j + \mathbf{x}_j'\mathbf{v}_{ij}^{(n+1)}, \qquad (1.18)
\end{aligned}$$

where:

$$\begin{aligned}
\tilde{\tilde{K}}_{ij}^{(n)} &= \tilde{K}_{ij}^{(n)} - \frac{\lambda}{2}[\boldsymbol{\theta}_i^{(n-1)}]'\boldsymbol{\theta}_i^{(n-1)} \\
K_{ij}^{(n+1)} &= \tilde{\tilde{K}}_{ij}^{(n)} + \frac{1}{2}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}]'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}] \\
\mathbf{Q}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}\mathbf{S}_{ij} \\
\mathbf{v}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}].
\end{aligned}$$

We still have not discussed the selection of the value $\boldsymbol{\theta}_i^{(n-1)}$. Suppose that the posterior distribution associated with node i at iteration $n-1$ is given by $\text{post}_i^{(n-1)}(\mathbf{x}_i)$, then we propose using:

$$\boldsymbol{\theta}_i^{(n-1)} = \underset{\mathbf{x}_i}{\text{argmax}}\{\text{post}_i^{(n-1)}(\mathbf{x}_i)\}. \qquad (1.19)$$

In the next section we will see that $\text{post}_i^{(n-1)}(\mathbf{x}_i)$ is proportional to a Gaussian distribution and hence,

$$\boldsymbol{\theta}_i^{(n-1)} = \boldsymbol{\mu}_i^{(n-1)}. \qquad (1.20)$$

1.4.2 Processing the Posteriors

Normal GaBP gives the correct marginal means under the assumption of convergence. In order to preserve this property for sGaBP, we need to treat the

posterior distributions in a slightly different way than for GaBP. We form the posterior distribution at iteration n as:

$$\text{post}_i^{(n)}(\mathbf{x}_i) \propto \exp\left[-\frac{1}{2}\mathbf{x}_i'\mathbf{P}_i^{(n)}\mathbf{x}_i + \mathbf{x}_i'\mathbf{P}_i^{(n)}[\lambda\mathbf{I}_{d_i} + \mathbf{P}_i^{(n)}]^{-1}[\mathbf{z}_i^{(n)} + \lambda\boldsymbol{\mu}_i^{(n-1)}]\right] \quad (1.21)$$

where $\mathbf{P}_i^{(n)} = \mathbf{S}_{ii} + \sum_{t\in\mathcal{N}_i}\mathbf{Q}_{ti}^{(n)}$ and $\mathbf{z}_i^{(n)} = \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i}\mathbf{v}_{ti}^{(n)}$. The precision associated with Equation (1.21) is similar to the precision associated with GaBP, however $\mathbf{Q}_{ti}^{(n)}$ now depends on the regularisation parameter. For clarity we note that $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ remains

$$\psi_{ii}(\mathbf{x}_i)\prod_{t\in\mathcal{N}_i\backslash j}\exp\left\{m_{ti}^{(n)}(\mathbf{x}_i)\right\}$$

as in Equation (1.9), but now depends on λ. The marginal mean associated with Equation (1.21) can be shown to be:

$$\boldsymbol{\mu}_i^{(n)} = [\lambda\mathbf{I}_{d_i} + \mathbf{P}_i^{(n)}]^{-1}[\mathbf{z}_i^{(n)} + \lambda\boldsymbol{\mu}_i^{(n-1)}]. \quad (1.22)$$

Essentially we are introducing a form of damping on the progression of the posterior means over the iterations. This damping is necessary to ensure that the posterior means provided at convergence give the correct marginal means (see Theorems 3 and 14).

We implement sGaBP in Algorithm 2 (see next page), where we use the notation:

$$\mathbf{P}_i^{(n)}(\lambda) = \lambda\mathbf{I}_{d_i} + \mathbf{P}_i^{(n)}.$$

1.4.3 Sum-product sGaBP

In this section we discuss a sum-product (as opposed to max-sum) formulation of sGaBP. The sum-product message updates in the unregularised setting was given as,

$$m_{ij}^{(n+1)}(\mathbf{x}_j) = \int \psi_{ii}(\mathbf{x}_i)\psi_{ij}(\mathbf{x}_i,\mathbf{x}_j)\prod_{t\in\mathcal{N}_i\backslash j} m_{ti}^{(n)}(\mathbf{x}_i)d\mathbf{x}_i,$$

in Equation (1.5). Within the context of regularisation, we define the message-updates to be:

$$m_{ij}^{(n+1)}(\mathbf{x}_j) = \int \psi_{ii}(\mathbf{x}_i)\psi_{ij}(\mathbf{x}_i,\mathbf{x}_j)\exp\left(-\frac{\lambda}{q}||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_q^q\right)\prod_{t\in\mathcal{N}_i\backslash j} m_{ti}^{(n)}(\mathbf{x}_i)d\mathbf{x}_i, \quad (1.23)$$

Algorithm 2 Synchronous regularised GaBP

1. Provide a precision matrix $\mathbf{S} : k \times k$, a potential vector $\boldsymbol{b} : k \times 1$ and clusters $\mathcal{C}_i : i = 1, 2, \ldots, p$ as inputs.
2. Specify a tolerance ϵ, a maximum number of iterations m and a regularisation parameter λ.
3. Initialise $\mathbf{Q}_{ij}^{(0)} = \mathbf{0} : d_j \times d_j$, $\mathbf{v}_{ij}^{(0)} = \mathbf{0} : d_j \times 1$, $\boldsymbol{\mu}_i^{(-1)} = \boldsymbol{b}_i$ for all i and all $j \in \mathcal{N}_i$.
4. Set Err = Inf and $n = 0$.
5. While Err $> \epsilon$
   a) Compute $\mathbf{P}_i^{(n)}(\lambda) = \lambda \mathbf{I}_{d_i} + \mathbf{S}_{ii} + \sum_{j \in \mathcal{N}_i} \mathbf{Q}_{ji}^{(n)}$ and $\mathbf{z}_i^{(n)} = \boldsymbol{b}_i + \sum_{j \in \mathcal{N}_i} \mathbf{v}_{ji}^{(n)}$ for $i = 1, 2, \ldots, p$.
   b) Set $\boldsymbol{\mu}_i^{(n)} = [\mathbf{P}_i^{(n)}(\lambda)]^{-1}[\lambda \boldsymbol{\mu}_i^{(n-1)} + \mathbf{z}_i^{(n)}]$, $\boldsymbol{e}_i^{(n)} = \sum_j \mathbf{S}_{ij} \boldsymbol{\mu}_j^{(n)} - \boldsymbol{b}_i$ and Err $= \max_i \{||\boldsymbol{e}_i^{(n)}||_\infty\}$.
   c) If Err $> \epsilon$, do for all $i \in \{1, 2, \ldots, p\}$ and all $j \in \mathcal{N}_i$: $\mathbf{Q}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)}(\lambda) - \mathbf{Q}_{ji}^{(n)}]^{-1}\mathbf{S}_{ij}$ and $\mathbf{v}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)}(\lambda) - \mathbf{Q}_{ji}^{(n)}]^{-1}[\lambda \boldsymbol{\mu}_i^{(n-1)} + \mathbf{z}_i^{(n)} - \mathbf{v}_{ji}^{(n)}]$.
   d) Increment n.
   e) If $n = m$, break.
6. End.

for all $i \in \mathcal{V}$ and all $j \in \mathcal{N}_i$. In the remainder of this section we will show that in the Gaussian case, with $q = 2$, max-sum sGaBP and sum-product sGaBP are equivalent, assuming that $\boldsymbol{\theta}_i^{(n-1)} = \boldsymbol{\mu}_i^{(n-1)}$ and that the posterior distributions (for both variants) are processed as discussed in Section 1.4.2.

Consider a Gaussian random vector $\mathbf{Y} : m \times 1$ with precision matrix $\mathbf{S}_Y$ and potential vector $\boldsymbol{b}_Y$. The density function of $\mathbf{Y}$ can be written as:

$$f_Y(\mathbf{y}) = (2\pi)^{-\frac{m}{2}} (\det(\mathbf{S}_Y))^{\frac{1}{2}} \exp\left(-\frac{1}{2}\mathbf{y}'\mathbf{S}_Y\mathbf{y} + \mathbf{y}'\boldsymbol{b}_Y - \frac{1}{2}\boldsymbol{b}_Y'\mathbf{S}_Y^{-1}\boldsymbol{b}_Y \right). \quad (1.24)$$

A consequence of Equation (1.24) is that:

$$\int_{\mathbf{y}} \exp\left(-\frac{1}{2}\mathbf{y}'\mathbf{S}_Y\mathbf{y} + \mathbf{y}'\boldsymbol{b}_Y \right) d\mathbf{y} = (2\pi)^{\frac{m}{2}} (\det(\mathbf{S}_Y))^{-\frac{1}{2}} \exp\left(\frac{1}{2}\boldsymbol{b}_Y'\mathbf{S}_Y^{-1}\boldsymbol{b}_Y \right). \quad (1.25)$$

Consider sum-product sGaBP where we assume that,

$$m_{ti}^{(n)}(\mathbf{x}_i) \propto \exp\Big(-\frac{1}{2}\mathbf{x}_i'\mathbf{Q}_{ti}^{(n)}\mathbf{x}_i + \mathbf{x}_i'\mathbf{v}_{ti}^{(n)}\Big), \tag{1.26}$$

for all $t \in \mathcal{V}$ and all $i \in \mathcal{N}_t$. Writing $\text{post}_{i\setminus j}(\mathbf{x}_i) = \psi_{ii}(\mathbf{x}_i)\prod_{t\in\mathcal{N}_i\setminus j} m_{ti}^{(n)}(\mathbf{x}_i) = \exp\Big(-\frac{1}{2}\mathbf{x}_i'\mathbf{S}_{ii}\mathbf{x}_i + \mathbf{x}_i'\boldsymbol{b}_i\Big)\prod_{t\in\mathcal{N}_i\setminus j} m_{ti}^{(n)}(\mathbf{x}_i)$, we see that

$$\text{post}_{i\setminus j}(\mathbf{x}_i) \propto \exp\Big(-\frac{1}{2}\mathbf{x}_i'\mathbf{P}_{ij}^{(n)}\mathbf{x}_i + \mathbf{x}_i'\mathbf{z}_{ij}^{(n)}\Big), \tag{1.27}$$

where $\mathbf{P}_{ij}^{(n)}$ and $\mathbf{z}_{ij}^{(n)}$ are defined as in Section 1.3.1. The following holds:

$$\begin{aligned}
&\psi_{ii}(\mathbf{x}_i)\psi_{ij}(\mathbf{x}_i,\mathbf{x}_j)\exp\Big(-\frac{\lambda}{2}||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_2^2\Big)\prod_{t\in\mathcal{N}_i\setminus j} m_{ti}^{(n)}(\mathbf{x}_i)\\
&= \text{post}_{i\setminus j}(\mathbf{x}_i)\exp\Big(-\mathbf{x}_i'\mathbf{S}_{ij}\mathbf{x}_j\Big)\exp\Big(-\frac{\lambda}{2}[\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}]'[\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}]\Big)\\
&\propto \exp\Big(-\frac{1}{2}\mathbf{x}_i'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]\mathbf{x}_i + \mathbf{x}_i'[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)} - \mathbf{S}_{ij}\mathbf{x}_j]\Big).
\end{aligned} \tag{1.28}$$

The message from a node i to a neighbouring node j can be computed using:

$$\begin{aligned}
m_{ij}^{(n+1)}(\mathbf{x}_j) &\propto \int_{\mathbf{x}_i} \exp\Big(-\frac{1}{2}\mathbf{x}_i'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]\mathbf{x}_i + \mathbf{x}_i'[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)} - \mathbf{S}_{ij}\mathbf{x}_j]\Big)d\mathbf{x}_i\\
&\propto \exp\Big(\frac{1}{2}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)} - \mathbf{S}_{ij}\mathbf{x}_j]'[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)} - \mathbf{S}_{ij}\mathbf{x}_j],\Big)
\end{aligned} \tag{1.29}$$

by Equation (1.25). Further simplification of (1.29) yields,

$$\begin{aligned}
m_{ij}^{(n+1)}(\mathbf{x}_j) \propto \exp\Big(&\frac{1}{2}\mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}\mathbf{S}_{ij}\mathbf{x}_j\\
&- \mathbf{x}_j'\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}]\Big).
\end{aligned} \tag{1.30}$$

We see that,

$$\begin{aligned}
\mathbf{Q}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}\mathbf{S}_{ij}\\
\mathbf{v}_{ij}^{(n+1)} &= -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)} + \lambda\mathbf{I}_{d_i}]^{-1}[\mathbf{z}_{ij}^{(n)} + \lambda\boldsymbol{\theta}_i^{(n-1)}],
\end{aligned} \tag{1.31}$$

which give the same updates as derived in Section 1.4.1.

1.5 Overview of this Dissertation

We give a brief overview of the material covered in each chapter:

- **Chapter 2**. We consider the behaviour of sGaBP in the one-dimensional node setting. We prove convergence of the sGaBP algorithm in this context and show that the posterior means are the exact marginal means. We show empirically that sGaBP compares favourably to certain other GaBP variants, both in terms of convergence speed and inference quality. We compare sGaBP to the CG algorithm in cases where basic GaBP does not converge. Essentially, we are covering the work done by Kamper *et al.* (2018*a*).

- **Chapter 3**. We give an unwrapped tree analysis of GaBP on a graph where nodes can be of arbitrary dimension (Kamper *et al.*, 2018*b*). This UWT analysis is used to derive a preconditioned walk-summability condition as a sufficient condition for the convergence of GaBP. This condition moves beyond ordinary walk-summability and shows that multivariate GaBP may converge in cases where univariate GaBP does not. We also describe a certain automatic preconditioning done by GaBP. This type of preconditioning can have a significant effect on the performance of the CG method.

- **Chapter 4**. This chapter continues a generalisation of the results from Chapter 2 to higher-dimensional nodes (Kamper *et al.*, 2018*c*). Some of the results from Chapter 3 are used to derive the asymptotic behaviour of $\mathbf{Q}_{ij}^{(n)}$ as $\lambda \to \infty$. These asymptotic expressions are used to prove convergence. We extend some of the GaBP variants in the literature to allow for implementation in higher-dimensional settings. We observe empirically that sGaBP compares favourably to these algorithms in terms of convergence speed and inference quality. We also investigate a heuristic for the selection of λ.

- **Chapter 5**. We consider the implications of the work done in this dissertation for future research.

Chapter 2

Univariate Regularised Gaussian Belief Propagation

In this chapter we investigate sGaBP where the nodes are one-dimensional. In our experience, most of the literature is focused on the univariate case and this provides a good starting point for comparing sGaBP to other variants of GaBP. From this point on we refer to our high-level approach as the principle of node regularisation. For the purpose of this chapter, the problem is marginalising a multivariate Gaussian with precision matrix $\mathbf{S} : p \times p$ and potential vector $\boldsymbol{b}$ where our set of nodes is $\mathcal{V} = \{1, 2, \ldots, p\}$. Since nodes are one-dimensional, we replace $\mathbf{S}_{ij}$ with S_{ij}. For $i < j$ we have $(i, j) \in \mathcal{E}$ if and only if $S_{ij} \neq 0$. In this chapter, we assume that the precision matrix has been preconditioned to have $S_{ii} = 1$ for all i. The chapter starts by simplifying the message-update rules for univariate sGaBP. We then proceed to prove the convergence of this algorithm and show that convergence can be obtained by setting the regularisation parameter sufficiently large. We investigate some heuristics aimed at selecting the degree of regularisation. This chapter concludes with an empirical comparison of sGaBP with other GaBP variants, as well as with the CG method.

2.1 Univariate Message Updates

In the case of univariate nodes, the message updates given in Algorithm 2 can be simplified to

$$Q_{ij}^{(n+1)} = \frac{-S_{ij}^2}{\lambda + q_i^{(n)} - Q_{ji}^{(n)}} \tag{2.1}$$

$$V_{ij}^{(n+1)} = \frac{Q_{ij}^{(n+1)}}{S_{ij}}(\lambda\mu_i^{(n-1)} + z_i^{(n)} - V_{ji}^{(n)}) \tag{2.2}$$

$$\mu_i^{(n+1)} = \frac{\lambda\mu_i^{(n)} + z_i^{(n)}}{\lambda + q_i^{(n)}}, \tag{2.3}$$

where $q_i^{(n)} = 1 + \sum_{t\in\mathcal{N}_i} Q_{ti}^{(n)}$ and $z_i^{(n)} = b_i + \sum_{t\in\mathcal{N}_i} V_{ti}^{(n)}$. The updates given in (2.1), (2.2) and (2.3) are valid for all nodes i and all $j \in \mathcal{N}_i$. The use of this notation is in agreement with the notation used in Kamper *et al.* (2018*a*). The simplified version of Algorithm 2 for univariate sGaBP is given in Algorithm 3. We do note that Algorithm 3 uses a different method of computing the convergence threshold (Err) when compared to Algorithm 2. The convergence criterion used in Algorithm 2 is more accurate while the method used by Algorithm 3 is cheaper to compute. The convergence criterion defined by Algorithm 3 is relevant for this chapter while the definition in Algorithm 2 is relevant for the remainder of the thesis.

2.2 Convergence Analysis

In this section we prove the convergence of sGaBP with univariate nodes and discuss some of the steps given in Algorithm 3. We start with a discussion of the computation of the posterior distributions after each iteration. These computations are important, since they are sufficient to ensure the convergence of sGaBP (for large enough λ) while preserving the posterior means as the exact marginal means. This is followed by a study of the convergence behaviour of the precision components of the messages, i.e. the behaviour of $\mathbf{Q}^{(n)} = [Q_{ij}^{(n)}]$ in Algorithm 3. We then proceed to the mean/potential components of the messages by assuming convergence of the precision components. As mentioned, we assume a certain preconditioning of the precision matrix. If the precision matrix is $\mathbf{S}$ (before preconditioning), this can be achieved by setting $\mathbf{D} = \text{diag}(\frac{1}{\sqrt{S_{11}}}, \frac{1}{\sqrt{S_{22}}}, \ldots, \frac{1}{\sqrt{S_{pp}}})$ and computing $\mathbf{DSD}$. The potential vector can be preconditioned by computing $\mathbf{D}\boldsymbol{b}$. This type of preconditioning does not entail any loss of information, in the sense that both the marginal means and precisions of the distribution in its original scale can be recovered.

Algorithm 3 Univariate Synchronous sGaBP

1. Provide $\mathbf{S} : p \times p$ (after preconditioning), $\boldsymbol{b} : p \times 1$ (after preconditioning), λ, m and ϵ as inputs to the algorithm. Here we wish to marginalise a multivariate Gaussian with precision matrix $\mathbf{S}$ and potential vector $\boldsymbol{b}$ into univariate nodes. The parameters λ, m and ϵ denote the regularisation parameter, the maximum number of iterations allowed and the tolerance used to define convergence respectively.

2. Initialise $\mathbf{Q}^{(0)} : p \times p = \text{diag}(1, 1, \ldots, 1)$, $\mathbf{V}^{(0)} : p \times p = \text{diag}(b_1, b_2, \ldots, b_p)$ and $\boldsymbol{\mu}^{(-1)} : p \times 1 = \mathbf{0}$.

3. Set Err = Inf and $n = 0$.

4. While Err $> \epsilon$

   a) Compute $q_i^{(n)} = 1 + \sum_{j \in \mathcal{N}_i} Q_{ji}^{(n)}$ and $z_i^{(n)} = b_i + \sum_{j \in \mathcal{N}_i} V_{ji}^{(n)}$ for $i = 1, 2, \ldots, p$.

   b) Set $\mu_i^{(n)} = \frac{\lambda \mu_i^{(n-1)} + z_i^{(n)}}{\lambda + q_i^{(n)}}$ for $i = 1, 2, \ldots, p$.

   c) For all i and all $j \in \mathcal{N}_i$, set $Q_{ij}^{(n+1)} = \frac{-S_{ij}^2}{\lambda + q_i^{(n)} - Q_{ji}^{(n)}}$ and $V_{ij}^{(n+1)} = \frac{Q_{ij}^{(n+1)}}{S_{ij}} (\lambda \mu_i^{(n-1)} + z_i^{(n)} - V_{ji}^{(n)})$.

   d) Set Err $= \sqrt{\frac{\sum_k (\mu_k^{(n)} - \mu_k^{(n-1)})^2}{\sum_k (\mu_k^{(n)})^2}}$ and increment n.

   e) If $n = m$, break.

5. End.

2.2.1 Computation of Posterior Distributions

In order to ensure convergence of sGaBP and to have the posterior means (at convergence) equal to the correct marginal means, it is necessary to adjust the manner in which posterior distributions are computed. Consider the computation of the posterior distribution of node i at stage n. As a first step, we instruct node i to collect all incoming messages, which can be characterised by the parameters $\sum_{t \in \mathcal{N}_i} Q_{ti}^{(n)}$ (precision components) and $\sum_{t \in \mathcal{N}_i} V_{ti}^{(n)}$ (mean/potential components). We suggest keeping the posterior precisions as in normal belief propagation, that is $1 + \sum_{t \in \mathcal{N}_i} Q_{ti}^{(n)}$. Later we investigate the role of λ in the tuning of the posterior precisions to better approximate the marginal precisions. The posterior mean of node i at stage n is given by

$$\mu_i^{(n)} = \frac{\lambda \mu_i^{(n-1)} + z_i^{(n)}}{\lambda + q_i^{(n)}} = \gamma_i^{(n)} \mu_i^{(n-1)} + (1 - \gamma_i^{(n)}) \frac{z_i^{(n)}}{q_i^{(n)}}, \tag{2.4}$$

where $z_i^{(n)} = b_i + \sum_{t \in \mathcal{N}_i} V_{ti}^{(n)}$, $q_i^{(n)} = 1 + \sum_{t \in \mathcal{N}_i} Q_{ti}^{(n)}$ and $\gamma_i^{(n)} = \frac{\lambda}{\lambda + q_i^{(n)}}$. Note that $\frac{z_i^{(n)}}{q_i^{(n)}}$ is the posterior mean we would have computed if no adjustment was made to the computation of the posterior distribution. Hence, we can interpret (2.4) as damping between the posterior mean, under normal belief propagation, and the posterior mean computed in the previous round. What is attractive here is that these damping factors are computed automatically (using λ and the current posterior precisions), and no additional parameters are required. The values, $\gamma_i^{(n)} : i = 1, 2, \ldots, p$, can also be relaxation factors that correspond to negative λ. The rest of this section is dedicated towards showing that these adjustments are sufficient for the convergence of sGaBP and the preservation of the (converged) posterior means as the exact marginal means.

2.2.2 The Precision Components

The convergence analysis of the precision components is simpler, compared to the mean components, since we can apply results found in the literature (Malioutov *et al.*, 2006; Bickson, 2008). This is because the analysis of $\mathbf{Q}^{(n)}$ is identical to that of the precision components provided by ordinary GaBP applied to the matrix $\lambda\mathbf{I} + \mathbf{S}$. Therefore, we only need to select λ large enough for $\lambda\mathbf{I} + \mathbf{S}$ to be walk-summable (although smaller selections of λ can also suffice) to obtain convergence. Selecting $\lambda > \tilde{\rho}(|\mathbf{S}|) - 1$ is sufficient for the precision components in Algorithm 3 to converge since it guarantees $\lambda\mathbf{I} + \mathbf{S}$ to be walk-summable. An alternative proof of convergence is given in Theorem 1 and the proof is contained in Appendix A.

Theorem 1 *Let $I_n = 1$ if all of the following conditions hold at iteration n ($I_n = 0$ otherwise):*

1. $Q_{ij}^{(n)} \leq 0$ *for all* $i, j \in \mathcal{N}_i$.
2. $|Q_{ij}^{(n)}| > |Q_{ij}^{(n-1)}|$ *for all* $i, j \in \mathcal{N}_i$.
3. *For all* i, $\delta_i^{(n)} = \sum_{t \in \mathcal{N}_i} |Q_{ti}^{(n)}| \leq \delta_i$ *for a* δ_i *satisfying* $0 \leq \delta_i < 1 + \lambda$.
4. $\sum_{t \in \mathcal{N}_i} \frac{S_{ti}^2}{1 + \lambda - \delta_t + |Q_{it}^{(n)}|} \leq \delta_i$ *for all* i.

If $I_n = 1$, then $I_k = 1$ for all $k > n$.

If $I_k = 1$, then the $Q_{ij}^{(n)}$s are monotone decreasing and bounded from below by $\frac{-S_{ij}^2}{1 + \lambda - \delta_i}$, for $k > n$, and will therefore converge. Consider the case where $\delta_i = \delta$ for all i and suppose we want $I_0 = 1$. Since $Q_{ij}^{(0)} = 0$ for $i \neq j$, we

need a $0 \leq \delta < 1 + \lambda$ satisfying $\sum_{t \in \mathcal{N}_i} \frac{S_{ti}^2}{1+\lambda-\delta} \leq \delta$ for all i. This inequality is equivalent (for $1 + \lambda - \delta > 0$) to a quadratic inequality in δ with roots,

$$\frac{(1+\lambda) \pm \sqrt{(1+\lambda)^2 - 4\sum_{t \in \mathcal{N}_i} S_{ti}^2}}{2}. \tag{2.5}$$

If we select $(1+\lambda)^2 - 4 \times \max_i \left\{ \sum_{t \in \mathcal{N}_i} S_{ti}^2 \right\} \geq 0$ which is satisfied by $\lambda \geq 2\sqrt{\max_i \left\{ \sum_{t \in \mathcal{N}_i} S_{ti}^2 \right\}} - 1$, then we can select $\delta = \frac{1+\lambda}{2}$ to guarantee monotone convergence of all the precisions. For this selection, the bounds on the precisions are

$$-\frac{2S_{ij}^2}{1+\lambda} \leq Q_{ij}^{(n)} \leq 0. \tag{2.6}$$

An important consequence of (2.6) is that

$$\lim_{\lambda \to \infty} Q_{ij}^{(n)} = 0 \tag{2.7}$$

for all $j \in \mathcal{N}_i$ and for all n. Here we emphasise the role of λ in the tuning of the posterior precisions. Note that we can tune the converged precision components, Q_{ij}, to be in the interval $[-\frac{2S_{ij}^2}{1+\lambda_0}; 0]$ (this interval can be wider), where $\lambda_0 = 2\sqrt{\max_i \left\{ \sum_{t \in \mathcal{N}_i} S_{ti}^2 \right\}} - 1$, although there is dependence among the Q_{ij}'s in terms of λ. This, in turn, can be used to tune the posterior precisions, $1 + \sum_{t \in \mathcal{N}_i} Q_{ti}$, under certain restrictions. The tuning can be made more flexible by introducing multiple tuning parameters.

2.2.3 The Mean Components

In the previous section, we saw that the precision components of the messages will converge for sufficiently large choices of λ. In this section, we proceed under the assumption that the precision components have converged (which is guaranteed for sufficiently large λ). Nothing precludes the marginal means to have converged at this stage, although, based on simulations, this is not usually the case. We denote the converged precision message components, posterior precisions and damping factors by Q_{ij}, q_i and γ_i respectively. The updates of the mean components are

$$V_{ij}^{(n+1)} = \frac{Q_{ij}}{S_{ij}} \left[\lambda \mu_i^{(n-1)} + b_i + \sum_{t \in \mathcal{N}_i \setminus j} V_{ti}^{(n)} \right]. \tag{2.8}$$

We define $\boldsymbol{\theta}^{(n+1)}$ to be the vector obtained by stacking the column vectors of $\mathbf{V}^{(n+1)}$, where the diagonal entry associated with each column vector is

ignored, and appending $\boldsymbol{\mu}^{(n)}$ (after the column vectors of $\mathbf{V}^{(n+1)}$). This vector can be expressed as

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta} + \mathbf{L}\boldsymbol{\theta}^{(n)} \tag{2.9}$$

for a matrix $\mathbf{L} : p^2 \times p^2$ and a vector of constants $\boldsymbol{\theta} : p^2 \times 1$. The first $l = p^2 - p$ entries of $\boldsymbol{\theta}$ can be obtained by constructing the matrix $\mathbf{C} = [\frac{Q_{ij}}{S_{ij}} b_i]$; with the understanding that the diagonals are zero. We then compute the first l entries of $\boldsymbol{\theta}$ in the same way as we computed those of $\boldsymbol{\theta}^{(n+1)}$, using $\mathbf{C}$ instead of $\mathbf{V}^{(n+1)}$. The final p entries of $\boldsymbol{\theta}$ are $\frac{1-\gamma_i}{q_i} b_i$ in order $i = 1, 2, \ldots, p$.

The construction of $\mathbf{L}$ is more complex. We use the following decomposition,

$$\mathbf{L} : p^2 \times p^2 = \begin{bmatrix} \mathbf{L}_{11} : l \times l & \mathbf{L}_{12} : l \times p \\ \mathbf{L}_{21} : p \times l & \mathbf{L}_{22} : p \times p \end{bmatrix}. \tag{2.10}$$

Consider one of the first l elements of $\boldsymbol{\theta}^{(n+1)}$, say m. This element corresponds to an entry in the matrix $\mathbf{V}^{(n+1)}$, say $V_{ij}^{(n+1)}$. The next step is to identify the neighbours of i, that is the set $\mathcal{N}_i$. For each $k \in \mathcal{N}_i \setminus j$, we find the element in $\boldsymbol{\theta}^{(n)}$ corresponding to $V_{ki}^{(n)}$ and note its position. The entry in row m of $\mathbf{L}$ in this position is $\frac{Q_{ij}}{S_{ij}}$. This accounts for the matrix $\mathbf{L}_{11}$, with the understanding that all elements not accessed are zero. Continuing with this notation, the entry in row m of $\mathbf{L}_{12}$ in position i is $\frac{\lambda Q_{ij}}{S_{ij}}$, and all other elements in this row are zero. We see that $\mathbf{L}_{22}$ is a diagonal matrix with entries γ_i in order $i = 1, 2, \ldots, p$. Consider the matrix $\mathbf{L}_{21}$. The first step is to identify the neighbours of node i, that is $\mathcal{N}_i$. We then move along the vector $\boldsymbol{\theta}^{(n)}$ and identify all the positions corresponding to $V_{ti}^{(n)} : t \in \mathcal{N}_i$. In row i of $\mathbf{L}_{21}$ we place the value $\frac{1-\gamma_i}{q_i}$ in the identified positions, and the rest of the entries are zero.

Our goal is to analyse the spectral radius of $\mathbf{L}$. We note that the eigenvalues of $\mathbf{L}$ can possibly be complex. If the spectral radius of $\mathbf{L}$ is less than 1, sGaBP will converge (assuming that the precisions converge). The value of the spectral radius has a heavy influence on the convergence speed of sGaBP and can play a role in deciding how to select λ. A natural way of selecting the level of regularisation is to seek λ such that the spectral radius (of $\mathbf{L}$) is a minimum. We make some comments on the form of the spectrum later in this section. For now we consider the asymptotic behaviour of the spectral radius and show that the spectral radius approaches 1 from below as $\lambda \to \infty$. The selection of λ is considered in the section on heuristic measures. Theorem 2 provides information on the asymptotic behaviour of the spectrum, and the proof is given in Appendix A.

Theorem 2 *Consider sGaBP applied to a multivariate Gaussian with potential* $\boldsymbol{b}$ *and precision matrix* $\mathbf{S} : p \times p = [S_{ij}]$. *The eigenvalues of the linear*

update matrix $\mathbf{L}$ *can be characterised as*

$$1 - \frac{\sigma_i}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2});\ i = 1, 2, \ldots, p \tag{2.11}$$

$$\frac{\pm S_{ij}}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2});\ 1 \leq i < j \leq p, \tag{2.12}$$

where $\sigma_i, i = 1, 2, \ldots, p$ *represent the eigenvalues of* $\mathbf{S}$.

In particular, we see that, if $\mathbf{S}$ is positive definite, the maximum of the eigenvalues in Theorem 2 tends to 1 from below as $\lambda \to \infty$. We see that the precisions will converge for large enough λ and will eventually generate a linear update matrix with a spectral radius of less than 1; that is, sGaBP will converge for large enough λ. In Appendix A we show that the posterior means provided by sGaBP (under the assumption of convergence) match the exact marginal means.

We now make some comments on the behaviour of the spectrum (eigenvalues) of $\mathbf{L}$. One interesting aspect of the spectrum is when sGaBP is applied to tree-structured precision matrices. We generated a few of these and in each case we found the matrix $\mathbf{L}$ to be nilpotent when $\lambda = 0$. This relates to BP as an efficient and exact marginalisation algorithm on tree structures. The use of values of λ other than zero is nonsensical in this case.

A typical plot of the spectral radius as a function of λ is given in Figure 2.1. In this case, the spectral radius has a global minimum at a value of λ just under 0.4. The spectral radius can correspond to either a complex or a real eigenvalue, and the graph of the spectral radius seems to change curvature when the eigenvalue responsible for the spectral radius changes from real to complex (and vice versa). This can be seen in Figure 2.1, with the solid red line corresponding to a complex spectral radius and the broken black line to a real spectral radius. We also see that the spectral radius eventually becomes real, which is consistent with Theorem 2. Another important observation is that the value of λ that minimises the spectral radius seems to occur at a point where the eigenvalue responsible switches between real and complex. There can be more than one point where this change occurs. Our simulations show similar results for other precision matrices, suggesting that this result holds generally. The interaction between complex and real eigenvalues could prove useful in the minimisation of the spectral radius and should be considered in further research.

2.2.4 The Converged Posteriors

Having proved the convergence of sGaBP, we now turn to the posterior distributions as approximations of the marginal distributions. Theorem 3 states

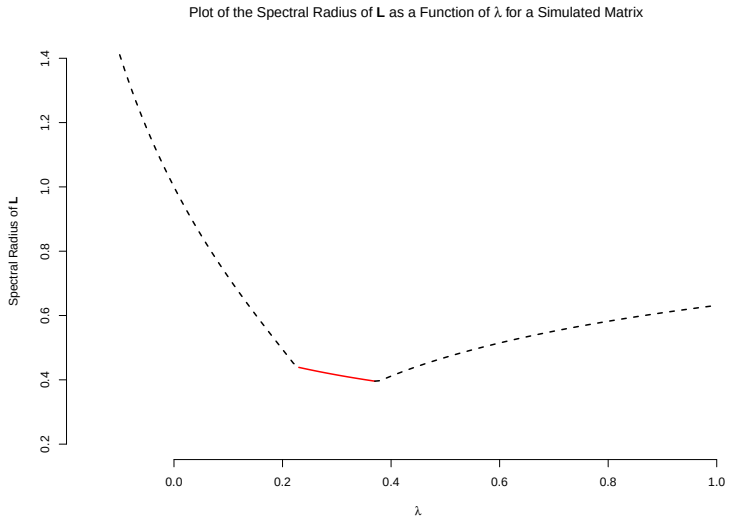


Figure 2.1: Plot of the spectral radius of the linear update matrix as a function of λ for a simulated 10×10 matrix with a zero-diagonal spectral radius equal to one. The solid red line corresponds to a complex spectral radius. The broken black line corresponds to a real spectral radius.

that the posterior means are the exact marginal means (the proof is provided in Appendix A).

Theorem 3 *Under the assumption that sGaBP converges, with precision matrix* $\mathbf{S}$ *and potential vector* $\boldsymbol{b}$ *as inputs and setting* $\boldsymbol{\mu}$ *equal to the converged posterior means, we have that* $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$.

A consequence of Theorem 3 is that sGaBP can be used to solve linear systems, $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$, as long as $\mathbf{S}$ is a valid precision matrix. Unfortunately, the posterior precisions are not necessarily equal to the true marginal precisions. In the empirical section we will investigate the Kullback-Leibler (KL) divergence of the exact marginal distributions to the posterior distributions. By this we mean that, if $\hat{f}$ is the posterior distribution (suitably normalised) and f is the true marginal, we are computing

$$D_{KL}(f||\hat{f}) = \int_{\mathbf{y}\in\Re^l} f(\mathbf{y}) \log\left(\frac{f(\mathbf{y})}{\hat{f}(\mathbf{y})}\right) d\mathbf{y}, \tag{2.13}$$

where we assume that the density functions involve l variables. In the experimental section we show that the posterior precisions provided by sGaBP can be useful as approximate quantities in the sense that the KL divergence defined in (2.13) can be small. The results compare well with other GaBP-based methods in this regard.

[1] since we are comparing two Gaussian distributions, we have $\hat{f}(\mathbf{y}) > 0$ and $f(\mathbf{y}) > 0$ for all $\mathbf{y} \in \Re^l$.

Algorithm 4 Heuristic Selection of λ

1. In Algorithm 3, Step 1, add the specification of a lag (d) and a step size (α).
2. In Algorithm 3, Step 2, add the initialisation $e^{\text{best}} = \max_i |b_i|$.
3. Before Algorithm 3, Step 4a, add the following:
   a) If $\text{mod}(n,d) = 0$
      - If $e^{\text{best}} > \text{Err}$ then $\lambda \leftarrow \lambda - \alpha$ and $e^{\text{best}} \leftarrow \text{Err}$, else $\lambda \leftarrow \lambda + \alpha$.

2.3 Heuristic Measures

In this section we propose some heuristic measures for the selection of λ. These measures vary in degree of complexity, and we discuss some of their advantages and disadvantages.

2.3.1 Search Heuristic

The search heuristic (SH) is basically the same as the one proposed by El-Kurdi *et al.* (2012*a*), adjusted for sGaBP, and is given in Algorithm 4. The main advantage of this heuristic is that it is easy to implement. There are some drawbacks to this measure, arising from the monotone way in which the tuning is adjusted. When the current tuning provides posterior means with a smaller (larger) error, the heuristic will always decrement (increment) the tuning. The heuristic seeks tuning for which the spectral radius of $\mathbf{L}$ is less than one, and not necessarily tuning for which the value of the spectral radius is a minimum.

2.3.2 Gradient Descent Heuristic

The gradient descent heuristic (GDH) is more complex to implement, but does not have the monotonicity of the SH as described in Section 2.3.1. The heuristic is aimed at determining the direction in which the tuning needs to be adjusted to achieve the smallest possible spectral radius. The tuning is then adjusted in this direction in steps, the size of which should not be overly large.

Suppose we have completed iteration n of sGaBP and are preparing to perform the next round of updates. We wish to adjust the value of the tuning parameter in a direction that yields faster convergence. At this point, we have the posterior precisions $q_i^{(n)}$ and posterior means $\mu_i^{(n)}$ for $i = 1, 2, \ldots, p$. The

posterior means are computed using

$$\begin{aligned}\mu_i^{(n)} &= \frac{q_i^{(n)}}{\lambda + q_i^{(n)}} \frac{z_i^{(n)}}{q_i^{(n)}} + \frac{\lambda}{\lambda + q_i^{(n)}} \mu_i^{(n-1)} \\ &= [1 - \gamma_i^{(n)}(\lambda)]\tilde{\mu}_i^{(n)} + \gamma_i^{(n)}(\lambda)\mu_i^{(n-1)}, \end{aligned} \tag{2.14}$$

where $\tilde{\mu}_i^{(n)} = \frac{z_i^{(n)}}{q_i^{(n)}}$ Although this is not technically correct, we assume that $q_i^{(n)}, z_i^{(n)}$ and $\mu_i^{(n-1)}$ are constant and do not depend on λ. The GDH starts by instructing each node to send its posterior mean to its neighbours. Each node then computes $e_j = \sum_{i \in \tilde{\mathcal{N}}_j} S_{ji}\mu_i^{(n)} - b_j$, where $\tilde{\mathcal{N}}_j = \mathcal{N}_j \cup \{j\}$. Let $k = \text{argmax}_j\{|e_j|\}$. The node k and each of its neighbours are instructed to compute the derivative of their own mean (can be done in parallel) by differentiating (2.14) with respect to λ and evaluating this at the current value of the tuning, say λ_0. The neighbours of node k send these derivatives to node k, and this node computes $d_k = \sum_{j \in \tilde{\mathcal{N}}_k} S_{kj} \bigtriangledown \mu_j^{(n)}$, where $\bigtriangledown\mu_j^{(n)}$ is the derivative received from node j. Node k is then instructed to adjust the tuning $\lambda_0 \leftarrow \lambda_0 - \alpha\,\text{sign}(d_k)$, for a specified step size parameter α, and to send this new tuning value to the other nodes.

2.3.3 Comparing SH and GDH: A Concrete Example

We use simulation to illustrate the possible benefits of using the GDH instead of the SH. We start by simulating a 100×100 precision matrix, $\mathbf{S}$, and potential vector, $\boldsymbol{b}$. We use the method in Appendix A.4 to regulate the zero-diagonal spectral radius of the precision matrix to 1. We define convergence to occur when the error is less than 10^{-14}. Using a line search in increments of 0.01, we observed that initialising the tuning of sGaBP with values $0.33, 0.34$ and 0.35 yielded the fastest convergence, and that this occurred after 28 iterations. We applied two instances for each of GDH and SH where, for each method, we considered the step sizes 0.01 and 0.05. The results of this experiment are given in Table 2.1 and displayed in Figure 2.2.

Table 2.1: Comparison of SH and GDH

Method	Step size	# of iterations
Optimal	NA	28
GDH	0.01	45
SH	0.05	>100
GDH	0.01	32
SH	0.05	>100

For the simulated precision matrix, we found that the spectral radius of $\mathbf{L}$ is 1 when $\lambda = 0$ (this is typical when the zero-diagonal spectral radius of $\mathbf{S}$ is 1). The values of the spectral radius (of $\mathbf{L}$) corresponding to $\lambda = -0.01$ and $\lambda = 0.01$ are 1.029343 and 0.971501 respectively. Assuming convergence of the precision components of the messages, we observed the error to be increasing for negative tuning and decreasing for positive tuning. If the SH is used, there is the risk that the heuristic tuning will vary around the tuning corresponding to a spectral radius of one. This is because the SH seeks tuning for which the spectral radius of $\mathbf{L}$ is less than one, and not necessarily tuning that minimises the spectral radius of $\mathbf{L}$. This is illustrated in Figure 2.2. The y-axis shows the level of tuning used at the iteration number given on the x-axis. Figure 2.2 contains two lines for each of GDH and SH. These lines correspond to the step sizes 0.01 and 0.05. The tuning suggested by the SH varies around $\lambda = 0$, the level of tuning corresponding to a spectral radius of 1. The GDH is not restricted to a spectral radius of one and is able to make better adjustments on the tuning. Notice that the two graphs corresponding to the GDH are terminated at iteration 45 and 32, corresponding to step sizes 0.01 and 0.05 respectively. This was done to indicate that sGaBP had converged after these numbers of iterations. Both applications of the SH failed to converge after 100 iterations. This is not to say that the SH cannot be effective. Indeed, its simplicity of implementation is an advantage over the GDH, but the SH is more sensitive to the initialisation of λ, particularly when this starting value is close to the level of tuning yielding a spectral radius of one.

2.4 Asynchronous Message Updates

We have referred to the use of asynchronous message updates as opposed to the synchronous version. It is generally believed that asynchronous message updates can provide better convergence behaviour in applications of BP, in the sense that they may induce convergence where synchronous updates diverge or require the passing of a smaller number of messages to converge (Koller and Friedman, 2009). The major shortcoming of asynchronous updates is loss of distributive applicability, since the update of the messages cannot be decentralised. Another problem posed by asynchronous updates is deciding upon the order in which messages are passed, since this can have a significant effect on the convergence speed. In the context of GaBP, this problem is compounded by the fact that synchronous messages operate in iterations with $\mathcal{O}(p^2)$ computations, which discounts the complicated heuristics used in other applications of BP to decide on the message scheduling. Progress can be made by deciding on the message scheduling in advance. There are other considerations as well, such as deciding on the degree of regularisation. This should be considered from the viewpoint that the degree of regularisation yielding optimal conver-

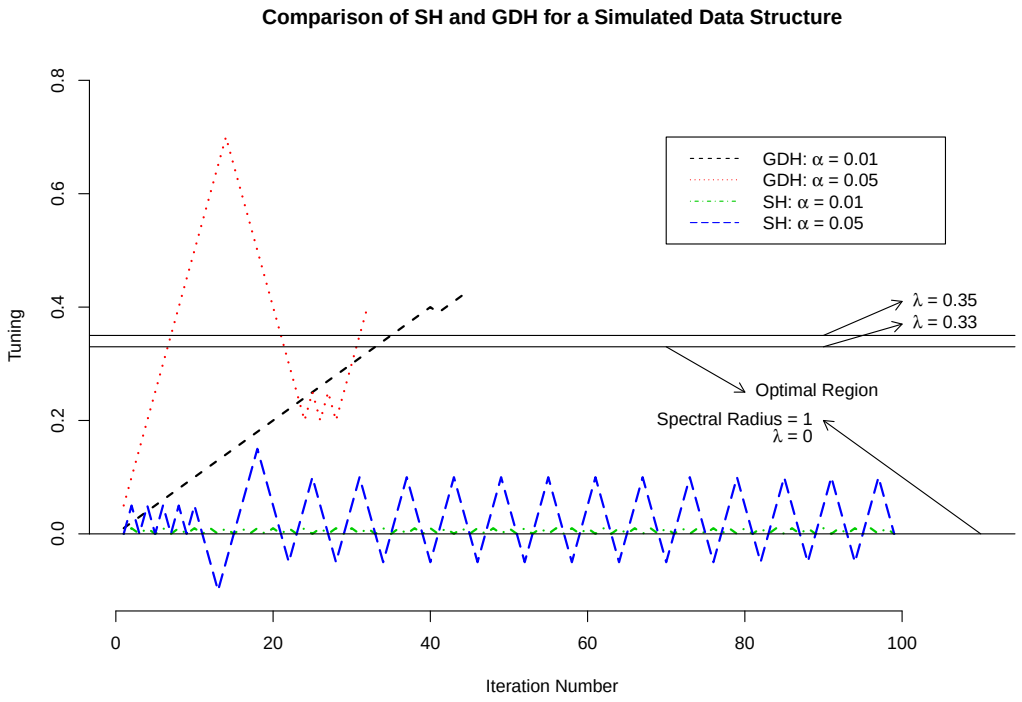


Figure 2.2: Comparison of SH and GDH for a simulated data structure. The precision matrix was regulated to have a zero-diagonal spectral radius equal to one. Some relevant quantities are given in the display. We see that the tuning provided by SH is stuck around $\lambda = 0$. Selecting $\lambda = 0$ a priori gives a spectral radius equal to one. GDH provides tuning closer to the values yielding the fastest convergence (determined using a line search in increments of 0.01). GDH converges faster than SH.

gence should naturally provide useful posterior precisions. An example of the advantages of asynchronous message passing can be found in the diabetes data (Efron *et al.*, 2004). The diabetes data was used to illustrate the advantages of the least-angle regression algorithm in settings involving a high degree of collinearity among the explanatory variables. Estimating the linear coefficients of the diabetes data is challenging for GaBP, since:

1. The number of explanatory variables is small (this is an empirical observation).

2. The zero-diagonal spectral radius of the sample correlation matrix is high (3.024214).

3. There is significant variation among the sample correlations (this is an empirical observation).

Algorithm 5 Univariate Asynchronous sGaBP

1. Provide $\mathbf{S} : p \times p$ (after preconditioning), $\boldsymbol{b} : p \times 1$ (after preconditioning), λ, m and ϵ as inputs to the algorithm. Here we wish to marginalise a multivariate Gaussian with precision matrix $\mathbf{S}$ and potential vector $\boldsymbol{b}$ into univariate nodes. The parameters λ, m and ϵ denote the degree of regularisation, the maximum number of iterations allowed and the tolerance used to define convergence respectively.

2. Initiate $\mathbf{Q} : p \times p = \text{diag}(1, 1, \ldots, 1)$, $\mathbf{V} : p \times p = \text{diag}(b_1, b_2, \ldots, b_p)$, $\boldsymbol{\mu} : p \times 1 = \mathbf{0}, \mathbf{q} = (1, 1, \ldots, 1)'$ and $\mathbf{z} : p \times 1 = \boldsymbol{b}$.

3. Set Err = Inf and $n = 0$.

4. While Err $> \epsilon$

   a) For $i = 1, 2, \ldots, p$, set $\mu_i^{\text{old}} = \mu_i$.

   b) For $j = 1, 2, \ldots, p$

      For $i = 1, 2, \ldots, p$

      i. If $j = i$ or $S_{ij} = 0$ then continue.

      ii. Set $a_1 = Q_{ij}, a_2 = V_{ij}$.

      iii. Update $Q_{ij} = \frac{-S_{ij}^2}{\lambda + q_i - Q_{ji}}$ and $V_{ij} = \frac{Q_{ij}}{S_{ij}}(\lambda \mu_i + z_i - V_{ji})$.

      iv. Update $q_j = q_j - a_1 + Q_{ij}$ and $z_j = z_j - a_2 + V_{ij}$.

      v. Update $\mu_j = \frac{\lambda \mu_j + z_j}{\lambda + q_j}$.

   c) Set Err $= \sqrt{\frac{\sum_k (\mu_k - \mu_k^{\text{old}})^2}{\sum_k (\mu_k)^2}}$ and increment n.

   d) If $n = m$ break.

5. End.

A line search in increments of 0.01 revealed that $\lambda = 1.29$ yields the fastest convergence for synchronous sGaBP (using a tolerance of 10^{-10}), and convergence occurred after 574 iterations. The 574 iterations required for convergence is substantial when compared to the number of explanatory variables (which is 10). A further complication is that synchronous sGaBP with $\lambda = 1.29$ yields negative posterior precisions for certain variables, although this can be addressed by increasing λ at the cost of slower convergence.

We now apply asynchronous sGaBP, which is formulated in Algorithm 5. Notice that the outer loop of the message updates iterates over j, indicating that the inner loop iterates over messages to node j. This was done because we found that iterating over incoming messages first was more efficient in our simulations. Each round of message updates requires $\mathcal{O}(p^2)$ computations (fewer

with sparsity), as in the synchronous case. Unlike the synchronous case it is not necessary to store old messages, and Algorithm 5 performs damping throughout the double loop (instead of after). The optimal tuning value was determined as 2.01 (line search as for the synchronous case), and convergence occurred after 131 iterations. All posterior precisions were positive. We see that asynchronous message passing improved the convergence speed and accuracy of the posterior distributions in the case of the diabetes data.

In general, our simulations show that asynchronous outperforms synchronous sGaBP in terms of convergence behaviour. This is further compounded by the fact that the asynchronous message passing does not require old messages to be stored, resulting in a lower computational burden on each iteration and lower memory requirements. We leave further aspects of asynchronous sGaBP, such as proof of convergence and heuristic measures, for further research.

2.5 Empirical Work

In this section we provide empirical comparisons of sGaBP with other GaBP variants in the literature, as well as with the CG solver. Our empirical work will be summarised using two quantities, namely the number of iterations required by a specified method to converge and, if relevant, the KL divergence of the true marginal distributions to the posterior distributions. All quantities are summarised using boxplots, with the blue boxplots representing sGaBP and the red boxplots the method it is being compared with. Each figure corresponds to a set of zero-diagonal spectral radii, which are indicated on the x-axis. For every zero-diagonal spectral radius indicated on the x-axis, we generate 100 data structures, each consisting of a precision matrix and a potential vector. We use the method described in Appendix A.4 to regulate the zero-diagonal spectral radius of the precision matrix to the appropriate value. We then apply sGaBP and the method it is being compared with to these data structures. With the exception of the CG solver, all other methods require the specification of hyper-parameter(s). We initialise these methods by finding the value(s) of the hyper-parameter(s) yielding the fastest convergence through a line (grid) search in increments of 0.01. We refer to sGaBP (for instance), initialised with the optimal hyper-parameter determined through the line search, as optimal sGaBP. Similar labels are used for the other methods.

We now have 100 data structures for every zero-diagonal spectral radius given on the x-axis of the figures. We apply optimal sGaBP and the (optimised) competitor and record the number of iterations required by each method to converge. The blue boxplot is constructed from the number of iterations required by sGaBP to converge, and the red boxplot from the number of iterations required by the competitor. The KL divergences are slightly more

complicated, since for each precision matrix we get multiple marginals. For each application of sGaBP (and its competitor), we determine the KL divergence of all the exact marginals to their respective posterior distributions , and a given data structure is represented by the mean of all these divergences (we refer to this as the mean KL divergence of the data structure). Boxplots are then constructed in a similar way to those for the iterations. To account for differences in the scaling of quantities provided by different methods, it may be necessary to focus (or zoom in) on certain parts of a figure.

2.5.1 Relaxed Gaussian Belief Propagation

El-Kurdi *et al.* (2012*a*) illustrate the advantages of RGaBP on large ill-conditioned and weakly diagonally dominant inverse covariance matrices. RGaBP does not allow tuning of the precision components and can therefore only be applied in settings where the precision components of ordinary GaBP converge. The relaxation is applied to the mean components by setting $z_i^{(n)} = \gamma(b_i + \sum_{j\in\mathcal{N}_j} V_{ji}^{(n)}) + (1-\gamma)q_i^{(n)}\mu_i^{(n-1)}$. Setting $\gamma = 1$ gives ordinary GaBP (similar to setting $\lambda = 0$ for sGaBP). Although El-Kurdi *et al.* (2012*a*) focus on relaxation factors ($\gamma > 1$), RGaBP can also be used to perform damping ($\gamma < 1$). There is an interesting relationship between RGaBP and sGaBP with regard to how posterior means are computed:

$$\text{RGaBP} : \mu_i^{(n)} = \gamma\frac{b_i + \sum_{j\in\mathcal{N}_j} V_{ji}^{(n)}}{q_i^{(n)}} + (1-\gamma)\mu_i^{(n-1)} \tag{2.15}$$

$$\begin{aligned}\text{sGaBP} : \mu_i^{(n)} &= \frac{q_i^{(n)}}{\lambda + q_i^{(n)}}\frac{b_i + \sum_{j\in\mathcal{N}_j} V_{ji}^{(n)}}{q_i^{(n)}} \\ &\quad + \frac{\lambda}{\lambda + q_i^{(n)}}\mu_i^{(n-1)}.\end{aligned} \tag{2.16}$$

We see that $\gamma = 1 - \frac{\lambda}{\lambda + q_i^{(n)}}$. In contrast to RGaBP, sGaBP computes adaptive damping/relaxation factors using the tuning parameter λ and the posterior precisions. In particular, we see that relaxation, $\gamma > 1$, and damping, $\gamma < 1$, correspond to negative and positive λ respectively. This would suggest that there is a role to play for negative λ. Part of our comparison is to give an indication of when to use relaxation versus damping. It is also worthwhile to emphasise that the posterior precisions provided by RGaBP are the posterior precisions provided by ordinary GaBP. Another important contribution in this regard is to provide empirical evidence that sGaBP can provide posterior precisions closer to the true marginal precisions when compared to ordinary GaBP.

In Figure 2.3, the convergence speed of optimal sGaBP and optimal RGaBP

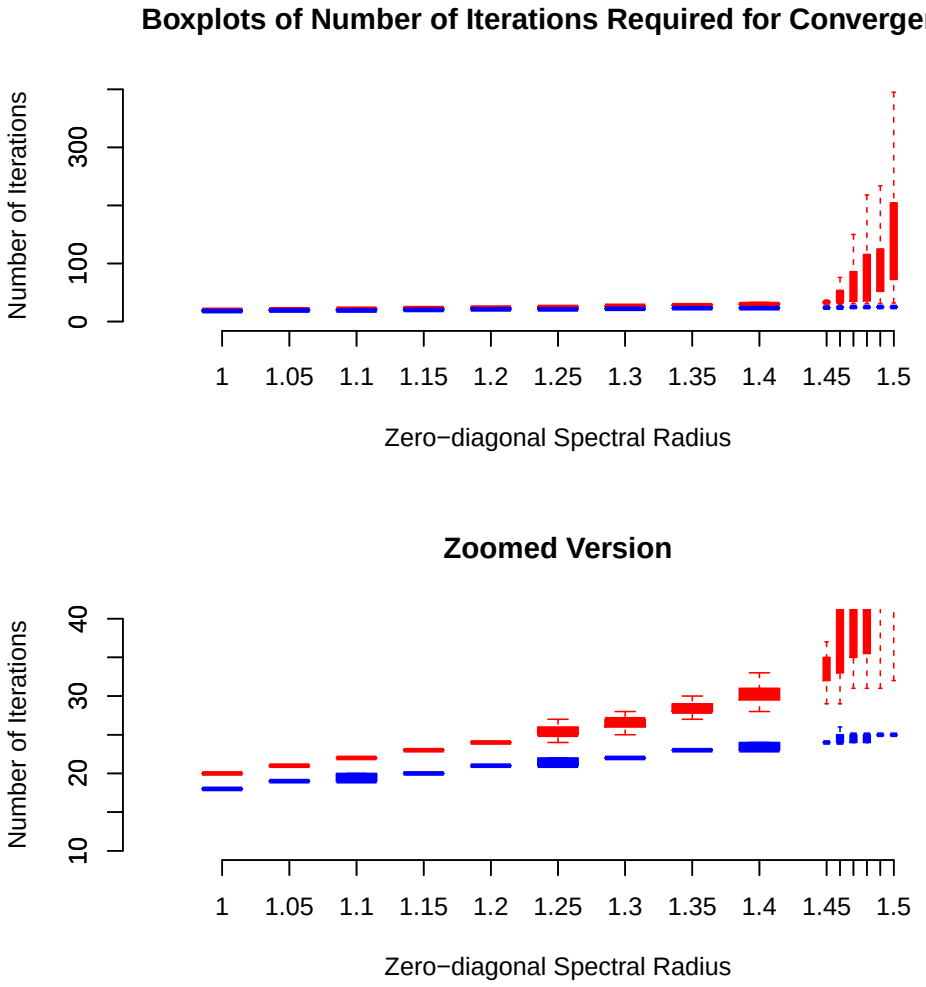


Figure 2.3: Comparison of the convergence speed of optimal sGaBP (blue) and optimal RGaBP (red) over different zero-diagonal spectral radii. Precision matrices were generated to be of dimension 442×442. sGaBP outperformed RGaBP in these simulations, with the relative convergence speed of RGaBP tending to decrease as the zero-diagonal spectral radius increases.

are compared. For smaller zero-diagonal spectral radii the methods are very similar, with sGaBP holding a slight advantage. As the zero-diagonal spectral radius approaches 1.5, the convergence speed of RGaBP starts to deteriorate. When considering the boxplot corresponding to a zero-diagonal spectral radius of 1.5, we see that sGaBP can converge up to 16 times faster than RGaBP (it is also worthwhile noting that outliers were suppressed in these boxplots).

In Figure 2.4, the mean KL divergences of optimal sGaBP and optimal RGaBP

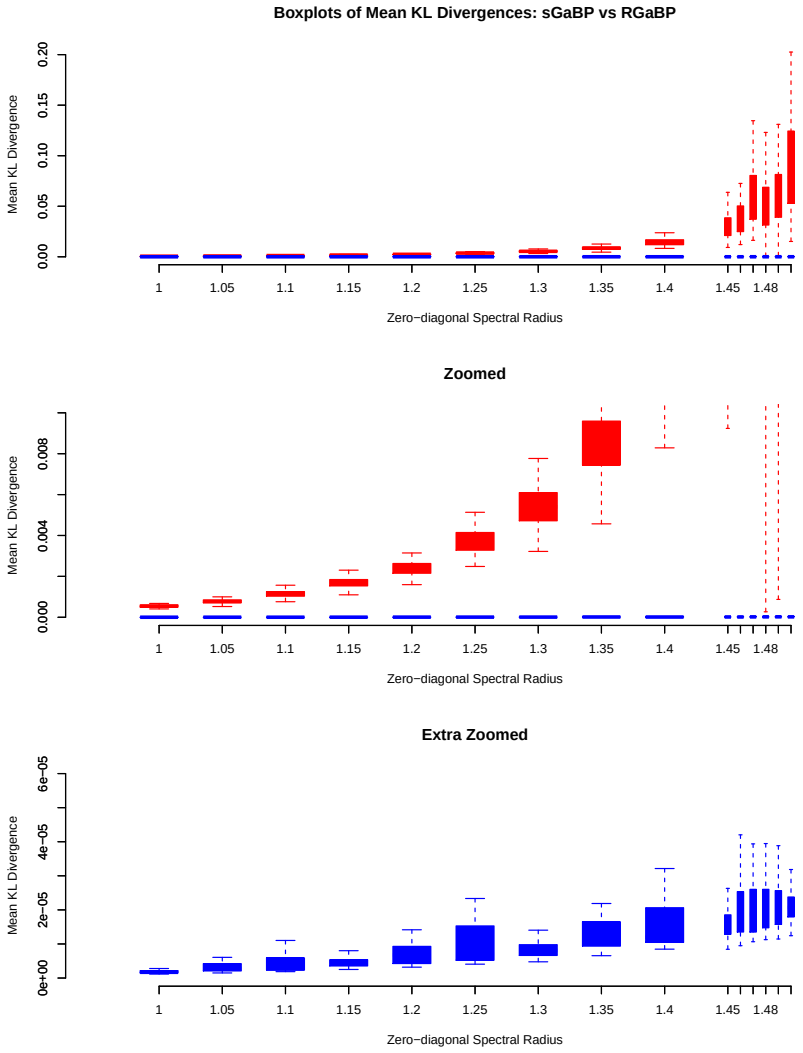


Figure 2.4: This is similar to Figure 2.3, but the boxplots now represent the mean KL divergence provided by each method. In these simulations, sGaBP (blue) provided more accurate approximations of the true marginals.

are compared. In the simulations, sGaBP provided far more accurate posterior distributions. The simulations provide evidence, even in cases where the optimal convergence speeds are comparable, that it is better to use sGaBP instead of RGaBP, since sGaBP provides posterior precisions closer to the true marginal precisions.

An interesting sub-plot is the role of relaxation versus damping in the acceleration of GaBP. Relaxation corresponds to $\gamma > 1$, or negative λ, while damping occurs when $\gamma < 1$, or positive λ. The zero-diagonal spectral radius of a precision matrix can be determined by one of two quantities, these being either the

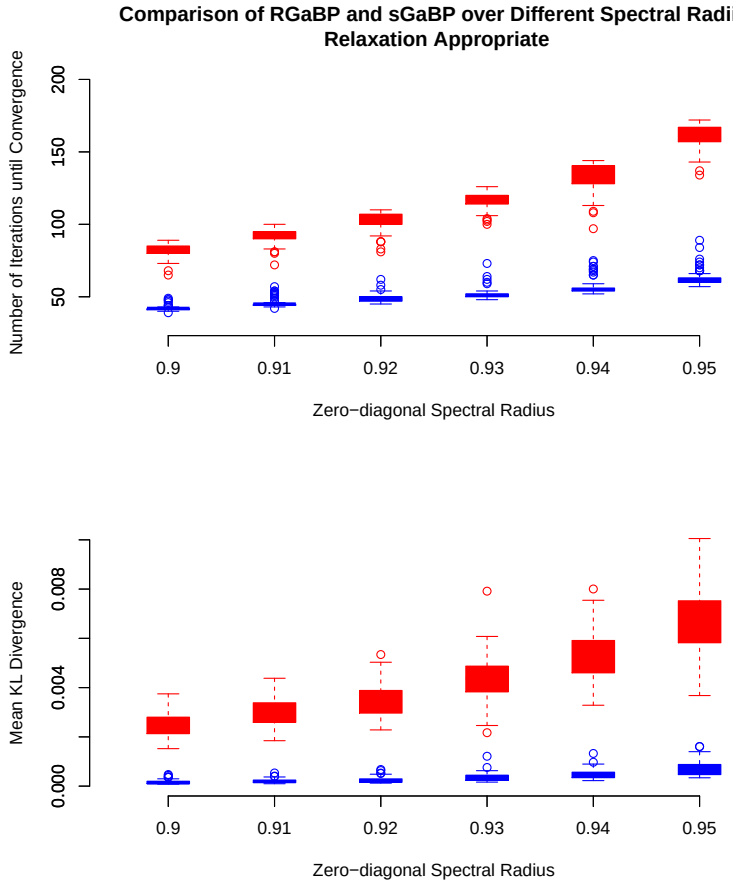


Figure 2.5: This is similar to Figures 2.3 and 2.4, however comparisons were made over different zero-diagonal spectral radii. In contrast to the previous figures the smallest eigenvalue was used to regulate the zero-diagonal spectral radius. In these simulations the optimal relaxation factor is greater than one and this corresponds to negative λ in the case of sGaBP (blue). sGaBP converged faster and provided more accurate posterior marginals.

largest or the smallest eigenvalue of the precision matrix. In our simulations, we found that optimal convergence occurs using relaxation factors when the zero-diagonal spectral radius is determined by the smallest eigenvalue, otherwise damping is preferable. This indicates that relaxation can only be applied when the spectral radius is less than one, because if the zero-diagonal spectral radius is at least one and is determined by the smallest eigenvalue, the (standardised) precision matrix will either be singular or negative definite. Figure 2.5 was constructed by considering zero-diagonal spectral radii less than one that were determined by the smallest eigenvalue of the precision matrix. Each application of optimal sGaBP and optimal RGaBP involved the use of relaxation factors. In terms of performance, we can make similar observations to those made about Figure 2.3 and Figure 2.4. In these simulations, optimal

sGaBP outperforms optimal RGaBP, both in terms of convergence speed and KL divergences, with the performance of sGaBP relative to RGaBP improving as the zero-diagonal spectral radius approaches one. One can argue that the comparisons made in Figure 2.5 are more relevant than the others made in this section, since, as the name suggests, the focus of RGaBP is on relaxation factors.

Another method proposed in the literature to improve on the convergence behaviour of GaBP is based on the principle of message damping (Malioutov *et al.*, 2006). As mentioned by Malioutov *et al.* (2006), we found in our simulations that the convergence/divergence of the precision components is independent of the degree of damping applied. Furthermore, when the precisions do converge, we found that the degree of damping does not influence the actual converged posterior precisions. We also observed that RGaBP tends to outperform the message-damping approach, based on optimal comparisons, and therefore we did not include this in our empirical comparisons.

2.5.2 Compressed Inner-loop Convergence Fix

The convergence fix (CFGaBP) method has been proposed in the literature as a method for solving arbitrary symmetric positive definite linear systems with GaBP (Johnson *et al.*, 2009). The basic idea is to solve systems of the form $(\mathbf{S}+\mathbf{\Gamma})\boldsymbol{\mu}^{(n+1)} = \boldsymbol{b} + \mathbf{\Gamma}\boldsymbol{\mu}^{(n)}$ using ordinary GaBP. Johnson *et al.* (2009) show that, if $\mathbf{S}+\mathbf{\Gamma}$ is walk-summable, CFGaBP will converge and provide the correct solution to the system $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$. We restrict our focus to the case where $\mathbf{\Gamma} = \lambda\mathbf{I}$. Johnson *et al.* (2009) refer briefly to a compressed inner-loop version of CFGaBP, where each application of GaBP is limited to one iteration. Johnson *et al.* (2009) report that compressed inner-loop CFGaBP can be more efficient than the original method, but may require damping of the potential vector. The closeness of the compressed CFGaBP variant to sGaBP depends largely on the interpretation of the description in the literature. Johnson *et al.* (2009) do not prove the convergence of compressed CFGaBP and do not consider the potential usefulness of the diagonal loadings of the precision matrix in the tuning of the posterior precisions. We could not find any reference to compressed CFGaBP in the source code provided by Bickson (2008), and formed our interpretation hereof by considering the source code provided for the original CFGaBP method, along with the description provided by Johnson *et al.* (2009). We give this interpretation in Algorithm 6. We now compare our interpretation of compressed CFGaBP with sGaBP.

The visual summaries of the number of iterations required for convergence and the mean KL divergences are given in Figure 2.6 and Figure 2.7 respectively. In the simulations, sGaBP outperformed CFGaBP in terms of convergence speed. Both methods were relatively stable in terms of the number of itera-

Algorithm 6 Compressed Inner-loop Convergence Fix GaBP

1. Provide $\mathbf{S} : p \times p$ (after preconditioning), $\boldsymbol{b} : p \times 1$ (after preconditioning), λ, m, ϵ and s as inputs to the algorithm. Here we wish to solve $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$, where $\mathbf{S}$ is positive definite and symmetric. The parameters λ, m, ϵ and s denote the degree of diagonal loading, the maximum number of iterations allowed, the tolerance used to define convergence and the damping factor respectively.

2. Initialise $\mathbf{Q}^{(0)} : p \times p = \text{diag}(1, 1, \ldots, 1)$, $\mathbf{V}^{(0)} : p \times p = \text{diag}(b_1, b_2, \ldots, b_p)$ and $\boldsymbol{\mu}^{(-1)} : p \times 1 = \mathbf{0}$.

3. Set Err = Inf and $n = 0$.

4. While Err $> \epsilon$

   a) Compute $q_i^{(n)} = 1 + \lambda + \sum_{j \in \mathcal{N}_i} Q_{ji}^{(n)}$ and $z_i^{(n)} = V_{ii} + \sum_{j \in \mathcal{N}_i} V_{ji}^{(n)}$ for $i = 1, 2, \ldots, p$.

   b) Set $\mu_i^{(n)} = \mu_i^{(n-1)} + \frac{z_i^{(n)}}{q_i^{(n)}}$ for $i = 1, 2, \ldots, p$.

   c) For all i and all $j \in \mathcal{N}_i$, set $Q_{ij}^{(n+1)} = \frac{-S_{ij}^2}{q_i^{(n)} - Q_{ji}^{(n)}}$ and $V_{ij}^{(n+1)} = \frac{Q_{ij}^{(n+1)}}{S_{ij}}(z_i^{(n)} - V_{ji}^{(n)})$.

   d) Set $e_i^{(n+1)} = b_i - \sum_j S_{ij}\mu_j^{(n)}$, Err $= \sqrt{\frac{\sum_k (\mu_k^{(n)} - \mu_k^{(n-1)})^2}{\sum_k (\mu_k^{(n)})^2}}$, $V_{ii}^{(n+1)} = s \times V_{ii}^{(n)} + (1 - s)e_i^{(n+1)}$ and increment n.

   e) If $n = m$, break.

5. End.

tions required for convergence. The performance of CFGaBP in terms of KL divergences was poor relative to the performance of sGaBP. In our simulations, we found that the degree of diagonal loadings required by CFGaBP to converge optimally was substantially higher than the tuning parameter required by optimal sGaBP. These simulations provide empirical evidence that sGaBP should be used instead of our interpretation of CFGaBP, both in terms of convergence speed and accuracy of the posterior distributions.

2.5.3 Conjugate Gradient

One of the attractive properties of GaBP as a solver of large and sparse systems of linear equations lies in distributed computing. In general, BP algorithms are well suited to distributed implementation under synchronous message schedul-

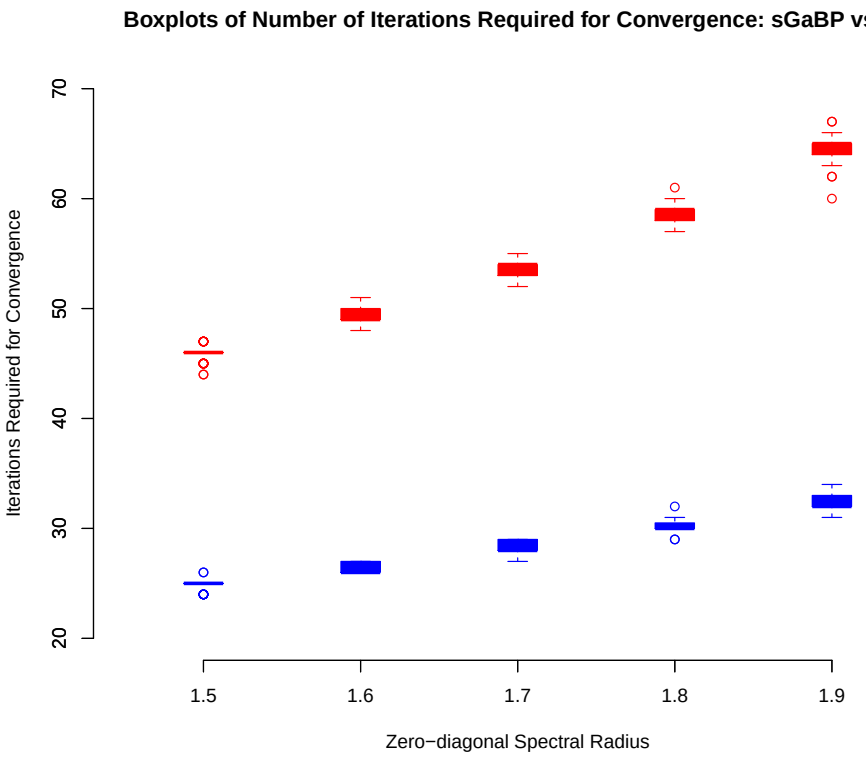


Figure 2.6: Illustration of the iterations required for convergence of optimal sGaBP (blue) and optimal CFGaBP (red). Precision matrices were generated to be of dimension 600×600. Both methods are relatively stable, although sGaBP converged faster in the simulations. The relative performance of sGaBP seems to improve with growth in the zero-diagonal spectral radius.

ing, since no communication is required between nodes not linked in the graph. Like GaBP, the CG method is a solver of linear systems and can be applied in distributed settings. A description of the CG solver can be found in Shewchuk (1994). Unlike GaBP, CG is guaranteed to converge for all symmetric and positive definite linear systems. Furthermore, CG is guaranteed to converge in at most p iterations, where p is the number of variables in the system. This causes sGaBP to compare unfavourably with CG in small linear systems, and hence our focus is on systems with a large number of variables. In practice, the CG method converges much faster than p iterations, and the convergence becomes faster for linear systems with a smaller condition number. One of the contributions of this dissertation is a message-passing scheme that guarantees convergence, and therefore we include a comparison with the CG method. The main advantage of the CG method is that it does not require any regularisation, while sGaBP provides approximate precisions.

We now compare CG with optimal sGaBP in linear systems with 700 vari-

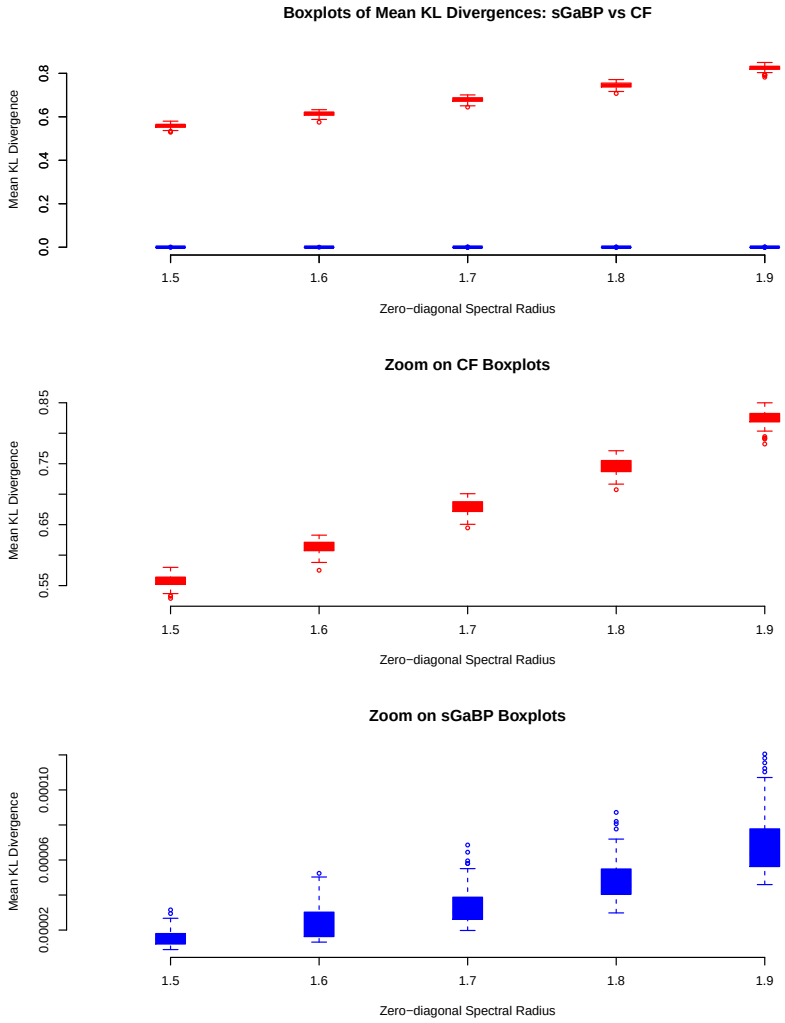


Figure 2.7: Illustration of the accuracy of the posterior distributions of optimal sGaBP (blue) and optimal CFGaBP (red) to the true marginals. In the simulations, sGaBP provided more accurate approximations. The poor performance of CFGaBP is due to the high values of the diagonal loadings it requires to converge optimally.

ables. The results are given in Figure 2.8. In these simulations, we see that both methods are quite stable and very comparable, although CG has a small advantage in the simulations involving larger zero-diagonal spectral radii. The bottom plot of Figure 2.8 shows the mean KL divergences obtained for sGaBP. We see that these divergences are small, and therefore the posterior precisions can be useful as approximations of the true marginal precisions.

There are strategies that can be used to accelerate sGaBP. One approach would be to consider asynchronous message passing. The main drawback of this strategy is the loss of distributed applicability. Another approach is to

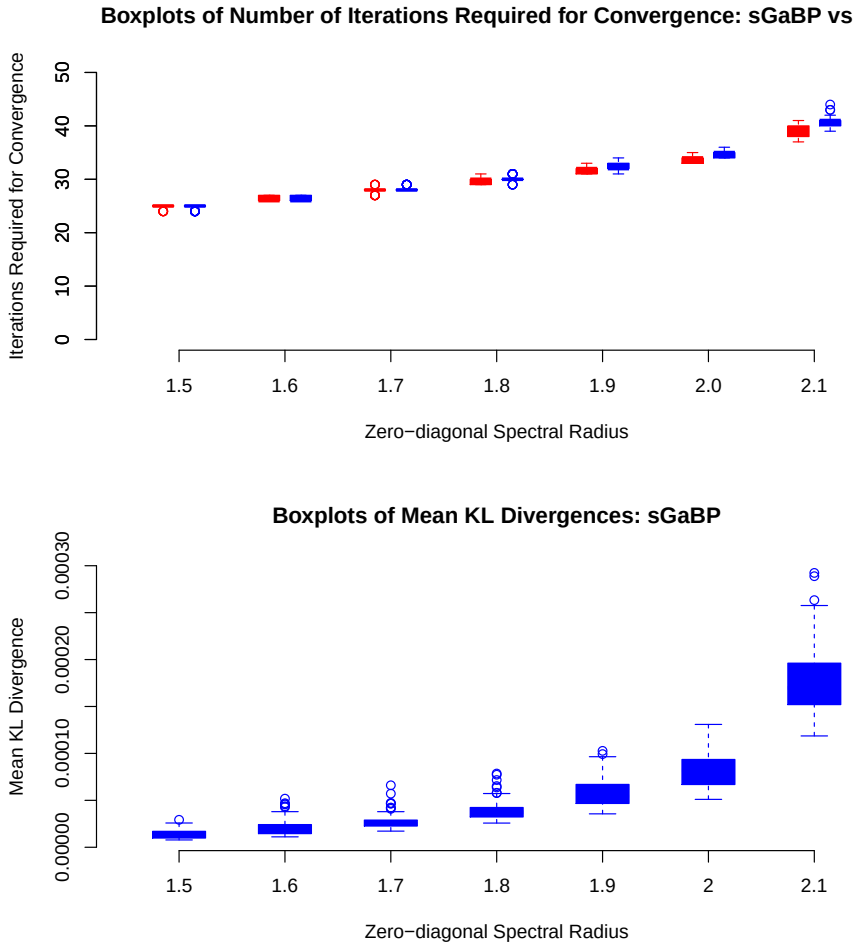


Figure 2.8: Comparison of the conjugate gradient solver with sGaBP (blue) for 700×700 precision matrices. The CG (red) method does not give approximations of the marginal precisions, but we do include the mean KL divergences for sGaBP. In our simulations, these methods are comparable in terms of the number of iterations required for convergence. The boxplots corresponding to each method are reasonably stable. The CG boxplots have a slight advantage for larger zero-diagonal spectral radii. Note that for each zero-diagonal spectral radius the boxplots corresponding to sGaBP and CG were plotted adjacent to each other.

use multiple tuning parameters, that is one tuning parameter for each node. This will not only improve convergence speed, but could also be used to obtain (even) more accurate approximations of the marginal precisions. The disadvantage is that the complexity of deciding on the level of tuning is amplified. Another interesting strategy is to increase the dimension of nodes, that is, assigning more than one variable to each node. The difficulty here is deciding on which variables to cluster together in nodes. Also, communication between higher-dimensional nodes is computationally more expensive. In certain sit-

uations we found that the GDH can improve on optimal sGaBP in terms of convergence speed. A possible explanation for this is that GDH facilitates a more flexible regularisation scheme in the sense that it adjusts λ dynamically. It is also possible to extend GDH to allow for multiple tuning parameters, which (hopefully) will accelerate convergence. The main problem surrounding GDH is the specification of the step size.

There are strategies to accelerate CG as well, with the the most prominent being that of preconditioning. Consider solving the system $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$. The idea behind preconditioning is to select an invertible matrix $\mathbf{P}$, solve $\mathbf{PSP}'\tilde{\boldsymbol{\mu}} = \mathbf{P}\boldsymbol{b}$ and transform back to the original system using $\boldsymbol{\mu} = \mathbf{P}'\tilde{\boldsymbol{\mu}}$. The matrix $\mathbf{P}$ should be selected such that the condition number of $\mathbf{PSP}'$ is smaller than that of $\mathbf{S}$, and the computational cost of computing $\mathbf{PSP}'$ is low. We note that $\mathbf{PSP}'$ is always symmetric (regardless of the choice of $\mathbf{P}$) and positive definite (since $\mathbf{P}$ is specified to be invertible). Here we wish to emphasise that sGaBP can also benefit substantially from this type of preconditioning, even more so because it makes the selection of tuning parameters easier. The major loss is in terms of the accuracy of the posterior precisions as approximations of the marginal precisions. Setting $\tilde{\mathbf{S}} = \mathbf{PSP}'$ we see that $\mathbf{S}^{-1} = \mathbf{P}'\tilde{\mathbf{S}}^{-1}\mathbf{P}$ and, depending on the choice of $\mathbf{P}$, knowing the diagonal entries of $\tilde{\mathbf{S}}^{-1}$ is not sufficient for knowing the diagonals of $\mathbf{S}^{-1}$. The consequence is that we cannot (directly) transform the posterior precisions to their original scale without knowing the off-diagonal entries of $\tilde{\mathbf{S}}^{-1}$ (this is not given by sGaBP). Finding a method to sensibly transform the approximate precisions back to the original scale will be very rewarding.

2.5.4 Further Empirical Considerations

For the purpose of this chapter, empirical convergence is defined to occur at the number of iterations required to reach an error threshold of 10^{-10}. It is possible that the comparison of the different methods may differ for different selections of the error threshold. To investigate this concern, we simulated two 1000×1000 precision matrices and corresponding 1000×1 potential vectors to compare traces of the log convergence error as a function of the iteration number for RGaBP, sGaBP and CG. The results are shown in Figure 2.9. The precision matrices corresponding to the top and bottom graph of Figure 2.9 were simulated to have zero-diagonal spectral radii equal to 1.3 and 1.4 respectively. We note that the hyper-parameters for RGaBP and sGaBP were initialised to yield convergence to an error threshold of 10^{-10} in as few iterations as possible (according to a line search in increments of 0.01).

The graphs in Figure 2.9 exhibit similar patterns. For lower iteration numbers, we see that the convergence error supplied by the different methods are very similar. For higher iteration numbers, we see that the convergence error pro-

vided by RGaBP starts to increase relative to those of the other methods. The sGaBP and CG methods are very comparable across all iterations, although the CG method tends to provide smaller convergence errors. The graphs in Figure 2.9 seem to agree with the observations made based on the 10^{-10} error threshold, although it indicates that the difference between the methods will be more substantial for smaller error thresholds.

It is possible to measure inference quality by computing the KL divergence of the posterior distribution to its corresponding exact marginal (this is the opposite direction used in the simulations). We note that changing the order of the KL divergence does not effect the conclusions made in the empirical section.

2.6 Conclusion

In this chapter, we applied the principle of node regularisation to a pairwise MG constructed from a multivariate Gaussian distribution in canonical form. We proved convergence of this algorithm, given sufficient regularisation, and showed that the converged posterior means equal the exact marginal means. In an empirical study, sGaBP compared favourably with other GaBP variants, both in terms of convergence speed and accuracy of the converged posterior precisions as approximations for the marginal precisions. The remainder of this dissertation is concerned with sGaBP where nodes are allowed to be of any size. In the next chapter, we conduct a UWT (unwrapped tree) analysis of higher-dimensional ordinary GaBP. We derive a new sufficient condition for the convergence of GaBP that depends on how variables are assigned to nodes. We also highlight a certain type of preconditioning done automatically by GaBP. This preconditioning can have a significant effect on the performance of other iterative solvers of linear systems, such as the CG method.

Some of the material covered in Chapter 3 is used in Chapter 4 to derive asymptotic expressions for the precision components of sGaBP. These asymptotic expressions are used to prove the convergence of higher-dimensional sGaBP.

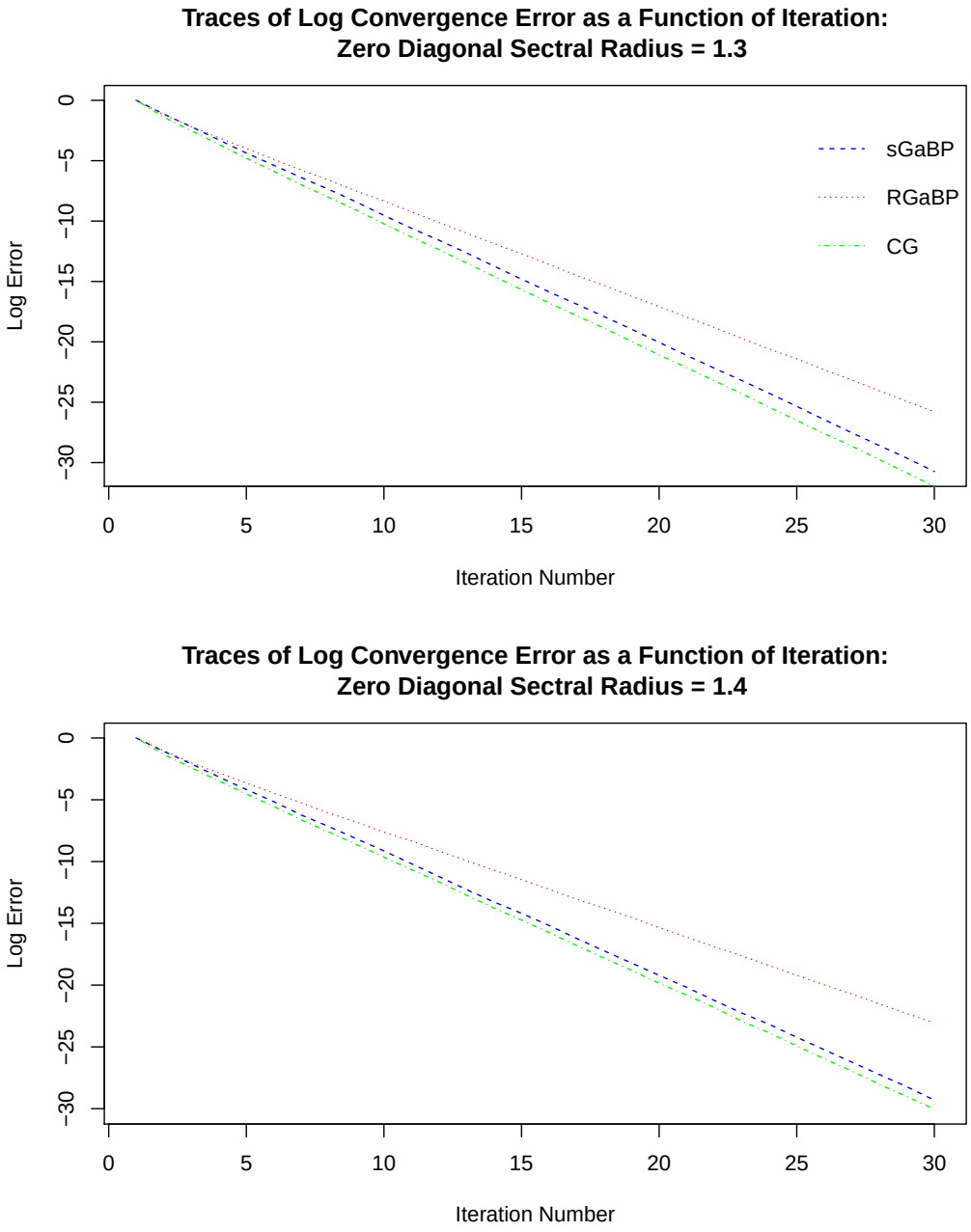


Figure 2.9: Comparison of the convergence error as a function of the iteration number for RGaBP, sGaBP and CG for two simulated data structures.

Chapter 3

Gaussian Belief Propagation with Nodes of Arbitrary Size

This chapter is concerned with deriving some new properties of GaBP where nodes are allowed to be of any dimensionality. We extend the unwrapped tree (UWT) analysis conducted by Weiss and Freeman (2001) on univariate nodes to the multivariate case. This UWT analysis is used to derive certain scaling attributes of the GaBP algorithm, which allows the interpretation of this algorithm as an automatic preconditioner, where the preconditioner is based on how variables are assigned to nodes. We develop a new, sufficient condition for the convergence of GaBP with multivariate nodes that goes beyond the walk-summability of the precision matrix. Based on these convergence conditions, we derive an upper bound for the number of iterations required for convergence of the GaBP algorithm. We also include an empirical section to illustrate and test the theoretical work covered in this chapter.

Although it may be unclear, at first, how this chapter relates to the rest of this dissertation, we note that we are building up towards Chapter 4, where we prove convergence of sGaBP for any clustering of the nodes (in contrast to Chapter 2, where nodes were constrained to be one-dimensional). In particular, we establish some results (see for instance Equation (3.14)) that will enable us to study multivariate sGaBP asymptotically, as $\lambda \to \infty$.

The idea behind UWTs is to convert the MG to a more convenient topology. This new topology allows us to formulate analytical expressions for the posterior precisions and posterior means after each iteration. These analytical expressions are used to develop sufficient conditions for the convergence of the GaBP algorithm.

3.1 Preliminaries

We provide a brief recap of the notation used in Algorithm 1 of Chapter 1. We assume a multivariate Gaussian vector $\mathbf{X} : k \times 1$ with precision matrix $\mathbf{S}$ and potential vector $\boldsymbol{b}$. The variables of $\mathbf{X}$ are divided into p clusters, which we label $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_p$. These clusters are non-overlapping and their union covers all the variables.

Suppose that $\mathbf{X}_i$ is the subvector of $\mathbf{X}$ corresponding to the variables in $\mathcal{C}_i$. In this chapter, the MG is used to represent the conditional independencies of $\mathbf{X}_i : i = 1, 2, \ldots, p$. Consider the MG given in the left panel of Figure 3.1. Let us assume this graph was drawn for the random vector $\mathbf{X} = (X_1, X_2, \ldots, X_8)'$, with clusters $\mathcal{C}_1 = \{1, 2\}$, $\mathcal{C}_2 = \{3, 4\}$, $\mathcal{C}_3 = \{5, 6\}$ and $\mathcal{C}_4 = \{7, 8\}$. The MG illustrates that $\mathbf{X}_1 = (X_1, X_2)'$ and $\mathbf{X}_4 = (X_7, X_8)'$ are conditionally independent, given the other variables.

Note that in the above example we have $\mathbf{X} = (\mathbf{X}_1', \mathbf{X}_2', \mathbf{X}_3', \mathbf{X}_4')'$. This does not hold for general clustering. Consider, for example, keeping $\mathcal{C}_2$ and $\mathcal{C}_3$ the same, but changing $\mathcal{C}_1$ and $\mathcal{C}_4$ to $\{1, 8\}$ and $\{2, 7\}$ respectively. We can easily ensure that $\mathbf{X} = (\mathbf{X}_1', \mathbf{X}_2', \mathbf{X}_3', \mathbf{X}_4')'$ by reordering $\mathbf{X}$ to $(X_1, X_8, X_3, \ldots, X_6, X_2, X_7)'$. We assume, without loss of generality, that $\mathbf{X} = (\mathbf{X}_1', \mathbf{X}_2', \ldots, \mathbf{X}_p')'$ for the clusters $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_p$.

3.1.1 Notes on the Perron-Frobenius Theorem

In this section we discuss results from the Perron-Frobenius theorem. These results play an important role in the theoretical considerations of Chapter 3 and Chapter 4. We start with a few definitions.

Definition 6 *A matrix $\mathbf{A} : k \times k$ is defined to be reducible if there exists a permutation matrix $\mathbf{P} : k \times k$ such that $\mathbf{PAP}' = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{0} & \mathbf{B}_{22} \end{bmatrix}$.*

We note that if $\mathbf{A}$ is symmetric, then $\mathbf{A}$ is reducible if there is a permutation matrix $\mathbf{P}$ such that $\mathbf{PAP}' = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{22} \end{bmatrix}$.

Definition 7 *A matrix $\mathbf{A} : k \times k$ that is not reducible is called irreducible.*

Definition 8 *Consider a symmetric real matrix $\mathbf{A} : k \times k = [a_{ij}]$. Let $\mathcal{G}(\mathbf{A})$ be the undirected graph of $\mathbf{A}$ obtained by creating a node for each variable in $\mathbf{A}$, and where there is an edge between the node representing variable i and the node representing variable j if and only if $a_{ij} \neq 0$.*

If a real symmetric matrix $\mathbf{A}$ is reducible then its graph $\mathcal{G}(\mathbf{A})$ will consist of two unconnected subgraphs $\mathcal{G}(\mathbf{B}_{11})$ and $\mathcal{G}(\mathbf{B}_{22})$. If either $\mathbf{B}_{11}$ or $\mathbf{B}_{22}$ are still

reducible we can split these graphs into more unconnected subgraphs. This process can be continued until we are left with a specified number of mutually unconnected irreducible subgraphs and a specified number of subgraphs with only one node.

When faced with applying GaBP to a reducible precision matrix we can, without loss of any information, perform GaBP separately on the irreducible subgraphs of the precision matrix.

Definition 9 (Vector and matrix inequalities) *Consider the vectors* $\mathbf{x} = (x_1, x_2, \ldots, x_k)'$ *and* $\mathbf{y} = (y_1, y_2, \ldots, y_k)'$. *By* $\mathbf{x} > \mathbf{y}$ *we mean that* $x_i > y_i$ *for* $i = 1, 2, \ldots, k$. *Analogous definitions apply to other inequality signs and matrices.*

We are now ready to formulate the Perron-Frobenius theorem and the Collatz-Wielandt formula.

Theorem 4 (Perron-Frobenius theorem) *Consider a non-negative and irreducible real matrix* $\mathbf{A} : k \times k$. *There exists a* $\mathbf{v} > \mathbf{0}$ *such that* $\mathbf{A}\mathbf{v} = \rho(\mathbf{A})\mathbf{v}$.

Theorem 5 (Collatz-Wielandt formula) *Consider a non-negative and irreducible real matrix* $\mathbf{A} : k \times k = [a_{ij}]$. *Define:*

$$h_{\mathbf{A}}(\mathbf{x}) = \min_{i:x_i \neq 0} \left\{ \frac{\sum_{j=1}^{k} a_{ij} x_j}{x_i} \right\}, \tag{3.1}$$

where $\mathbf{x} \neq \mathbf{0}$ *and* $\mathbf{x} \in \Re^k$. *Define* $\mathcal{U} = \{\mathbf{x} \in \Re^k : \mathbf{x} \geq 0 \text{ and } \mathbf{x} \neq \mathbf{0}\}$. *The Collatz-Wielandt formula states that:*

$$\rho(\mathbf{A}) = \max_{\mathbf{x} \in \mathcal{U}} \{h_{\mathbf{A}}(\mathbf{x})\}. \tag{3.2}$$

Theorem 6 *Consider any real matrix* $\mathbf{A} : k \times k$, *then* $\rho(\mathbf{A}) \leq \rho(|\mathbf{A}|)$.

Proof.

Consider the case where $\mathbf{A}$ is irreducible (this implies that $|\mathbf{A}|$ is irreducible) and let λ and $\mathbf{x} \neq \mathbf{0}$ be any eigenpair (possibly complex) of $\mathbf{A}$. Since $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ we see that:

$$|\lambda x_i| = |\lambda||x_i| = |\sum_{j=1}^{k} a_{ij} x_j| \leq \sum_{j=1}^{k} |a_{ij}||x_j|. \tag{3.3}$$

For any $i : |x_i| \neq 0$ we have:

$$|\lambda| \leq \frac{\sum_{j=1}^{k} |a_{ij}||x_j|}{|x_i|}, \tag{3.4}$$

and therefore $|\lambda| \leq h_{|\mathbf{A}|}(|\mathbf{x}|)$. Since $|\mathbf{x}| \in \mathcal{U}$ we obtain:

$$|\lambda| \leq h_{|\mathbf{A}|}(|\mathbf{x}|) \leq \rho(|\mathbf{A}|). \tag{3.5}$$

In particular, Equation (3.5) holds for all eigenvalues λ of $\mathbf{A}$ and therefore $\rho(\mathbf{A}) \leq \rho(|\mathbf{A}|)$. For the case where $\mathbf{A}$ is not reducible we note that we can find a permutation matrix $\mathbf{P}$ such that:

$$\mathbf{PAP}' = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} & \dots & \mathbf{B}_{1;(t-2)} & \mathbf{B}_{1;(t-1)} & \mathbf{B}_{1t} \\ \mathbf{0} & \mathbf{B}_{22} & \mathbf{B}_{23} & \dots & \mathbf{B}_{2;(t-2)} & \mathbf{B}_{2;(t-1)} & \mathbf{B}_{2t} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \dots & \mathbf{B}_{3;(t-2)} & \mathbf{B}_{3;(t-1)} & \mathbf{B}_{2t} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{B}_{(t-1);(t-1)} & \mathbf{B}_{(t-1);t} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{B}_{tt} \end{bmatrix},$$

where the matrices $\mathbf{B}_{ii} : i = 1, 2, \dots, t$ are either irreducible or 1×1 matrices and $\mathbf{PAP}'$ is block upper triangular. We can apply Equation (3.5) to the matrices $\mathbf{B}_{ii}$ (the inequality follows trivially for the 1×1 case) to obtain:

$$\rho(\mathbf{B}_{ii}) \leq \rho(|\mathbf{B}_{ii}|), \tag{3.6}$$

for $i = 1, 2, \dots, t$. Since $\rho(\mathbf{A}) = \rho(\mathbf{PAP}') = \max_i\{\rho(\mathbf{B}_{ii})\}$ and because $\rho(|\mathbf{A}|) = \rho(\mathbf{P}|\mathbf{A}|\mathbf{P}') = \max_i\{\rho(|\mathbf{B}_{ii})|\}$ we have:

$$\rho(\mathbf{A}) \leq \rho(|\mathbf{A}|). \tag{3.7}$$

Having discussed the preliminary concepts necessary to understand the theoretical discussions of this chapter, we turn to a discussion of the concept of a UWT in the next section.

3.2 Constructing UWTs

The idea behind a UWT is to represent the computations of GaBP as inference on a tree-structured multivariate Gaussian. In this section, we discuss the construction of these trees by considering their topology, precision matrix and potential vector. Consider the cyclical MG in the left panel of Figure 3.1. Each node of a cyclical MG has its own UWT, and we denote the UWT for node i by $\mathcal{T}_i^{(n)}$, where n is the number of iterations performed by synchronous GaBP. In this section, we illustrate the construction of the UWTs using the running example given in the previous section. The precision matrix and potential vector of this MG are

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{S}_{23} & \mathbf{S}_{24} \\ \mathbf{S}_{31} & \mathbf{S}_{32} & \mathbf{S}_{33} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{S}_{44} \end{bmatrix} \tag{3.8}$$

and

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_3 \\ \boldsymbol{b}_4 \end{bmatrix} \tag{3.9}$$

respectively. Recall that $\mathbf{S}_{ij}$ is the submatrix of $\mathbf{S}$ corresponding to the variables of cluster $\mathcal{C}_i$ (for the rows) and cluster $\mathcal{C}_j$ (for the columns). A similar interpretation holds for $\boldsymbol{b}_i$. The form of the precision matrix and potential vector given in Equation (3.8) and (3.9) respectively is a consequence of the reordering assumption given in the previous section.

3.2.1 Topology of a UWT

Under the assumption of synchronous message passing, $\mathcal{T}_i^{(n)}$ can be decomposed into n different layers according to a tree topology, as illustrated in the middle panel of Figure 3.1. Each node in $\mathcal{T}_i^{(n)}$ is associated with a specific cluster, and we keep track of this reference. We discuss the process of creating the UWT topology for node 1 and $n = 4$ of our running example. To avoid confusion, we refer to the nodes of the original MG (left side of Figure 3.1) as clusters. In our discussion, a node can have only one parent.

1. The first layer consists of a single node, known as the root node, and this node refers to the cluster for which the UWT is constructed (cluster 1 in this case).

2. To construct the second layer we determine the neighbours of cluster 1 in the original MG. For our running example, these are cluster 2 and cluster 3. Layer two consists of two mutually unconnected nodes, both connected to the root node, one with a reference to cluster 2 and the other to cluster 3.

3. Each node in layer 2 will act as a parent node for certain nodes in layer 3.

    - Let us start with the node in the second layer of the UWT corresponding to cluster 2 (see the red node). The neighbours of cluster 2 are the clusters 1, 3 and 4. We exclude cluster 1, since the parent of the red node refers to this cluster. We draw two additional mutually unconnected nodes in layer 3 of the UWT, with one referring to cluster 3 and the other to cluster 4, with the red node as the parent.
    - Consider the node in the second layer corresponding to cluster 3 (see the blue node). The neighbours of cluster 3 are the clusters 1 and 2. We exclude cluster 1, since the parent of the blue node

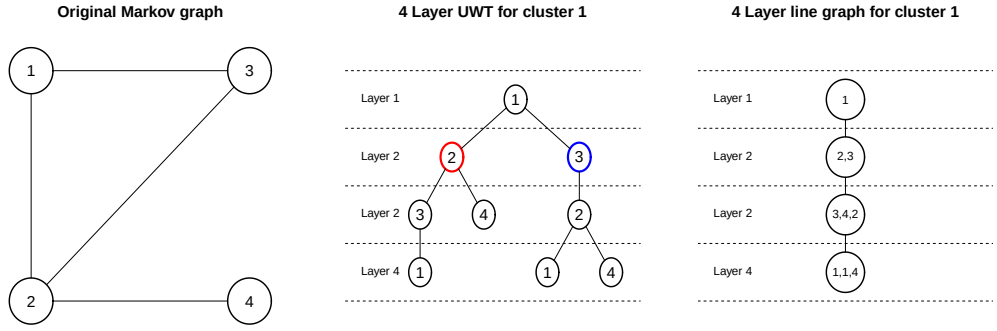


Figure 3.1: Different graph representations of a MG. The original MG is displayed on the left, the unwrapped tree for cluster 1 is displayed in the middle, while the line graph representation for the same cluster is displayed on the right. Each node (in all three graphs) contains references to certain clusters.

refers to this cluster. We draw one additional node in layer 3 with a reference to cluster 2.

4. Each node in layer 3 will act as a parent node for certain nodes in layer 4. The process of determining the children are similar to the discussion in point 3. For a particular node in layer 3, we determine the neighbourhood of the cluster it refers to and exclude from this set the cluster of its parent node. Mutually unconnected nodes are drawn in layer 4, each corresponding to one of the clusters in this set. This procedure is repeated for all nodes in layer 3.

We can continue this process to generate $\mathcal{T}_1^{(n)}$ for any n. A further useful topology is obtained by clustering together all nodes in different layers of the UWT. For the purpose of our discussion, we cluster together nodes from left to right in a given layer. This method of clustering yields a line topology (right panel of Figure 3.1). The line topology conversion for a node is denoted by $\mathcal{L}_i^{(n)}$.

3.2.2 Specifying the Precision and Potential

We can focus, without loss of generality, on the UWT associated with node 1. The UWT and line topology for node 1 are denoted by $\mathcal{T}_n$ and $\mathcal{L}_n$ respectively.

A movement along a UWT is defined to be first by layer, starting at layer 1, and within each layer from left to right. Unless explicitly stated otherwise, we assume that the order of the nodes in the precision matrix and potential vector, associated with $\mathcal{T}_n$, is according to the movement along $\mathcal{T}_n$.

We interpret $\mathcal{T}_n$ as a Gaussian MG. Associate with $\mathcal{T}_n$ the precision matrix $\mathbf{T}_n : m_n \times m_n$ and potential vector $\mathbf{t}_n : m_n \times 1$. If a node in $\mathcal{T}_n$ has a reference to cluster t, it receives a precision matrix $\mathbf{S}_{tt}$ and potential vector $\boldsymbol{b}_t$. If a node (reference j) has a link to a node (reference t) in $\mathcal{T}_n$, then this link receives $\mathbf{S}_{jt}$, otherwise the link receives $\mathbf{0} : d_j \times d_t$. Recall that $|\mathcal{C}_j| = d_j$. For our running example, we have:

$$\mathbf{T}_4 = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{S}_{24} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{31} & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{31} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{44} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{22} & \mathbf{0} & \mathbf{S}_{21} & \mathbf{S}_{24} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{11} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{12} & \mathbf{0} & \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{44} \end{bmatrix}$$

$$\mathbf{t}_4 = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_3 \\ \boldsymbol{b}_3 \\ \boldsymbol{b}_4 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_1 \\ \boldsymbol{b}_1 \\ \boldsymbol{b}_4 \end{bmatrix}.$$

The remainder of this section is dedicated to relating $\mathbf{T}_n$ and $\mathbf{t}_n$ to $\mathbf{S}$ and $\boldsymbol{b}$ respectively. These relations are important for the remainder of this chapter. We now introduce certain special matrices.

Notation 4 *Row-extractor matrix of cluster t. Define*

$$\mathbf{F}_t : d_t \times k = \begin{bmatrix} \mathbf{0} : d_t \times d_1 & \dots & \mathbf{0} : d_t \times d_{t-1} & \mathbf{I}_{d_t} & \mathbf{0} : d_t \times d_{t+1} & \dots & \mathbf{0} : d_t \times d_p \end{bmatrix}$$

Due to the reordering assumption, we see that $\mathbf{F}_t\mathbf{S}$ extracts the rows of $\mathbf{S}$ corresponding to the variables in $\mathcal{C}_t$. Also note that:

$$\mathbf{S}_{jt}\mathbf{F}_t = \begin{bmatrix} \mathbf{0} : d_j \times d_1 & \dots & \mathbf{0} : d_j \times d_{t-1} & \mathbf{S}_{jt} & \mathbf{0} : d_j \times d_{t+1} & \dots & \mathbf{0} : d_j \times d_p \end{bmatrix}.$$

Notation 5 *Row-extractor matrix. The row-extractor matrix $\mathbf{E}_n : m_n \times p$ is obtained by moving along the UWT, noting the cluster reference of each node*

and stacking the row-extractor matrix of this cluster. For our running example:

$$\mathbf{E}_4 = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{F}_3 \\ \mathbf{F}_3 \\ \mathbf{F}_4 \\ \mathbf{F}_2 \\ \mathbf{F}_1 \\ \mathbf{F}_1 \\ \mathbf{F}_4 \end{bmatrix}.$$

Using our running example, it is easy to see that:

$$\mathbf{t}_n = \mathbf{E}_n \boldsymbol{b}. \tag{3.10}$$

Notation 6 *Node extractor matrix. Consider node j of the UWT (read from left to right along the layers). For our running example, the sixth node of the UWT is the rightmost node in layer 3. Let $\mathbf{G}_n^{(j)}$ be such that $[\mathbf{G}_n^{(j)}]'\mathbf{T}_n$ extracts the rows of $\mathbf{T}_n$ corresponding to node j. For our running example:*

$$[\mathbf{G}_4^{(6)}]'\mathbf{T}_4 = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{22} & \mathbf{0} & \mathbf{S}_{21} & \mathbf{S}_{24} \end{bmatrix}.$$

Define the root extractor matrix as $\mathbf{G}_n = \mathbf{G}_n^{(1)}$.

Our next objective is to validate the formula,

$$\mathbf{T}_n\mathbf{E}_n = \mathbf{E}_n\mathbf{S} + \mathbf{L}_n, \tag{3.11}$$

where the entries of $\mathbf{L}_n$ are all zero, except possibly in the rows corresponding to the nodes in layer n of the UWT. For our running example:

$$\mathbf{T}_4\mathbf{E}_4 = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{S}_{23} & \mathbf{S}_{24} \\ \mathbf{S}_{31} & \mathbf{S}_{32} & \mathbf{S}_{33} & \mathbf{0} \\ \mathbf{S}_{31} & \mathbf{S}_{32} & \mathbf{S}_{33} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{S}_{44} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{S}_{23} & \mathbf{S}_{24} \\ \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} \\ \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{S}_{44} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{S}_{12} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{S}_{13} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

To see why Equation (3.11) is true in general, note that each node in the UWT (except for nodes in the final layer) is locally connected once to nodes with references to the neighbours of its corresponding cluster. Consider a node j,

not in the final layer, in the UWT. Suppose that this node refers to cluster i. We see that:

$$[\mathbf{G}_n^{(j)}]'\mathbf{T}_n\mathbf{E}_n = \sum_{l\in\mathcal{N}_i\cup\{i\}} \mathbf{S}_{il}\mathbf{F}_l = \mathbf{F}_i\mathbf{S}. \tag{3.12}$$

Equation (3.12) can also be validated using our running example. A node in the final layer is not connected to nodes that encompass all the neighbours of its corresponding cluster. For instance, node 7 (left-most node, final layer) is only connected to a single node with reference to cluster 3. Node 7 refers to cluster 1, which has clusters 2 and 3 as neighbours in the original MG. We see that node 7 in the UWT has a neighbouring node with reference to cluster 3, but no link to a node with a reference to cluster 2. Hence, if j is in the final layer of the UWT, then

$$[\mathbf{G}_n^{(j)}]'\mathbf{T}_n\mathbf{E}_n = \sum_{l\in\mathcal{A}\cup\{i\}} \mathbf{S}_{il}\mathbf{F}_l, \tag{3.13}$$

where $\mathcal{A} \subset \mathcal{N}_i$. Hence $[\mathbf{G}_n^{(j)}]'\mathbf{T}_n\mathbf{E}_n$ in Equation (3.13) is a sparse replicate of $\mathbf{F}_i\mathbf{S}$. These considerations validate the formula in Equation (3.11).

3.3 Analytical Formulas for the Posterior Means and Posterior Precisions

In this section we derive a method of moving from inference on the UWT to inference on the original MG. In particular, we show that

$$\mathbf{P}_1^{(n-1)} = [\mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{G}_n]^{-1} \tag{3.14}$$

$$\boldsymbol{\mu}_1^{(n-1)} = \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{E}_n\boldsymbol{b}, \tag{3.15}$$

where $\mathbf{P}_1^{(n-1)}$ and $\boldsymbol{\mu}_1^{(n-1)}$ denote the posterior precision and posterior mean of node 1 at iteration $n-1$. First, we discuss a pruning procedure on a general tree-structured precision matrix $\mathbf{T}$ and potential vector $\mathbf{t}$. We then show how this elimination procedure can be used to validate the formulas given in Equations (3.14) and (3.15).

3.3.1 Tree-pruning Procedure

Consider a general tree $\mathcal{T}$ with precision matrix $\mathbf{T}$ and potential vector $\mathbf{t}$. Suppose that $r_\mu(\mathcal{T})$ and $r_P(\mathcal{T})$ give the marginal mean and marginal precision respectively of the root node of $\mathcal{T}$. We can obtain $r_P(\mathcal{T})$ by inverting the diagonal block of $\mathbf{T}^{-1}$ corresponding to the root node. In a similar way, $r_\mu(\mathcal{T})$ is obtained by extracting the first subvector (corresponding to the root node) of $\mathbf{T}^{-1}\mathbf{t}$.

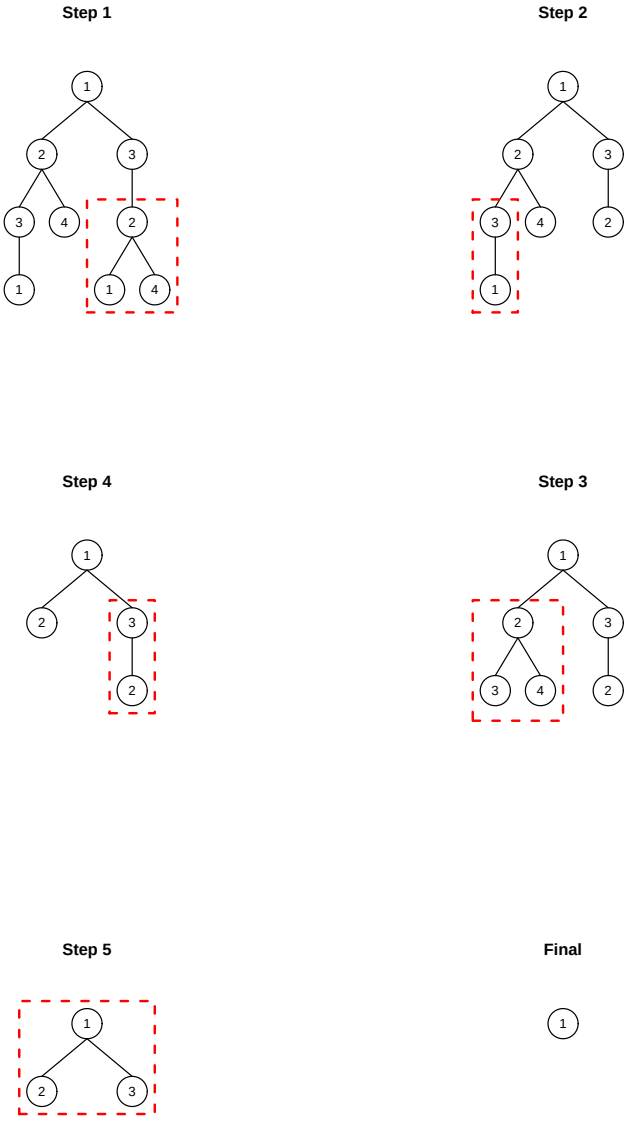


Figure 3.2: Visualisation of the process used to invert a tree-structured precision matrix/solve a linear system through node pruning, where the block of the inverse/solution of the root node is of interest.

The tree-pruning procedure iteratively prunes terminal nodes such that the marginal precision and marginal mean of the root node of the remaining (trimmed) tree remain unchanged. Let us return to our running example. The goal is to modify a tree, as in Step 1 of Figure 3.2, to one of its trimmed trees, as in Step 2, in such a way that inference on the root node remains unchanged. Let $\tilde{\mathcal{T}}$ be the trimmed tree obtained after eliminating certain terminal nodes from $\mathcal{T}$, then we must have that $r_\mu(\mathcal{T}) = r_\mu(\tilde{\mathcal{T}})$ and $r_P(\mathcal{T}) = r_P(\tilde{\mathcal{T}})$. We use the conversion from the tree in Step 1 to the tree in Step 2 as an example.

At each step of the pruning process, we select one of the nodes in the second last layer (we will call this node the bereaved parent, this would be the node corresponding to cluster 2 in the red rectangle of Step 1), the objective being to prune its children in the terminal layer. This must be done in such a way that inference at the root node remains unchanged. In order to achieve this, we need to change certain elements of the precision and potential associated with the nodes in the trimmed tree, from their corresponding values in the untrimmed tree. In the remainder of this section, we will show that it is only necessary to change the precision matrix and potential vector associated with the bereaved parent. Moreover, these adjustments can be obtained by marginalising the tree with the bereaved parent as the root node (this is the nodes in the red rectangle of Step 1).

Let $\mathbf{T}$ and $\mathbf{t}$ be the precision matrix and potential vector respectively of $\mathcal{T}$ in Step 1. We can write without loss of generality:

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} & \mathbf{0} \\ \mathbf{T}_{21} & \mathbf{T}_{22} & \mathbf{T}_{23} \\ \mathbf{0} & \mathbf{T}_{32} & \mathbf{T}_{33} \end{bmatrix}$$
$$\mathbf{t} = \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \mathbf{t}_3 \end{bmatrix}.$$

Here, $\mathbf{t}_1$, $\mathbf{t}_2$ and $\mathbf{t}_3$ are the subvectors of $\mathbf{t}$ corresponding to all nodes outside the red rectangle, node 2 within the red rectangle, and nodes 1 and 4 inside the red rectangle respectively. The precision matrix is decomposed in a similar way. In general, we associate $\mathbf{t}_3$ with the nodes to be pruned, $\mathbf{t}_2$ with the bereaved parent and $\mathbf{t}_1$ with the remaining nodes. Consider

$$\mathbf{T} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$
$$\mathbf{a} = \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \end{bmatrix}$$

where $\mathbf{A}_{11} = \begin{bmatrix}\mathbf{T}_{11} & \mathbf{T}_{12}\\ \mathbf{T}_{21} & \mathbf{T}_{22}\end{bmatrix}$, $\mathbf{A}_{12} = \begin{bmatrix}\mathbf{0}\\ \mathbf{T}_{23}\end{bmatrix}$, $\mathbf{A}_{21} = \begin{bmatrix}\mathbf{0} & \mathbf{T}_{32}\end{bmatrix}$, $\mathbf{A}_{22} = \mathbf{T}_{33}$, $\mathbf{a}_1 = \begin{bmatrix}\mathbf{t}_1\\ \mathbf{t}_2\end{bmatrix}$ and $\mathbf{a}_2 = \mathbf{t}_3$. Blockwise matrix inversion yields,

$$\mathbf{T}^{-1} = \begin{bmatrix}\mathbf{A}_{11.2}^{-1} & -\mathbf{A}_{11.2}^{-1}\mathbf{A}_{12}\mathbf{A}_{22}^{-1}\\ \dots & \dots\end{bmatrix},$$

where $\mathbf{A}_{11.2} = \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}$ and the $\dots$ represents an irrelevant part of the matrix (nodes to be pruned). Let $\tilde{\mathcal{T}}$ denote the trimmed tree. In order to preserve root inference, the precision matrix and potential vector associated with $\tilde{\mathcal{T}}$ must be,

$$\begin{aligned}\tilde{\mathbf{T}} &= \mathbf{A}_{11.2}\\ \tilde{\mathbf{t}} &= \mathbf{a}_1 - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{a}_2,\end{aligned}$$

respectively. It can be shown that:

$$\mathbf{A}_{11.2} = \begin{bmatrix}\mathbf{T}_{11} & \mathbf{T}_{12}\\ \mathbf{T}_{21} & \mathbf{T}_{22} - \mathbf{T}_{23}\mathbf{T}_{33}^{-1}\mathbf{T}_{32}\end{bmatrix} \tag{3.16}$$

$$\mathbf{a}_1 - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{a}_2 = \begin{bmatrix}\mathbf{t}_1\\ \mathbf{t}_2 - \mathbf{T}_{23}\mathbf{T}_{33}^{-1}\mathbf{t}_3\end{bmatrix}. \tag{3.17}$$

Equations (3.16) and (3.17) show that it is only necessary to adjust the precision and potential associated with the bereaved parent. Thereafter, we can prune the terminal nodes.

Let $\mathcal{M}$ be the tree containing the nodes to be pruned, with the bereaved parent as the root node. The precision and potential associated with $\mathcal{M}$ are

$$\begin{aligned}\mathbf{M} &= \begin{bmatrix}\mathbf{T}_{22} & \mathbf{T}_{23}\\ \mathbf{T}_{32} & \mathbf{T}_{33}\end{bmatrix}\\ \mathbf{m} &= \begin{bmatrix}\mathbf{t}_2\\ \mathbf{t}_3\end{bmatrix},\end{aligned}$$

respectively. Setting $\mathbf{T}_{22.3} = \mathbf{T}_{22} - \mathbf{T}_{23}\mathbf{T}_{33}^{-1}\mathbf{T}_{32}$ we see, using blockwise matrix inversion, that:

$$\begin{aligned}r_P(\mathcal{M}) &= \mathbf{T}_{22.3}\\ r_\mu(\mathcal{M}) &= \mathbf{T}_{22.3}^{-1}[\mathbf{t}_2 - \mathbf{T}_{23}\mathbf{T}_{33}^{-1}\mathbf{t}_3].\end{aligned}$$

In summary, to prune terminal nodes while preserving the marginal quantities of the root node, we need to do the following:

1. Compute $r_\mu(\mathcal{M})$ and $r_P(\mathcal{M})$.

2. To preserve the marginal quantities of the root node, the trimmed tree is obtained by eliminating the terminal nodes from the graph and only adjusting the potential and the precision of the bereaved parent to $r_P(\mathcal{M})r_\mu(\mathcal{M})$ and $r_P(\mathcal{M})$ respectively.

In the next section, we present a proof of UWT correctness, which, according to our knowledge, has not appeared in the literature before. We prove UWT correctness by showing that the computations done by GaBP can be represented by the pruning procedure described in this section.

3.3.2 UWT Correctness

By UWT correctness we mean the validity of the formulas given in (3.14) and (3.15). In order to establish this, recall that $\mathbf{z}_{ij}^{(l)} = \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{v}_{ti}^{(l)}$, $\mathbf{P}_{ij}^{(l)} = \mathbf{S}_{ii} + \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{Q}_{ti}^{(l)}$ and define

$$\boldsymbol{\mu}_{ij}^{(l)} = [\mathbf{P}_{ij}^{(l)}]^{-1}\mathbf{z}_{ij}^{(l)}.$$

Consider the following recursive expansions (recall Equations (1.15) and (1.16)):

$$\mathbf{P}_{ij}^{(l)} = \mathbf{S}_{ii} - \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{S}_{it}[\mathbf{P}_{ti}^{(l-1)}]^{-1}\mathbf{S}_{it}$$
$$\mathbf{z}_{ij}^{(l)} = \boldsymbol{b}_i - \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{S}_{it}[\mathbf{P}_{ti}^{(l-1)}]^{-1}\mathbf{z}_{ti}^{(n-1)}.$$

Suppose we have pruned $\mathcal{T}_n$ to have only l layers, where $3 \leq l \leq n$. Consider a bereaved parent, in layer $l-1$, assumed to have a reference to cluster i and its parent in layer $l-2$ with a reference to cluster j. Due to the construction of $\mathcal{T}_n$, we know that the bereaved parent will have mutually unconnected terminal nodes as children, each with a reference to a different cluster in $\mathcal{N}_i \setminus j$. Let $|\mathcal{N}_i \setminus j| = s_{ij}$, $\mathcal{N}_i \setminus j = \{i_1, i_2, \ldots, i_{s_{ij}}\}$ and suppose that $\mathcal{M}$ is the tree with the selected bereaved parent as root, connected to its children in the terminal layer. We assume that the precision and potential associated with one of these terminal nodes, with a reference to cluster i_t, are $\mathbf{P}_{i_t i}^{(n-l)}$ and $\mathbf{z}_{i_t i}^{(n-l)}$ respectively.

If $\mathbf{M}$ and $\mathbf{m}$ denote the precision and potential of $\mathcal{M}$ respectively, we have:

$$\mathbf{M} = \begin{bmatrix} \mathbf{S}_{ii} & \mathbf{S}_{ii_1} & \mathbf{S}_{ii_2} & \dots & \mathbf{S}_{ii_{s_{ij}}} \\ \mathbf{S}_{i_1 i} & \mathbf{P}_{i_1 i}^{(n-l)} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{S}_{i_2 i} & \mathbf{0} & \mathbf{P}_{i_2 i}^{(n-l)} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{S}_{i_{s_{ij}} i} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{P}_{i_{s_{ij}} i}^{(n-l)} \end{bmatrix} \tag{3.18}$$

$$\mathbf{m} = \begin{bmatrix} \boldsymbol{b}_i \\ \mathbf{z}_{i_1 i}^{(n-l)} \\ \mathbf{z}_{i_2 i}^{(n-l)} \\ \vdots \\ \mathbf{z}_{i_{s_{ij}} i}^{(n-l)} \end{bmatrix}. \tag{3.19}$$

Using blockwise matrix inversion, it is easy to see that:

$$r_P(\mathcal{M}) = \mathbf{S}_{ii} - \sum_{t=1}^{s_{ij}} \mathbf{S}_{ii_t} [\mathbf{P}_{i_t i}^{(n-l)}]^{-1} \mathbf{S}_{i_t i} = \mathbf{P}_{ij}^{(n-l+1)} \tag{3.20}$$

$$r_\mu(\mathcal{M}) = [\mathbf{P}_{ij}^{(n-l)}]^{-1} [\boldsymbol{b}_i - \sum_{t=1}^{s_{ij}} \mathbf{S}_{ii_t} [\mathbf{P}_{i_t i}^{(n-l)}]^{-1} \mathbf{z}_{i_t i}^{(n-l)}] = [\mathbf{P}_{ij}^{(n-l+1)}]^{-1} \mathbf{z}_{ij}^{(n-l+1)}. \tag{3.21}$$

To prune these terminal nodes, the tree-pruning procedure requires adjusting the precision and potential of the bereaved parent to $\mathbf{P}_{ij}^{(n-l+1)}$ and $\mathbf{z}_{ij}^{(n-l+1)}$ respectively. Once we have pruned all the terminal nodes, we see that we can apply a similar process to the new terminal layer $(l-1)$.

Let us consider the case where $l = n$, where a terminal node with reference to cluster i_1 and associated parent with reference to cluster i, is considered. At this stage, the precision and potential associated with this terminal node are $\mathbf{S}_{i_1 i_1}$ and $\boldsymbol{b}_{i_1}$ respectively. Since,

$$\begin{aligned} \mathbf{Q}_{i_1 i}^{(1)} &= -\mathbf{S}_{i_1 i} \mathbf{S}_{i_1 i_1}^{-1} \mathbf{S}_{i_1 i} \\ \mathbf{v}_{i_1 i}^{(1)} &= -\mathbf{S}_{i_1 i} \mathbf{S}_{i_1 i_1}^{-1} \boldsymbol{b}_{i_1} \end{aligned}$$

we see that $\mathbf{P}_{i_1 i}^{(0)} = \mathbf{S}_{i_1 i_1}$ and $\mathbf{z}_{i_1 i}^{(0)} = \boldsymbol{b}_{i_1}$. Hence, the type of pruning described in Equations (3.18) and (3.19) apply at all stages of pruning by induction.

Suppose we have completed the pruning such that only two layers remain. The trimmed tree will have a root node corresponding to cluster 1 and this node is connected to a terminal node, each of which has a reference to a different cluster in $\mathcal{N}_1$. Let $\mathcal{N}_1 = \{i_1, i_2, \dots, i_{s_1}\}$. Since we have followed the

pruning procedure, the precision matrix and potential vector associated with a terminal node, with reference to cluster i_t, are $\mathbf{z}_{i_t1}^{(n-2)}$ and $\mathbf{P}_{i_t1}^{(n-2)}$ respectively. The final step is to find the root marginal of the following precision matrix and potential vector:

$$\mathbf{M} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{1i_1} & \mathbf{S}_{1i_2} & \dots & \mathbf{S}_{1i_{n_1}} \\ \mathbf{S}_{i_11} & \mathbf{P}_{i_11}^{(n-2)} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{S}_{i_21} & \mathbf{0} & \mathbf{P}_{i_21}^{(n-2)} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{S}_{i_{s_1}1} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{P}_{i_{n_1}1}^{(n-2)} \end{bmatrix} \tag{3.22}$$

$$\mathbf{m} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{z}_{i_11}^{(n-2)} \\ \mathbf{z}_{i_21}^{(n-2)} \\ \vdots \\ \mathbf{z}_{i_{s_1}1}^{(n-2)} \end{bmatrix}. \tag{3.23}$$

One more application of blockwise matrix inversion reveals:

$$r_P(\mathcal{T}_n) = \mathbf{S}_{11} - \sum_{t=1}^{s_1} \mathbf{S}_{1i_t}[\mathbf{P}_{i_t1}^{(n-2)}]^{-1}\mathbf{S}_{i_t1} = \mathbf{S}_{11} + \sum_{t=1}^{s_1} \mathbf{Q}_{i_t1}^{(n-1)} = \mathbf{P}_1^{(n-1)} \tag{3.24}$$

$$\begin{aligned} r_\mu(\mathcal{T}_n) &= [\mathbf{P}_1^{(n-1)}]^{-1}[\boldsymbol{b}_1 - \sum_{t=1}^{s_1} \mathbf{S}_{1i_t}[\mathbf{P}_{i_t1}^{(n-2)}]^{-1}\mathbf{z}_{i_t1}^{(n-2)}] \\ &= [\mathbf{P}_1^{(n-1)}]^{-1}[\boldsymbol{b}_1 + \sum_{t=1}^{s_1} \mathbf{v}_{i_t1}^{(n-1)}] = \boldsymbol{\mu}_1^{(n-1)}. \end{aligned} \tag{3.25}$$

Equations (3.24) and (3.25) validate the formulas given in (3.14) and (3.15).

Let us return to our running example, where we consider moving from Step 1 to Step 2 in Figure 3.2. The tree-pruning procedure requires us to perform inference on:

$$\mathbf{M} = \begin{bmatrix} \mathbf{S}_{22} & \mathbf{S}_{21} & \mathbf{S}_{24} \\ \mathbf{S}_{12} & \mathbf{S}_{11} & \mathbf{0} \\ \mathbf{S}_{42} & \mathbf{0} & \mathbf{S}_{44} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{22} & \mathbf{S}_{21} & \mathbf{S}_{24} \\ \mathbf{S}_{12} & \mathbf{P}_{12}^{(0)} & \mathbf{0} \\ \mathbf{S}_{42} & \mathbf{0} & \mathbf{P}_{42}^{(0)} \end{bmatrix}$$

$$\mathbf{m} = \begin{bmatrix} \boldsymbol{b}_2 \\ \boldsymbol{b}_1 \\ \boldsymbol{b}_4 \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_2 \\ \mathbf{z}_{12}^{(0)} \\ \mathbf{z}_{42}^{(0)} \end{bmatrix}.$$

Equations (3.20) and (3.21) imply that $r_P(\mathcal{M}) = \mathbf{P}_{23}^{(1)}$ and $r_\mu(\mathcal{M}) = [\mathbf{P}_{23}^{(1)}]^{-1}\mathbf{z}_{23}^{(1)}$. We remove the nodes (nodes 1 and 4 in the red rectangle) and adjust the potential and precision of node 2 in the red rectangle to $r_P(\mathcal{M})r_\mu(\mathcal{M}) = \mathbf{z}_{23}^{(1)}$ and

$r_P(\mathcal{M}) = \mathbf{P}_{23}^{(1)}$ respectively. The potential vector and precision matrix of the trimmed tree are given in Equations (3.26) and (3.27) (the vector and matrix to the right of the arrows indicate the potential vector and precision matrix of the tree obtained after the next pruning step).

$$\tilde{\mathbf{t}} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_3 \\ \boldsymbol{b}_3 \\ \boldsymbol{b}_4 \\ \mathbf{z}_{23}^{(1)} \\ \boldsymbol{b}_1 \end{bmatrix} \rightarrow \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \boldsymbol{b}_3 \\ \mathbf{z}_{32}^{(1)} \\ \mathbf{z}_{42}^{(1)} \\ \mathbf{z}_{23}^{(1)} \end{bmatrix} \tag{3.26}$$

$$\tilde{\mathbf{T}} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{S}_{24} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{31} & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{31} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{44} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{P}_{23}^{(1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{11} \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{S}_{24} & \mathbf{0} \\ \mathbf{S}_{31} & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{32} \\ \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{P}_{32}^{(1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{P}_{42}^{(1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{P}_{23}^{(1)} \end{bmatrix}. \tag{3.27}$$

Note that, due to the specific sparsity pattern, we can write $\mathbf{z}_{42}^{(1)} = \mathbf{z}_{42}^{(0)} = \boldsymbol{b}_2$ and $\mathbf{P}_{42}^{(1)} = \mathbf{P}_{42}^{(0)} = \mathbf{S}_{44}$. We can repeat this process until we are left with:

$$\tilde{\mathbf{t}} = \begin{bmatrix} \boldsymbol{b}_1 \\ \mathbf{z}_{21}^{(2)} \\ \mathbf{z}_{31}^{(2)} \end{bmatrix}$$

$$\tilde{\mathbf{T}} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} \\ \mathbf{S}_{21} & \mathbf{P}_{21}^{(2)} & \mathbf{0} \\ \mathbf{S}_{31} & \mathbf{0} & \mathbf{P}_{31}^{(2)} \end{bmatrix}.$$

We then apply one more pruning step:

$$\begin{aligned} r_P(\mathcal{T}_4) &= \mathbf{S}_{11} - \mathbf{S}_{12}\mathbf{P}_{21}^{(2)}\mathbf{S}_{21} - \mathbf{S}_{13}\mathbf{P}_{31}^{(2)}\mathbf{S}_{31} = \mathbf{S}_{11} + \mathbf{Q}_{21}^{(3)} + \mathbf{Q}_{31}^{(3)} = \mathbf{P}_1^{(3)} \\ r_\mu(\mathcal{T}_4) &= [\mathbf{P}_1^{(3)}]^{-1}[\boldsymbol{b}_1 - \mathbf{S}_{12}[\mathbf{P}_{21}^{(2)}]^{-1}\mathbf{z}_{21}^{(2)} - \mathbf{S}_{13}[\mathbf{P}_{31}^{(2)}]^{-1}\mathbf{z}_{31}^{(2)}] \\ &= [\mathbf{P}_1^{(3)}]^{-1}[\boldsymbol{b}_1 + \mathbf{v}_{21}^{(2)} + \mathbf{v}_{31}^{(2)}] = \boldsymbol{\mu}_1^{(3)}. \end{aligned}$$

3.4 Automatic Preconditioning

We use the formulas (3.14) and (3.15) to consider a specific type of preconditioning that is done automatically by GaBP. Consider symmetric and positive definite matrices $\boldsymbol{\Lambda}_{ii} : d_i \times d_i$ for $i = 1, 2, \ldots, p$. Let $\boldsymbol{\Lambda} : k \times k$ be such that the

sub-matrix corresponding to the variables in $\mathcal{C}_i$ is $\mathbf{\Lambda}_{ii}$ for $i = 1, 2, \ldots, p$, and all other entries are zero. We now compare GaBP on $\mathbf{S}$ and $\boldsymbol{b}$ with GaBP on $\tilde{\mathbf{S}} = \mathbf{\Lambda}\mathbf{S}\mathbf{\Lambda}$ and $\tilde{\boldsymbol{b}} = \mathbf{\Lambda}\boldsymbol{b}$.

Suppose $\mathbf{B}_n : m_n \times m_n$ is constructed by moving along the UWT, noting the reference of each node (say i) and adding a symmetric and positive definite matrix $\mathbf{\Lambda}_{ii}$ as diagonal blocks of $\mathbf{B}_n$ (all other entries are zero). It is easy to see that $\tilde{\mathbf{T}}_n = \mathbf{B}_n\mathbf{T}_n\mathbf{B}_n$ is the precision matrix corresponding to the UWT (for node 1) constructed from $\tilde{\mathbf{S}}$. The following holds:

$$\begin{aligned}\mathbf{P}_1^{(n-1)} &= [\mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{G}_n]^{-1} \\ &= [\mathbf{G}_n'\mathbf{B}_n\tilde{\mathbf{T}}_n^{-1}\mathbf{B}_n\mathbf{G}_n]^{-1}. \end{aligned} \tag{3.28}$$

Note that $\mathbf{G}_n'\mathbf{B}_n$ equals the first d_1 rows of $\mathbf{B}_n$; therefore, since $\mathbf{B}_n$ is block-diagonal, $\mathbf{G}_n'\mathbf{B}_n = \mathbf{\Lambda}_{11}\mathbf{G}_n'$. Equation (3.28) becomes

$$\mathbf{P}_1^{(n-1)} = \mathbf{\Lambda}_{11}^{-1}\tilde{\mathbf{P}}_1^{(n-1)}\mathbf{\Lambda}_{11}^{-1}, \tag{3.29}$$

where $\tilde{\mathbf{P}}_1^{(n-1)} = [\mathbf{G}_n'\tilde{\mathbf{T}}_n^{-1}\mathbf{G}_n]^{-1}$. Consider

$$\begin{aligned}\boldsymbol{\mu}_1^{(n-1)} &= \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{E}_n\boldsymbol{b} \\ &= \mathbf{G}_n'\mathbf{B}_n\tilde{\mathbf{T}}_n^{-1}\mathbf{B}_n\mathbf{E}_n\boldsymbol{b} \\ &= \mathbf{\Lambda}_{11}\mathbf{G}_n'\tilde{\mathbf{T}}_n^{-1}\mathbf{B}_n\mathbf{E}_n\boldsymbol{b}.\end{aligned}$$

Because the diagonal blocks of $\mathbf{B}_n$ are chosen to correspond in size to the number of variables in the nodes of the UWT, we have that $\mathbf{B}_n\mathbf{E}_n = \mathbf{E}_n\mathbf{\Lambda}$ and

$$\boldsymbol{\mu}_1^{(n-1)} = \mathbf{\Lambda}_{11}\mathbf{G}_n'\tilde{\mathbf{T}}_n^{-1}\mathbf{E}_n\mathbf{\Lambda}\boldsymbol{b} = \mathbf{\Lambda}_{11}\tilde{\boldsymbol{\mu}}_1^{(n-1)}, \tag{3.30}$$

where $\tilde{\boldsymbol{\mu}}_1^{(n-1)} = \mathbf{G}_n'\tilde{\mathbf{T}}_n^{-1}\mathbf{E}_n\mathbf{\Lambda}\boldsymbol{b}$. Equations (3.29) and (3.30) show that there is an easy way of moving between the computations done by GaBP on $\mathbf{S}$; $\boldsymbol{b}$ and $\tilde{\mathbf{S}}$; $\tilde{\boldsymbol{b}}$ that does not depend on the iteration number. The implication is that the convergence behaviour of GaBP on $\mathbf{S}$; $\boldsymbol{b}$ will be similar (near identical) to the convergence behaviour of GaBP on $\tilde{\mathbf{S}}$; $\tilde{\boldsymbol{b}}$. The way in which $\tilde{\mathbf{S}}$ and $\tilde{\boldsymbol{b}}$ are obtained from $\mathbf{S}$ and $\boldsymbol{b}$ is typically used to precondition inputs to iterative solvers of linear systems, such as the conjugate gradient (CG) method. In the case of the CG method, the difference in the convergence behaviour for the different sets of inputs can be substantial in the sense that the preconditioned variant converges much faster (Shewchuk, 1994). In this sense, GaBP implicitly preconditions the inputs as long as the preconditioner matrix only has non-zero elements within the clusters selected ($\mathbf{\Lambda}_{ij} = \mathbf{0} : d_i \times d_j$ for $i \neq j$). We shall revisit this in the empirical section.

3.5 Exploiting the Tree and Line Topology

In this section, we discuss some of the advantages of converting the topology of the graph on which GaBP operates. The following proposition will be used in the theorems of this section.

Proposition 1 *Consider any square matrix (possibly non-symmetric)* $\mathbf{A}$ *clustered exactly as* $\mathbf{S}$*. Let* $\mathbf{T}_n(\mathbf{A})$ *be the UWT, according to node 1 (say), for* $\mathbf{A}$*. We have that* $\mathbf{T}_n(\mathbf{A})\mathbf{E}_n = \mathbf{E}_n\mathbf{A} + \mathbf{L}_n(\mathbf{A})$*, where* $\mathbf{L}_n(\mathbf{A})$ *contains zeros except for the rows corresponding to the final block of* $\mathcal{L}_n$*. Moreover, the final block of* $\mathbf{T}_n(\mathbf{A})\mathbf{E}_n$ *contains rows that are sparse replicates of the rows of* $\mathbf{A}$*.*

The proof of Proposition 1 is similar to the proof of Equation (3.11) discussed in Section 3.2.2. For the purpose of this section (Section 3.5) we assume that the precision matrix $\mathbf{S}$ has all diagonal entries equal to one. The automatic preconditioning described in the previous section can be used to relate general precision matrices to this case (unit-diagonal precision matrix). Under this assumption, the precision of the UWT can be expressed as:

$$\mathbf{T}_n = \mathbf{I}_{m_n} - \mathbf{R}_n,$$

where

$$\mathbf{R}_n = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} & & & & & & & & \\ \mathbf{C}_{21} & \mathbf{C}_{22} & \mathbf{C}_{23} & & & & & & & \\ & \mathbf{C}_{32} & \mathbf{C}_{33} & \mathbf{C}_{34} & & & & & & \\ & & & \cdots & \cdots & \cdots & & & & \\ & & & & \cdots & \cdots & \cdots & & & \\ & & & & & \cdots & \cdots & \cdots & & \\ & & & & & & & \mathbf{C}_{n-1;n-2} & \mathbf{C}_{n-1;n-1} & \mathbf{C}_{n-1;n} \\ & & & & & & & & \mathbf{C}_{n;n-1} & \mathbf{C}_{nn} \end{bmatrix},$$

with the understanding that the blank spaces are all zero. The $\mathbf{C}_{ii}$ blocks are zero-diagonal square matrices with numbers of rows equal to the sums of the size of all nodes in the ith layer of the UWT. We now give some theorems relevant to the next sections. The proofs of these are given in Appendix B.

Theorem 7 *If* $\mathbf{S} = \mathbf{I} - \mathbf{R}$*, then* $\rho(|\mathbf{R}_n|) \leq \rho(|\mathbf{R}|)$ *for all* n*.*

Theorem 8 *The quantity* $||\mathbf{R}_n^k||_\infty \leq \kappa_1[\rho(|\mathbf{R}|)]^k$ *for a constant* κ_1 *that does not depend on* n*. If* $\rho(|\mathbf{R}|) < 1$*, then* $||\mathbf{T}_n^{-1}||_\infty \leq \kappa_2$ *for a constant* κ_2 *that does not depend on* n*.*

The discussions in Appendix B assumes that the matrix $\mathbf{R}$ is irreducible (the reducible case represents an uninteresting case where the graph structure factors into unconnected subgraphs). The Perron-Frobenius theorem guarantees

the existence of a vector $\mathbf{v} = (v_1, v_2, \ldots, v_k)' > \mathbf{0}$ such that $|\mathbf{R}|\mathbf{v} = \rho(|\mathbf{R}|)\mathbf{v}$. The constants κ_1 and κ_2 given in Theorem 8 can be computed as:

$$\kappa_1 = \frac{\max_i\{v_i\}}{\min_j\{v_j\}} \tag{3.31}$$

$$\kappa_2 = \frac{\kappa_1}{1 - \rho(|\mathbf{R}|)}. \tag{3.32}$$

Although Theorems 7 and 8 uses the UWT for cluster 1 as an example, we see that this cluster plays no role in the computation of κ_1 and κ_2. Theorems 7 and 8 also applies to UWTs constructed for other clusters, and the computation of κ_1 and κ_2 remains as in (3.31) and (3.32) respectively (they do not depend on the cluster chosen to represent the root node).

In the Sections 3.5.1 and 3.5.2, we make the additional assumption that $\rho(|\mathbf{R}|) < 1$, which is equivalent to $\mathbf{S}$ being walk-summable.

3.5.1 Convergence of the Posterior Means

Consider again $\boldsymbol{\mu}_1^{(n-1)} = \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{E}_n\boldsymbol{b}$, where $\boldsymbol{b}$ is any vector in $\Re^k$. Since $\mathbf{S}$ is positive definite, there is one vector $\boldsymbol{\mu}$ in $\Re^k$ with the property $\mathbf{S}\boldsymbol{\mu} = \mathbf{b}$. Let us consider solving $\mathbf{z}_n$ in

$$\mathbf{T}_n\mathbf{z}_n = \mathbf{E}_n\boldsymbol{b}. \tag{3.33}$$

Since $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$ we have $\mathbf{T}_n\mathbf{z}_n = \mathbf{E}_n\mathbf{S}\boldsymbol{\mu}$. Substitution of Equation (3.11) into this expression yields $\mathbf{T}_n\mathbf{z}_n = (\mathbf{T}_n\mathbf{E}_n - \mathbf{L}_n)\boldsymbol{\mu}$ or

$$\mathbf{z}_n = \mathbf{E}_n\boldsymbol{\mu} - \mathbf{T}_n^{-1}\mathbf{L}_n\boldsymbol{\mu}. \tag{3.34}$$

Using the fact that $\boldsymbol{\mu}_1^{(n-1)} = \mathbf{G}_n'\mathbf{z}_n$, we have

$$\boldsymbol{\mu}_1^{(n-1)} = \mathbf{G}_n'\mathbf{E}_n\boldsymbol{\mu} - \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{L}_n\boldsymbol{\mu} = \boldsymbol{\mu}_1 - \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{L}_n\boldsymbol{\mu}, \tag{3.35}$$

where $\boldsymbol{\mu}_1$ is the subvector of $\boldsymbol{\mu}$ corresponding to the variables in cluster 1. Since $\rho(\mathbf{R}_n) \leq \rho(|\mathbf{R}_n|) \leq \rho(|\mathbf{R}|) < 1$, by Theorem 6 and 7, we can express $\mathbf{T}_n^{-1}$ as a Neumann power series,

$$\mathbf{T}_n^{-1} = \sum_{i=0}^{\infty} \mathbf{R}_n^i. \tag{3.36}$$

Consider the vector $\mathbf{L}_n\boldsymbol{\mu}$. Since the rows of $\mathbf{L}_n$ corresponding to the first $n-1$ blocks are all equal to zero (see Proposition 1), we can write $\mathbf{L}_n\boldsymbol{\mu} = (\mathbf{0}_1', \mathbf{0}_2', \ldots, \mathbf{0}_{n-1}', \mathbf{h}_n')'$. The notation $\mathbf{0}_l$ stands for a vector containing a number of zeros equal to the sum of the dimensionalities of the nodes in layer l. The vector $\mathbf{h}_n$ is a specified vector (entries can be non-zero) equal in size to the

sum of the dimensionalities of the nodes in layer n. With this notation, it is easy to see that

$$\mathbf{G}'_n \mathbf{R}_n^i \mathbf{L}_n \boldsymbol{\mu} = \mathbf{0}_1$$

for all $i < n-1$ (note that the first layer contains only one node corresponding to cluster 1, hence $\mathbf{0}_1$ will contain d_1 zeros). We have:

$$\begin{aligned}
\mathbf{G}'_n \mathbf{T}_n^{-1} \mathbf{L}_n \boldsymbol{\mu} &= \mathbf{G}'_n \sum_{i=0}^{\infty} \mathbf{R}_n^i \mathbf{L}_n \boldsymbol{\mu} \\
&= \mathbf{G}'_n \sum_{i=n-1}^{\infty} \mathbf{R}_n^i \mathbf{L}_n \boldsymbol{\mu} \\
&= \mathbf{G}'_n \mathbf{R}_n^{n-1} \sum_{i=0}^{\infty} \mathbf{R}_n^i \mathbf{L}_n \boldsymbol{\mu} \\
&= \mathbf{G}'_n \mathbf{R}_n^{n-1} \sum_{i=0}^{\infty} \mathbf{R}_n^i \mathbf{L}_n \boldsymbol{\mu} \\
&= \mathbf{G}'_n \mathbf{R}_n^{n-1} \mathbf{T}_n^{-1} \mathbf{L}_n \boldsymbol{\mu}. \qquad (3.37)
\end{aligned}$$

The rows with non-zero entries of $\mathbf{L}_n$ are (sparse) replicates of a row of $\mathbf{S}$ by Proposition 1, and we can therefore say that $||\mathbf{L}_n \boldsymbol{\mu}||_\infty \leq ||\mathbf{S}||_\infty ||\boldsymbol{\mu}||_\infty = \kappa_3$ (does not depend on n). Since $||\mathbf{G}'_n||_\infty = 1$, we have

$$||\boldsymbol{\mu}_1^{(n-1)} - \boldsymbol{\mu}_1||_\infty \leq \kappa_4 [\rho(|\mathbf{R}|)]^{n-1}, \qquad (3.38)$$

where $\kappa_4 = \kappa_1 \kappa_2 \kappa_3$. Hence $\boldsymbol{\mu}_1^{(n)}$ will converge to $\boldsymbol{\mu}_1$ as $n \to \infty$. We can repeat this analysis for all clusters in order to see that $\boldsymbol{\mu}^{(n)}$ (all the marginal means at iteration n) will converge to $\boldsymbol{\mu}$ ($\boldsymbol{\mu} = \mathbf{S}^{-1}\boldsymbol{b}$) as $n \to \infty$, and this is true for all possible $\boldsymbol{b}$.

3.5.2 Convergence of the Posterior Variances

In this section we prove convergence of the posterior variance matrix $[\mathbf{P}_1^{(n-1)}]^{-1} = \mathbf{G}'_n \mathbf{T}_n^{-1} \mathbf{G}_n$ (this is more convenient to work with than the formula for $\mathbf{P}_1^{(n-1)}$). A proof of convergence of the posterior precisions is given in Section 3.7.

Note that the results from the previous section also apply when $\boldsymbol{b}$ is a matrix. Due to the order of the variables (see Section 3.1), we have that cluster 1 will contain the variables $1, 2, \ldots, d_1$. Let $\mathbf{e}_l : k \times 1$ be a vector of zeros except for entry l, which contains 1. Set $\boldsymbol{b} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \ldots & \mathbf{e}_{d_1} \end{bmatrix}$. Under the assumptions of the previous section we have

$$\mathbf{G}'_n \mathbf{T}_n^{-1} \mathbf{E}_n \boldsymbol{b} \to \lfloor \mathbf{S}^{-1} \rfloor_{11}, \qquad (3.39)$$

as $n \to \infty$, where $\lfloor \mathbf{S}^{-1} \rfloor_{11}$ is the submatrix of $\mathbf{S}^{-1}$ corresponding to the variables in cluster 1. The matrix $\mathbf{E}_n \boldsymbol{b}$ has a row-block decomposition according

to the nodes in the UWT. If a node in the UWT does not reference cluster 1, then the corresponding block of $\mathbf{E}_n\boldsymbol{b}$ will be zero, otherwise the block is $\mathbf{I}_{d_1}$ (this is because $\mathbf{F}_t\mathbf{e}_l$, for $t \neq 1$ and $l \leq d_1$, extracts a subvector of $\mathbf{e}_l$ containing only zeros). As a consequence, we have $\mathbf{E}_n\boldsymbol{b} = \mathbf{G}_n + \tilde{\mathbf{E}}_n$, where $\tilde{\mathbf{E}}_n$ is equal to $\mathbf{E}_n\boldsymbol{b}$, except for the first block, which contains only zeros. We can write

$$[\mathbf{P}_1^{(n-1)}]^{-1} = \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{G}_n \sim \lfloor \mathbf{S}^{-1} \rfloor_{11} - \mathbf{G}_n'\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n \tag{3.40}$$

for large n. It remains to show that $\mathbf{Z}_n = \mathbf{G}_n'\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n \to \mathbf{Z}_\infty$ for a specified matrix $\mathbf{Z}_\infty$ as $n \to \infty$. We prove that $\mathbf{Z}_n$ does converge (under our assumptions) in Theorem 9. The proof is given in Appendix B.

Theorem 9 *If $\rho(|\mathbf{R}|) < 1$, the sequence $\mathbf{Z}_n$ is a Cauchy sequence and will converge to a specified matrix $\mathbf{Z}_\infty$.*

Hence, if $\rho(|\mathbf{R}|) < 1$, then

$$[\mathbf{P}_1^{(n)}]^{-1} \to \lfloor \mathbf{S}^{-1} \rfloor_{11} - \mathbf{Z}_\infty \tag{3.41}$$

as $n \to \infty$. In particular, we see that the converged posterior variances are not exact in general and the error depends on $\mathbf{Z}_\infty$. Note that, if the computation tree never references node 1 again (as in the case of a tree-structured $\mathbf{S}$), then the variance matrix will be equal to the exact marginal variance matrix.

We see that the error associated with the posterior variance matrix at iteration n is $[\mathbf{P}_1^{(n)}]^{-1} - \lfloor \mathbf{S}^{-1} \rfloor_{11} = -\mathbf{Z}_n$. Since the first two blocks (first two layers of the UWT) of $\tilde{\mathbf{E}}_n$ contain only zeros, we can write $\mathbf{Z}_n = \mathbf{G}_n'\mathbf{R}_n^2\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n$ if $n > 2$. Furthermore, since $||\mathbf{G}_n'\mathbf{R}_n^2\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n||_\infty \leq \kappa_1\kappa_2[\rho(|\mathbf{R}|)]^2$, we see that

$$||\mathbf{Z}_n||_\infty \leq \kappa_1\kappa_2[\rho(|\mathbf{R}|)]^2. \tag{3.42}$$

It is possible to derive better bounds on this infinity norm depending on the topology of $\mathbf{S}$. Consider the UWT for node 1 and suppose node 1 is referenced for the first time (after the root layer) at layer t. See, for instance, Figure 3.1, where $t = 4$. Since $\tilde{\mathbf{E}}_n$ will contain only zeros for the first $t-1$ layers, we can write $\mathbf{Z}_n = \mathbf{G}_n'\mathbf{R}_n^{t-1}\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n$ for $n > t-1$, and hence the bound changes to

$$||\mathbf{Z}_n||_\infty \leq \kappa_1\kappa_2[\rho(|\mathbf{R}|)]^{t-1}. \tag{3.43}$$

The bound in Inequality (3.43) improves on the bound given in Inequality (3.42). We note that the bounds in (3.42) and (3.43) are bounds on the accuracy of the approximate variance supplied by GaBP at convergence.

In Sections 3.5.1 and 3.5.2 we have shown that, for any walk-summable, unit-diagonal precision matrix $\mathbf{S}$ and any potential vector $\boldsymbol{b}$, the posterior mean and posterior variance matrix associated with cluster 1 will converge. In a similar way, these results can be extended to other clusters. We summarise these results in the following theorem:

Theorem 10 *Consider any walk-summable and unit-diagonal precision matrix* $\mathbf{S}$*, any potential vector* $\boldsymbol{b}$ *and clusters* $\mathcal{C}_i : i = 1, 2, \ldots, p$*. The posterior mean and posterior variance matrix, associated with each cluster, will converge as the number of iterations performed by GaBP tends to infinity. Moreover, the converged posterior means represent the exact marginal means, while the converged posterior variance matrices are not necessarily equal to the true marginal variance matrices.*

In the next section we use Theorem 10 to derive a sufficient condition for the convergence of sGaBP when applied to arbitrary precision matrices.

3.6 Convergence and Preconditioning

In this section we consider GaBP applied to an arbitrary precision matrix $\mathbf{S}$ and potential vector $\boldsymbol{b}$. We call the matrix $\mathbf{S}$ preconditioned walk-summable if there exists a $\mathbf{\Lambda}$ (of the type described in Section 3.4) such that $\tilde{\mathbf{S}} = \mathbf{\Lambda S \Lambda}$ is walk-summable. We note that we can assume, without loss of generality, that all the diagonal entries of $\tilde{\mathbf{S}}$ equal one (this can be achieved by diagonal preconditioning and we can incorporate this in $\mathbf{\Lambda}$). We see that we can apply Theorem 10 to the precision matrix $\tilde{\mathbf{S}}$ and the potential vector $\tilde{\boldsymbol{b}}$. Recall that $\tilde{\boldsymbol{b}} = \mathbf{\Lambda} \boldsymbol{b}$. Let $\boldsymbol{\mu} = \mathbf{S}^{-1}\boldsymbol{b}$, $\tilde{\boldsymbol{\mu}} = \tilde{\mathbf{S}}^{-1}\tilde{\boldsymbol{b}}$ and $\tilde{\mathbf{R}} = \mathbf{I} - \tilde{\mathbf{S}}$. It is easy to see that $\boldsymbol{\mu} = \mathbf{\Lambda}\tilde{\boldsymbol{\mu}}$. Recall that $\tilde{\boldsymbol{\mu}}_i^{(n-1)}$ and $\tilde{\mathbf{P}}_i^{(n-1)}$ represent the posterior mean and posterior precision, supplied by GaBP on the preconditioned inputs after $n-1$ iterations for cluster i, respectively.

From Theorem 10 we know that,

$$\lim_{n\to\infty} \tilde{\boldsymbol{\mu}}_i^{(n-1)} = \tilde{\boldsymbol{\mu}}_i,$$

and therefore

$$\lim_{n\to\infty} \boldsymbol{\mu}_i^{(n-1)} = \lim_{n\to\infty} \mathbf{\Lambda}_{ii}\tilde{\boldsymbol{\mu}}_i^{(n-1)} = \mathbf{\Lambda}_{ii}\tilde{\boldsymbol{\mu}}_i = \boldsymbol{\mu}_i, \tag{3.44}$$

by Equation (3.30). Equation (3.44) shows that, under preconditioned walk-summability, the posterior means will converge to the correct marginal means. In Equation (3.41) we established that:

$$\lim_{n\to\infty} [\tilde{\mathbf{P}}_i^{(n-1)}]^{-1} = \lfloor \tilde{\mathbf{S}}^{-1} \rfloor_{ii} - \tilde{\mathbf{Z}}_{i;\infty},$$

for a specified matrix $\tilde{\mathbf{Z}}_{i;\infty}$. Hence,

$$\lim_{n\to\infty} [\mathbf{P}_i^{(n-1)}]^{-1} = \lim_{n\to\infty} \mathbf{\Lambda}_{ii}[\tilde{\mathbf{P}}_i^{(n-1)}]^{-1}\mathbf{\Lambda}_{ii} = \mathbf{\Lambda}_{ii}\lfloor \tilde{\mathbf{S}}^{-1} \rfloor_{ii}\mathbf{\Lambda}_{ii} - \mathbf{\Lambda}_{ii}\tilde{\mathbf{Z}}_{i;\infty}\mathbf{\Lambda}_{ii}, \tag{3.45}$$

and we see that preconditioned walk-summability implies convergence of the posterior variances. Note that since, $\mathbf{S}^{-1} = \mathbf{\Lambda}\tilde{\mathbf{S}}^{-1}\mathbf{\Lambda}$, we have $\mathbf{\Lambda}_{ii}\lfloor \tilde{\mathbf{S}}^{-1} \rfloor_{ii}\mathbf{\Lambda}_{ii} = \lfloor \mathbf{S}^{-1} \rfloor_{ii}$. We define $\mathbf{\Lambda}_{ii}\tilde{\mathbf{Z}}_{i;\infty}\mathbf{\Lambda}_{ii} = \mathbf{Z}_{i;\infty}$. Equations (3.44) and (3.45) establish the following theorem:

Theorem 11 *Consider GaBP on a precision matrix* $\mathbf{S}$ *and potential vector* $\boldsymbol{b}$ *with clusters* $\mathcal{C}_i : i = 1, 2, \ldots, p$. *If* $\mathbf{S}$ *is preconditioned walk-summable with respect to the clusters* $\mathcal{C}_i : i = 1, 2, \ldots, p$, *then GaBP will converge. This convergence is in the sense that:*

$$\boldsymbol{\mu}_i^{(n)} \to \boldsymbol{\mu}_i \tag{3.46}$$

$$[\mathbf{P}_i^{(n)}]^{-1} \to \lfloor \mathbf{S}^{-1} \rfloor_{ii} - \mathbf{Z}_{i;\infty}, \tag{3.47}$$

as $n \to \infty$ *and* $\mathbf{Z}_{i;\infty}$ *is a certain matrix. This is true for all clusters* i.

In the remainder of this section, we discuss a bound on the rate of convergence and a bound on the accuracy of the posterior variance matrices as approximations for the true marginal variances. Inequality (3.38) implies that

$$||\tilde{\boldsymbol{\mu}}_i^{(n-1)} - \tilde{\boldsymbol{\mu}}_i||_\infty \leq \kappa_4 [\rho(|\tilde{\mathbf{R}}|)]^{n-1},$$

since κ_4 does not depend on the root node ($\kappa_4 = \kappa_1\kappa_2\kappa_3$ and these constants remain unchanged regardless of the cluster selected to represent the root node of the UWT). Since $\boldsymbol{\mu}_i^{(n-1)} = \boldsymbol{\Lambda}_{ii}\tilde{\boldsymbol{\mu}}_i^{(n-1)}$, we see that

$$\begin{aligned} ||\boldsymbol{\mu}_i^{(n-1)} - \boldsymbol{\mu}_i||_\infty &= ||\boldsymbol{\Lambda}_{ii}\tilde{\boldsymbol{\mu}}_i^{(n-1)} - \boldsymbol{\Lambda}_{ii}\tilde{\boldsymbol{\mu}}_i||_\infty \\ &\leq \kappa_4 ||\boldsymbol{\Lambda}_{ii}||_\infty [\rho(|\tilde{\mathbf{R}}|)]^{n-1}. \end{aligned}$$

Note that $||\boldsymbol{\mu}^{(n)} - \boldsymbol{\mu}||_\infty \leq \epsilon$ if $\kappa_4 \max_i\{||\boldsymbol{\Lambda}_{ii}||_\infty\}[\rho(|\tilde{\mathbf{R}}|)]^n \leq \epsilon$ or

$$n \geq \frac{\log(\frac{\epsilon}{\kappa_4 \max_i\{||\boldsymbol{\Lambda}_{ii}||_\infty\}})}{\log(\rho(|\tilde{\mathbf{R}}|))}. \tag{3.48}$$

Usually, convergence of the posterior means requires convergence of the posterior precisions and Inequality (3.48) should be an upper bound for the number of iterations required by the whole algorithm to converge.

If we combine Inequality (3.43) with Equation (3.29), we obtain the following bound:

$$\begin{aligned} ||[\mathbf{P}_i^{(n)}]^{-1} - \lfloor \mathbf{S}^{-1} \rfloor_{ii}||_\infty &= ||\boldsymbol{\Lambda}_{ii}[\tilde{\mathbf{P}}_i^{(n)}]^{-1}\boldsymbol{\Lambda}_{ii} - \boldsymbol{\Lambda}_{ii}\lfloor \tilde{\mathbf{S}}^{-1} \rfloor_{ii}\boldsymbol{\Lambda}_{ii}||_\infty \\ &\leq ||\boldsymbol{\Lambda}_{ii}||_\infty^2 ||[\tilde{\mathbf{P}}_i^{(n)}]^{-1} - \lfloor \tilde{\mathbf{S}}^{-1} \rfloor_{ii}||_\infty \\ &\leq \kappa_1\kappa_2 ||\boldsymbol{\Lambda}_{ii}||_\infty^2 [\rho(|\tilde{\mathbf{R}}|)]^{t-1}, \end{aligned} \tag{3.49}$$

where κ_1 and κ_2 are computed with respect to $\tilde{\mathbf{S}}$ and the largest t for which this inequality holds depends on the topology of the MG.

Our final theoretical discussion involve the use of UWTs for computing the precision components of the messages (as opposed to the precisions of the posterior distributions).

3.7 Using UWTs for Messages

This section is relevant for the discussion in Chapter 4. Let $i \in \mathcal{N}_1$, and suppose we want to use the UWT to compute:

$$\mathbf{Q}_{i1}^{(n-1)} = -\mathbf{S}_{1i}[\mathbf{P}_{i1}^{(n-2)}]^{-1}\mathbf{S}_{i1}. \tag{3.50}$$

The matrix $\mathbf{P}_{i1}^{(n-2)}$ can be computed by marginalising a certain subtree of the UWT associated with cluster 1 and extracting the precision matrix corresponding to the root node. Let $\mathcal{T}_n^{(i)}$ be the subtree of $\mathcal{T}_n$ rooted at the node in the second layer with a reference to cluster i. The tree-pruning procedure can be used to show that

$$\mathbf{P}_{i1}^{(n-2)} = r_P(\mathcal{T}_n^{(i)}). \tag{3.51}$$

Let $\mathbf{T}_n^{(i)}$ be the precision matrix associated with $\mathcal{T}_n^{(i)}$ and let $\mathbf{G}_{n;i}$ be defined analogous to $\mathbf{G}_n$. With this notation we see that:

$$\mathbf{P}_{i1}^{(n-2)} = [\mathbf{G}'_{n;i}\mathbf{T}_n^{(i)}\mathbf{G}_{n;i}]^{-1}. \tag{3.52}$$

The same arguments used to derive the convergence of $[\mathbf{P}_1^{(n-1)}]^{-1}$ can be applied to $[\mathbf{P}_{i1}^{(n-2)}]^{-1}$ in the sense that

$$\lim_{n\to\infty}[\mathbf{P}_{i1}^{(n-2)}]^{-1} \tag{3.53}$$

will exist if $\mathbf{S}$ is preconditioned walk-summable. These considerations should be enough to convince the reader that the following theorem holds:

Theorem 12 *Consider application of GaBP to a precision matrix* $\mathbf{S}$ *and potential vector* $\boldsymbol{b}$*. If* $\mathbf{S}$ *is preconditioned walk-summable then*

$$\lim_{n\to\infty}[\mathbf{P}_{ij}^{(n-2)}]^{-1} = \ddot{\mathbf{P}}_{ij}, \tag{3.54}$$

for a specified matrix $\ddot{\mathbf{P}}_{ij}$*. This holds for all clusters* i *and all* $j \in \mathcal{N}_i$*.*

A consequence of Theorem 12 is that

$$\lim_{n\to\infty}\mathbf{Q}_{ij}^{(n)} = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}, \tag{3.55}$$

and

$$\lim_{n\to\infty}\mathbf{P}_{i}^{(n)} = \mathbf{S}_{ii} - \sum_{t\in\mathcal{N}_i}\mathbf{S}_{it}\ddot{\mathbf{P}}_{ti}\mathbf{S}_{ti}. \tag{3.56}$$

In this chapter so far, we have covered several theoretical aspects of the GaBP algorithm with nodes of arbitrary size. In the next section we investigate the implications of these results empirically.

3.8 Empirical Results

In our first empirical investigation we construct a specific cluster-dependent convergence condition for GaBP based on the automatic preconditioning done by this algorithm and apply this to a practical dataset. We then illustrate the automatic preconditioning by comparing GaBP to the CG and preconditioned CG (PCG) methods.

3.8.1 Preconditioned Walk-summability

Consider applying GaBP to a precision matrix $\mathbf{S} : k \times k$ and potential vector $\boldsymbol{b} : k \times 1$ where the variables are assigned to nodes based on the clusters $\mathcal{C}_i : i = 1, 2, \ldots, p$. We have not discussed the selection of $\boldsymbol{\Lambda}$ in practice. One type of preconditioner we found to be effective in the prediction of the convergence of GaBP is the selection:

$$\boldsymbol{\Lambda}_{ii} = \mathbf{S}_{ii}^{-0.5} \text{ for } i = 1, 2, \ldots, p$$
$$\boldsymbol{\Lambda}_{ij} = \mathbf{0} : d_i \times d_j \text{ for } i \neq j.$$

We illustrate the effectiveness of this preconditioner by considering the diabetes data (Efron *et al.*, 2004).

For illustration purposes we only consider the variables age, sex, bmi, map, tc and ldl clustered into groups 1 = (age ; sex), 2 = (bmi ; map) and 3= (tc ; ldl). We use as our precision matrix and potential vector the correlation matrix and correlation vector of these variables respectively. This defines a fully connected MG according to the clusters. The precision matrix and potential vector supplied to GaBP are

$$\mathbf{S} = \begin{array}{c} \\ \text{age} \\ \text{sex} \\ \text{bmi} \\ \text{map} \\ \text{tc} \\ \text{ldl} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{age} & \text{sex} & \text{bmi} & \text{map} & \text{tc} & \text{ldl} \end{array} \\ \begin{bmatrix} 1.0000000 & 0.17373710 & 0.1850847 & 0.3354267 & 0.26006082 & 0.2192431 \\ 0.1737371 & 1.00000000 & 0.0881614 & 0.2410132 & 0.03527682 & 0.1426373 \\ 0.1850847 & 0.08816140 & 1.0000000 & 0.3954153 & 0.24977742 & 0.2611699 \\ 0.3354267 & 0.24101317 & 0.3954153 & 1.0000000 & 0.24246971 & 0.1855578 \\ 0.2600608 & 0.03527682 & 0.2497774 & 0.2424697 & 1.00000000 & 0.8966630 \\ 0.2192431 & 0.14263726 & 0.2611699 & 0.1855578 & 0.89666296 & 1.0000000 \end{bmatrix} \end{array}$$

$$\boldsymbol{b} = \begin{array}{c} \\ \text{age} \\ \text{sex} \\ \text{bmi} \\ \text{map} \\ \text{tc} \\ \text{ldl} \end{array} \begin{array}{c} \\ \begin{bmatrix} 0.1878888 \\ 0.0430620 \\ 0.5864501 \\ 0.4414838 \\ 0.2120225 \\ 0.1740536 \end{bmatrix} \end{array},$$

respectively. It can be shown that the spectral radius of $|\mathbf{I} - \mathbf{S}|$ is $1.4069 > 1$, which does not give a decisive answer on whether GaBP will converge; in fact, we see that GaBP converges after 19 iterations where $\epsilon = 10^{-10}$. The spectral radius of $|\mathbf{I} - \tilde{\mathbf{S}}| = 0.6689 < 1$, which shows that $\mathbf{S}$ is preconditioned walk-summable, and this explains the convergence of GaBP in this context.

Furthermore, we see that $\kappa_1 = 1.781554$, $\kappa_2 = \frac{\kappa_1}{1-0.6689} = 5.380973$, $\kappa_3 = 0.9290677$ and $\max_i\{||\mathbf{\Lambda}_{ii}||_\infty\} = 3.110799$. From this we see that Inequality (3.48) predicts convergence after at most 66 iterations (almost 3.5 times the actual number).

The bound in Equation (3.49), with $t = 3$, can be evaluated as 5.191402 for cluster 1, which is poor compared to the actual value of 0.0929461. The bound in Equation (3.49) will likely be better for sparse graphs.

3.8.2 Automatic Preconditioning

In this section, we compare GaBP to the CG and PCG algorithms. A description of the CG algorithm can be found in Shewchuk (1994). Our goal with this empirical comparison is not to promote GaBP as a solver of linear systems, but rather to illustrate the stability of GaBP with regard to certain preconditioning. The idea is to compare the performance of these algorithms by considering the application of the basic methods to an ill-conditioned $\mathbf{S}$. The performances on a well-conditioned matrix $\tilde{\mathbf{S}} = \mathbf{\Lambda S \Lambda}$ are also compared.

Kamper *et al.* (2018*a*) specify a method for generating precision matrices with an arbitrary zero-diagonal spectral radius. The zero-diagonal spectral radius of $\mathbf{S}$ is obtained as the spectral radius of $\mathbf{R}$ in $\mathbf{S} = \mathbf{I} - \mathbf{R}$, assuming that $\mathbf{S}$ has been scaled to have only ones along its diagonal. Our first illustration aims to illustrate the automatic diagonal preconditioning when we apply univariate GaBP. To generate a data structure, we do the following:

1. Select a zero-diagonal spectral radius ($\tilde{\rho}$) uniformly from the interval $[0.5; 0.9]$.

2. Use the method from Kamper *et al.* (2018*a*) to generate a precision matrix $\tilde{\mathbf{S}} : 1\,000 \times 1\,000$ and potential vector $\tilde{\boldsymbol{b}} : 1\,000 \times 1$. These will act as our preconditioned inputs.

3. Generate $\mathbf{u} : 1\,000 \times 1$, where each element is selected independently from a uniform random distribution over the interval $[0.01; 1]$. Set $\mathbf{D_u} = \text{diag}(\mathbf{u})$.

4. Our original (ill-conditioned) inputs are $\mathbf{S} = \mathbf{D_u}\tilde{\mathbf{S}}\mathbf{D_u}$ and $\boldsymbol{b} = \mathbf{D_u}\tilde{\boldsymbol{b}}$.

We generate 1 000 of these data structures randomly and compare four methods. The methods are univariate GaBP on $\{\mathbf{S}, \boldsymbol{b}\}$, univariate GaBP on $\{\tilde{\mathbf{S}}, \tilde{\boldsymbol{b}}\}$, CG on $\{\mathbf{S}, \boldsymbol{b}\}$ and CG on $\{\tilde{\mathbf{S}}, \tilde{\boldsymbol{b}}\}$. The labels for these methods are gabp.org, gabp.pre, cg.org and cg.pre respectively. The "org" denotes application to the original ill-conditioned inputs, while the "pre" denotes the preconditioned inputs. We compare these methods by recording the number of iterations required for convergence by each method. Convergence was defined to occur when $||\mathbf{S}\boldsymbol{\mu}^{(n)} - \boldsymbol{b}||_\infty \leq 10^{-6}$. The results of our simulations are given in the top graph of Figure 3.3. We see that the performance of GaBP on the different sets of inputs is nearly identical. This is in contrast to the CG method, where the difference in performance is substantial and the automatic preconditioning effect of GaBP is well illustrated. The CG method on the original inputs performs the worst out of all the methods by far; however, the preconditioned version outperforms the GaBP variants. It is important to note that the diagonal PCG method requires the specification of a preconditioner and does not keep the computations on the scale of the original inputs.

In our second illustration, we consider the automatic preconditioning done by GaBP with higher-dimensional nodes, which is effectively block-diagonal preconditioning. We consider 1 000 $\times$ 1 000 precision matrices where we have 31 clusters each of size 31 and one additional cluster of size 39. Consider the following simulation procedure:

1. Select a zero-diagonal spectral radius ($\tilde{\rho}$) uniformly from the interval $[0.5; 0.9]$.

2. Use the method from Kamper *et al.* (2018*a*) to generate a precision matrix $\tilde{\mathbf{S}}$: 1 000 $\times$ 1000 and potential vector $\tilde{\boldsymbol{b}}$: 1 000 $\times$ 1. These will act as our preconditioned inputs.

3. We define a 1 000 $\times$ 1 000 matrix $\mathbf{Q}$. For each cluster i of size d_i, we generate a $d_i \times d_i$ precision matrix with zero-diagonal spectral radius equal to 1.5, and set $\mathbf{Q}_{ii}$ equal to this matrix. When $i \neq j$, $\mathbf{Q}_{ij}$ is a matrix of zeros.

4. Our original (ill-conditioned) inputs are $\mathbf{S} = \mathbf{Q}\tilde{\mathbf{S}}\mathbf{Q}$ and $\boldsymbol{b} = \mathbf{Q}\tilde{\boldsymbol{b}}$.

Again, we generate 1 000 of these structures and compare four different methods. We apply GaBP to $\{\mathbf{S}, \boldsymbol{b}\}$ with clusters as used in the simulation, and GaBP to $\{\tilde{\mathbf{S}}, \tilde{\boldsymbol{b}}\}$ with clusters as used in the simulation. These methods are labelled gabp.org and gabp.pre respectively. Let $\mathbf{D}_Q$ be the diagonal matrix with diagonal obtained from the diagonal of $\mathbf{Q}$. We define a diagonal preconditioner $\mathbf{D}_Q^{-1}$ (this is what was used in the previous simulations) and apply CG to $\{\mathbf{D}_Q^{-1}\mathbf{S}\mathbf{D}_Q^{-1}, \mathbf{D}_Q^{-1}\boldsymbol{b}\}$. This method is labelled cg.diag. The last method is CG applied to $\{\tilde{\mathbf{S}}, \tilde{\boldsymbol{b}}\}$, which we label cg.pre. The results are illustrated on

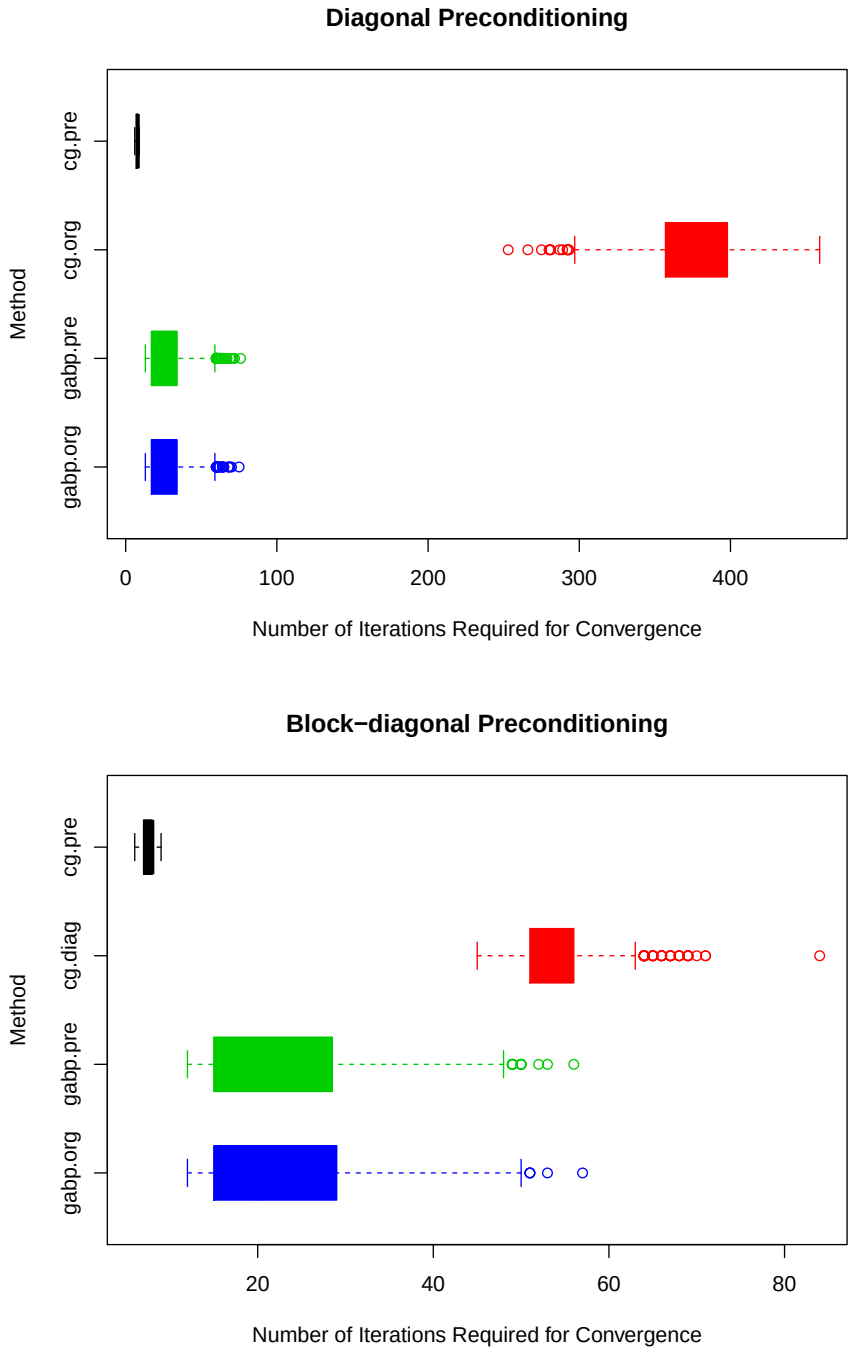


Figure 3.3: Visualisation of the results of our simulations regarding preconditioning. The top and bottom graph represent diagonal and block-diagonal preconditioning respectively. In both cases, we see that the performance of GaBP on the preconditioned and original inputs is nearly identical. This is in contrast to the performance of the CG method on both sets of inputs, where the difference is substantial. The automatic preconditioning of GaBP is well illustrated.

the bottom graph of Figure 3.3. We see a similar pattern as in the diagonal preconditioning case. The CG-based variants show the most volatility towards the preconditioning. In contrast to the diagonal case, we see that the GaBP variants now tend to converge with a smaller number of iterations when compared to the diagonally preconditioned CG variant. The full preconditioned CG variant performs the best of all the methods.

3.9 Conclusion

In this chapter, we conducted a UWT analysis of unregularised GaBP where nodes were allowed to be of any size. The idea was to convert GaBP to BP on a graph with a more convenient topology. The UWT analysis gave rise to analytical formulas for the posterior means and posterior precisions after each iteration. These formulas were used to describe a certain type of preconditioning done automatically by the GaBP algorithm. The automatic preconditioning done by GaBP was then used to derive a new sufficient condition for convergence. The implication of this property for comparison with other iterative solvers of linear systems was also discussed. The results of this chapter are also relevant for the proof of convergence of sGaBP with nodes of any size. This proof is provided in Chapter 4. In particular, the analytical formulas for the precision components of GaBP are used to derive certain asymptotic expressions. These expressions are used to study the asymptotic eigenvalues of a linear-update matrix (similar to Chapter 1), the spectral radius of which determines whether or not the algorithm converges.

Chapter 4

Higher-dimensional Regularised Gaussian Belief Propagation

In this chapter, we consider sGaBP where nodes are allowed to be of any size. Basically, we extend the results of Chapter 2 (univariate nodes) to the multivariate case. The form of this chapter is similar to that of Chapter 2.

4.1 Preliminaries

We give a brief recap of the notation used for Algorithm 2 given in Chapter 1. The problem addressed by sGaBP is marginalising a multivariate Gaussian distribution with precision matrix $\mathbf{S} : k \times k$ and potential vector $\boldsymbol{b} : k \times 1$ into clusters $\mathcal{C}_i : i = 1, 2, \ldots, p$, where $|\mathcal{C}_i| = d_i$, the $\mathcal{C}_i$ do not overlap and $\cup_{i=1}^{p} \mathcal{C}_i$ contains all the variables. The MG consists of p nodes, each corresponding to a different $\mathcal{C}_i$. We associate with node i the precision $\mathbf{S}_{ii}$ and the potential $\boldsymbol{b}_i$, where $\mathbf{S}_{ii}$ is the submatrix of $\mathbf{S}$ corresponding to the variables in $\mathcal{C}_i$ and $\boldsymbol{b}_i$ is the subvector of $\boldsymbol{b}$ corresponding to the same variables. Nodes i and j are linked by the matrix $\mathbf{S}_{ij}$, which is the submatrix of $\mathbf{S}$ corresponding to the variables in $\mathcal{C}_i$ and $\mathcal{C}_j$ for the rows and columns respectively. This notation is also used for other matrices of the same dimension as $\mathbf{S}$. For instance, if $\boldsymbol{\Lambda}$ is a $k \times k$ matrix, then $\boldsymbol{\Lambda}_{ij}$ is also the submatrix of $\boldsymbol{\Lambda}$ corresponding to the variables in $\mathcal{C}_i$ and $\mathcal{C}_j$ for the rows and columns respectively. The neighbourhood of node i is defined to be $\mathcal{N}_i = \{j \neq i : \mathbf{S}_{ij} \neq \mathbf{0} : d_i \times d_j\}$, and by $\mathcal{N}_i \setminus j$ we mean $\mathcal{N}_i$ with variable j removed.

Recall (see Section 1.4) that the synchronous message updates for sGaBP

can be done through

$$\mathbf{Q}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}(\lambda)]^{-1}\mathbf{S}_{ij} \tag{4.1}$$

$$\mathbf{v}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_{ij}^{(n)}(\lambda)]^{-1}[\lambda\boldsymbol{\mu}_i^{(n-1)} + \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{v}_{ti}^{(n)}] \tag{4.2}$$

$$\boldsymbol{\mu}_i^{(n+1)} = [\mathbf{P}_i^{(n+1)}(\lambda)]^{-1}[\lambda\boldsymbol{\mu}_i^{(n)} + \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i}\mathbf{v}_{ti}^{(n)}] \tag{4.3}$$

for all i and all $j \in \mathcal{N}_i$, $\mathbf{P}_{ij}^{(n)}(\lambda) = \lambda\mathbf{I}_{d_i} + \mathbf{S}_{ii} + \sum_{t\in\mathcal{N}_i\setminus j}\mathbf{Q}_{ti}^{(n)}$ and $\mathbf{P}_i^{(n+1)}(\lambda) = \mathbf{P}_{ij}^{(n+1)}(\lambda) + \mathbf{Q}_{ji}^{(n+1)}$. Note that $\mathbf{Q}_{ji}^{(n+1)}$ and $\mathbf{P}_{ij}^{(n)}(\lambda)$ are both $d_i \times d_i$ matrices, while $\mathbf{v}_{ji}^{(n+1)}$ and $\boldsymbol{\mu}_i^{(n+1)}$ are $d_i \times 1$ vectors. The message updates in Equations (4.1), (4.2) and (4.3) are implemented efficiently in Algorithm 2. At iteration n, the estimated mean and precision for the marginal of node i are $\hat{\boldsymbol{\mu}}_i^{(n)} = \boldsymbol{\mu}_i^{(n)}$ and $\hat{\mathbf{P}}_i^{(n)} = \mathbf{P}_i^{(n)}(\lambda) - \lambda\mathbf{I}_{d_i}$ respectively. We refer to these quantities as the posterior mean and posterior precision respectively of node i at iteration n. We introduce the following notation,

$$\ddot{\mathbf{P}}_i^{(n-1)}(\lambda) = [\mathbf{P}_i^{(n-1)}(\lambda)]^{-1}$$
$$\ddot{\mathbf{P}}_{ij}^{(n-2)}(\lambda) = [\mathbf{P}_{ij}^{(n-2)}(\lambda)]^{-1}.$$

4.1.1 UWT for the Precision Components

For the purpose of this chapter, we need to redefine some of the notation used in Chapter 3 to allow for UWTs associated with different clusters. Define $\mathbf{S}(\lambda) = \mathbf{S} + \lambda\mathbf{I}_k$.

Definition 10 *Consider the UWT, of depth n, constructed for cluster i of $\mathbf{S}(\lambda)$ as root. We denote the precision matrix of this UWT by $\mathbf{T}_{ii}^{(n)}(\lambda)$.*

Definition 11 *Suppose we have constructed the UWT, of depth n, for cluster i of $\mathbf{S}(\lambda)$ as root. Let $j \in \mathcal{N}_i$ and consider the subtree of the UWT, rooted at the node in the second layer corresponding to cluster j. We denote the precision matrix of this subtree by $\mathbf{T}_{ji}^{(n)}(\lambda)$.*

We refer to Appendix D for examples of $\mathbf{T}_{ii}^{(n)}(\lambda)$ and $\mathbf{T}_{ji}^{(n)}(\lambda)$.

From Equation (4.1), we see that $\mathbf{Q}_{ij}^{(n)}$ from sGaBP applied to $\mathbf{S}$ can be obtained by taking $\mathbf{Q}_{ij}^{(n)}$ from unregularised GaBP applied to $\mathbf{S}(\lambda)$. As a consequence:

$$\ddot{\mathbf{P}}_i^{(n-1)}(\lambda) = [\mathbf{P}_i^{(n-1)}(\lambda)]^{-1} = [\mathbf{G}_{ii}^{(n)}]'[\mathbf{T}_{ii}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ii}^{(n)}$$
$$\ddot{\mathbf{P}}_{ij}^{(n-2)}(\lambda) = [\mathbf{P}_{ij}^{(n-2)}(\lambda)]^{-1} = [\mathbf{G}_{ij}^{(n)}]'[\mathbf{T}_{ij}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ij}^{(n)},$$

where $[\mathbf{G}_{ii}^{(n)}]'$ and $[\mathbf{G}_{ij}^{(n)}]'$ (analogous to $\mathbf{G}'_n$ and $\mathbf{G}'_{n;i}$in Chapter 3) are matrices in which the first d_i columns are $\mathbf{I}_{d_i}$ and the rest are zeros ($n > 2$).

We make some notes regarding the notation used:

1. $\mathbf{S}(\lambda)$ is of the same dimensionality as $\mathbf{S}$. By $\mathbf{S}_{ij}(\lambda)$ we mean the submatrix of $\mathbf{S}(\lambda)$ corresponding to the clusters in $\mathcal{C}_i$ (for the rows) and $\mathcal{C}_j$ (for the columns). For the purpose of this chapter, we call this type of referencing clustered subscripting.

2. In $\mathbf{T}_{ii}^{(n)}(\lambda)$, the i in the subscript refers to the root cluster of the associated UWT (and not a submatrix).

4.1.2 Convergence of the Precision Components

The results from Chapter 3 can be applied to show that if, $\mathbf{S}(\lambda) = \mathbf{S} + \lambda\mathbf{I}_k$ is preconditioned walk-summable, then the precision components of sGaBP on $\mathbf{S}$ will converge (since this is identical to unregularised GaBP on $\mathbf{S}(\lambda)$). The preconditioned walk-summability can always be obtained by setting λ sufficiently large (for instance large enough for $\mathbf{S}(\lambda)$ to be diagonally dominant). Let λ_0 be such that $\mathbf{S}(\lambda)$ is preconditioned walk-summable if $\lambda > \lambda_0$. From Theorem 11 we know that

$$\lim_{n\to\infty} \ddot{\mathbf{P}}_i^{(n-1)}(\lambda) = \ddot{\mathbf{P}}_i(\lambda), \tag{4.4}$$

for a specified matrix $\ddot{\mathbf{P}}_i(\lambda)$. Furthermore, from Theorem 12,

$$\lim_{n\to\infty} \ddot{\mathbf{P}}_{ij}^{(n-2)}(\lambda) = \ddot{\mathbf{P}}_{ij}(\lambda), \tag{4.5}$$

for a specified matrix $\ddot{\mathbf{P}}_{ij}(\lambda)$. In both (4.4) and (4.5) we are assuming that $\lambda > \lambda_0$.

4.2 Convergence of sGaBP

This section is dedicated to proving the convergence of sGaBP given sufficient tuning. We start by deriving asymptotic expressions for $\ddot{\mathbf{P}}_i(\lambda)$ and $\ddot{\mathbf{P}}_{ij}(\lambda)$ as $\lambda \to \infty$. We then show that sGaBP converges to a point where the updates of the mean components become linear. The proof is completed by showing that the spectral radius of the linear update matrix approaches 1 from below as $\lambda \to \infty$.

4.2.1 Asymptotic Expressions

In this section we study the asymptotic behaviour of $\ddot{\mathbf{P}}_i(\lambda)$. Denote the number of columns (and rows) of $\mathbf{T}_{ii}^{(n)}(\lambda)$ by $m_{n;i}$. We start by defining the following:

1. Set $\mathbf{S} = [s_{ij}]$, $\mathbf{D} = \text{diag}(s_{11}, s_{22}, \ldots, s_{kk})$ and $\mathbf{R} = \mathbf{D} - \mathbf{S}$. Note that clustered subscripting is used for these matrices.

2. $\bar{\mathbf{S}}(\lambda) = (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}(\lambda\mathbf{I}_k + \mathbf{S})(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}$. Note that clustered subscripting is used for this matrix.

3. Set $\bar{\mathbf{R}}(\lambda) = \mathbf{I}_k - \bar{\mathbf{S}}(\lambda)$. This matrix has diagonal entries equal to zero and clustered subscripting is appropriate.

4. $\bar{\mathbf{T}}_{ii}^{(n)}(\lambda)$ and $\bar{\mathbf{T}}_{ij}^{(n)}(\lambda)$ are defined analogous to $\mathbf{T}_{ii}^{(n)}(\lambda)$ and $\mathbf{T}_{ij}^{(n)}(\lambda)$ respectively, however they are constructed from $\bar{\mathbf{S}}(\lambda)$.

5. $\bar{\mathcal{R}}_{ii}^{(n)}(\lambda) = \mathbf{I}_{m_{n;i}} - \bar{\mathbf{T}}_{ii}^{(n)}(\lambda)$. The matrix $\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)$ contains only zeros on its diagonal. Note that $\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)$ is the precision matrix of the UWT for cluster i, with a depth of n, constructed from the matrix $\bar{\mathbf{R}}(\lambda)$.

6. It can be shown that:

$$[\mathbf{G}_{ii}^{(n)}]'[\mathbf{T}_{ii}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ii}^{(n)} = (\lambda\mathbf{I}_{d_i}+\mathbf{D}_{ii})^{-0.5}[\mathbf{G}_{ii}^{(n)}]'[\bar{\mathbf{T}}_{ii}^{(n)}(\lambda)]^{-1}[\mathbf{G}_{ii}^{(n)}](\lambda\mathbf{I}_{d_i}+\mathbf{D}_{ii})^{-0.5}.$$

A consequence of Lemma 3 (see Appendix B) is that:

$$||\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)||_\infty \leq ||\bar{\mathbf{R}}(\lambda)||_\infty, \tag{4.6}$$

where equality will hold if the UWT is of sufficient depth such that all clusters are referenced before the terminal layer. A further consequence of Equation (4.6) is:

$$||[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t||_\infty \leq ||\bar{\mathbf{R}}(\lambda)||_\infty^t. \tag{4.7}$$

Consider:

$$\begin{aligned}
\bar{\mathbf{R}}(\lambda) &= \mathbf{I}_k - \bar{\mathbf{S}}(\lambda) \\
&= (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}(\lambda\mathbf{I}_k + \mathbf{D})(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5} \\
&\quad - (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}(\lambda\mathbf{I}_k + \mathbf{S})(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5} \\
&= (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}(\mathbf{D} - \mathbf{S})(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5} \\
&= (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}\mathbf{R}(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}.
\end{aligned} \tag{4.8}$$

From (4.8) we have:

$$\begin{aligned}
||\bar{\mathbf{R}}(\lambda)||_\infty &= ||(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}\mathbf{R}(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}||_\infty \\
&\leq ||(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}||_\infty||\mathbf{R}||_\infty||(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}||_\infty \\
&= \frac{||\mathbf{R}||_\infty}{\lambda + \min_l\{S_{ll}\}}
\end{aligned} \tag{4.9}$$

The bound in (4.9) shows that there will always be a selection of λ such that $\bar{\mathbf{S}}(\lambda)$ is strictly diagonally dominant. Furthermore, $\rho(\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)) \leq ||\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)||_\infty \leq$

$||\bar{\mathbf{R}}(\lambda)||_\infty \leq r_\lambda$, where $r_\lambda = \frac{||\mathbf{R}||_\infty}{\lambda + \min_l\{S_{ll}\}}$. From this point onwards we assume $\lambda > ||\mathbf{R}||_\infty - \min_l\{S_{ll}\}$ such that $r_\lambda < 1$. Hence, $\rho(\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)) < 1$, and we can apply the Neumann power series $[\mathbf{I}_{m_n} - \bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^{-1} = \sum_{t=0}^{\infty}[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t$ to obtain:

$$\begin{aligned}[\mathbf{G}_{ii}^{(n)}]'[\mathbf{I}_{m_n} - \bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ii}^{(n)} &= \sum_{t=0}^{\infty}[\mathbf{G}_{ii}^{(n)}]'[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t\mathbf{G}_{ii}^{(n)} \\ &= \mathbf{I}_{d_i} + \bar{\mathbf{R}}_{ii}(\lambda) + \sum_{l\in\{\mathcal{N}_i\cup i\}}\bar{\mathbf{R}}_{il}(\lambda)\bar{\mathbf{R}}_{li}(\lambda) + \mathbf{\Omega}_{ii}^{(n)}(\lambda),\end{aligned} \tag{4.10}$$

where $\mathbf{\Omega}_{ii}^{(n)}(\lambda) = \sum_{t=3}^{\infty}[\mathbf{G}_{ii}^{(n)}]'[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t\mathbf{G}_{ii}^{(n)}$.

We note three important consequences of (4.10):

1. Since $\bar{\mathbf{S}}(\lambda)$ is strictly diagonally dominant, $\lim_{n\to\infty}[\mathbf{G}_{ii}^{(n)}]'[\mathbf{I}_{m_n} - \bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ii}^{(n)}$ exists and therefore $\lim_{n\to\infty}\mathbf{\Omega}_{ii}^{(n)}(\lambda) = \mathbf{\Omega}_{ii}(\lambda)$ for a specified matrix $\mathbf{\Omega}_{ii}(\lambda)$.

2. We see that:

$$\begin{aligned}||\mathbf{\Omega}_{ii}^{(n)}(\lambda)||_\infty &\leq \sum_{t=3}^{\infty}||[\mathbf{G}_{ii}^{(n)}]'||_\infty||[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t||_\infty||\mathbf{G}_{ii}^{(n)}||_\infty \\ &= \sum_{t=3}^{\infty}||[\bar{\mathcal{R}}_{ii}^{(n)}(\lambda)]^t||_\infty \\ &\leq \sum_{t=3}^{\infty} r_\lambda^t \\ &= \frac{r_\lambda^3}{1-r_\lambda} = \mathcal{O}(\frac{1}{\lambda^3})\end{aligned}$$

3. Points (1) and (2) guarantee that $\mathbf{\Omega}_{ii}(\lambda) = \mathcal{O}(\frac{1}{\lambda^2})$ (see Lemma 8 in Appendix D).

Consider further simplification of (4.8). We note that $(\mathbf{I}_k + \frac{\mathbf{D}}{\lambda})^{-0.5}$ is a diagonal matrix with the lth diagonal entry equal to:

$$\left(1 + \frac{S_{ll}}{\lambda}\right)^{-0.5} = 1 + \sum_{t=1}^{\infty}\binom{-0.5}{t}\frac{S_{ll}^t}{\lambda^t} = 1 + \mathcal{O}(\frac{1}{\lambda}), \tag{4.11}$$

where we assume that $\lambda > S_{ll}$ and $\binom{\alpha}{t} = \frac{\alpha(\alpha-1)\ldots(\alpha-t+1)}{t!}$ denotes the generalised binomial coefficients. As a consequence we have:

$$(\mathbf{I}_k + \frac{\mathbf{D}}{\lambda})^{-0.5} = \mathbf{I}_k + \mathcal{O}(\frac{1}{\lambda}), \tag{4.12}$$

and

$$\begin{aligned}\bar{\mathbf{R}}(\lambda) &= (\lambda\mathbf{I}_k + \mathbf{D})^{-0.5}\mathbf{R}(\lambda\mathbf{I}_k + \mathbf{D})^{-0.5} \\ &= \frac{1}{\lambda}(\mathbf{I}_k + \frac{\mathbf{D}}{\lambda})^{-0.5}\mathbf{R}(\mathbf{I}_k + \frac{\mathbf{D}}{\lambda})^{-0.5} \\ &= \frac{1}{\lambda}(\mathbf{I}_k + \mathcal{O}(\frac{1}{\lambda}))\mathbf{R}(\mathbf{I}_k + \mathcal{O}(\frac{1}{\lambda})) \\ &= \frac{1}{\lambda}\mathbf{R} + \mathcal{O}(\frac{1}{\lambda^2}). \end{aligned} \tag{4.13}$$

From (4.13) we see that:

$$\mathbf{I}_{d_i} + \bar{\mathbf{R}}_{ii}(\lambda) + \sum_{l\in\{\mathcal{N}_i\cup i\}} \bar{\mathbf{R}}_{il}(\lambda)\bar{\mathbf{R}}_{li}(\lambda) = \mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii} + \mathcal{O}(\frac{1}{\lambda^2}), \tag{4.14}$$

where no terms involve the iteration number n. Consider:

$$\begin{aligned}\lambda\ddot{\mathbf{P}}_i^{(n-1)}(\lambda) &= \lambda[\mathbf{G}_{ii}^{(n)}]'[\mathbf{T}_{ii}^{(n)}(\lambda)]^{-1}\mathbf{G}_{ii}^{(n)} \\ &= \lambda(\lambda\mathbf{I}_{d_i} + \mathbf{D}_{ii})^{-0.5}[\mathbf{G}_{ii}^{(n)}]'[\bar{\mathbf{T}}_{ii}^{(n)}(\lambda)]^{-1}[\mathbf{G}_{ii}^{(n)}](\lambda\mathbf{I}_{d_i} + \mathbf{D}_{ii})^{-0.5} \\ &= \lambda(\lambda\mathbf{I}_{d_i} + \mathbf{D}_{ii})^{-0.5}\left[\mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii} + \mathcal{O}(\frac{1}{\lambda^2}) + \mathbf{\Omega}_{ii}^{(n)}(\lambda)\right](\lambda\mathbf{I}_{d_i} + \mathbf{D}_{ii})^{-0.5} \\ &\to \left(\mathbf{I}_{d_i} + \frac{\mathbf{D}_{ii}}{\lambda}\right)^{-0.5}\left[\mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii} + \mathcal{O}(\frac{1}{\lambda^2})\right]\left(\mathbf{I}_{d_i} + \frac{\mathbf{D}_{ii}}{\lambda}\right)^{-0.5}, \end{aligned} \tag{4.15}$$

as $n \to \infty$ and were it should be noted that $\mathbf{\Omega}_{ii}(\lambda) = \mathcal{O}(\frac{1}{\lambda^2})$. An extension of (4.11) yields:

$$\left(1 + \frac{S_{ll}}{\lambda}\right)^{-0.5} = 1 - 0.5\frac{S_{ll}}{\lambda} + \sum_{t=2}^{\infty}\binom{-0.5}{t}\frac{S_{ll}^t}{\lambda^t} = 1 - 0.5\frac{S_{ll}}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2}), \tag{4.16}$$

which gives $\left(\mathbf{I}_{d_i} + \frac{\mathbf{D}_{ii}}{\lambda}\right)^{-0.5} = \mathbf{I}_{d_i} - 0.5\frac{\mathbf{D}_{ii}}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2})$. Equation (4.15) becomes:

$$\begin{aligned}&\left(\mathbf{I}_{d_i} + \frac{\mathbf{D}_{ii}}{\lambda}\right)^{-0.5}\left[\mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii} + \mathcal{O}(\frac{1}{\lambda^2})\right]\left(\mathbf{I}_{d_i} + \frac{\mathbf{D}_{ii}}{\lambda}\right)^{-0.5} \\ &= \left(\mathbf{I}_{d_i} - 0.5\frac{\mathbf{D}_{ii}}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2})\right)\left[\mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii} + \mathcal{O}(\frac{1}{\lambda^2})\right]\left(\mathbf{I}_{d_i} - 0.5\frac{\mathbf{D}_{ii}}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2})\right) \\ &= \left(\mathbf{I}_{d_i} - 0.5\frac{\mathbf{D}_{ii}}{\lambda}\right)\left[\mathbf{I}_{d_i} + \frac{1}{\lambda}\mathbf{R}_{ii}\right]\left(\mathbf{I}_{d_i} - 0.5\frac{\mathbf{D}_{ii}}{\lambda}\right) + \mathcal{O}(\frac{1}{\lambda^2}) \\ &= \mathbf{I}_{d_i} + \frac{1}{\lambda}(\mathbf{R}_{ii} - \mathbf{D}_{ii}) + \mathcal{O}(\frac{1}{\lambda^2}) \\ &= \mathbf{I}_{d_i} - \frac{1}{\lambda}\mathbf{S}_{ii} + \mathcal{O}(\frac{1}{\lambda^2}). \end{aligned} \tag{4.17}$$

Finally we obtain:

$$\lambda\ddot{\mathbf{P}}_i(\lambda) = \lim_{n\to\infty} \lambda\ddot{\mathbf{P}}_i^{(n-1)}(\lambda) = \mathbf{I}_{d_i} - \frac{1}{\lambda}\mathbf{S}_{ii} + \mathcal{O}(\frac{1}{\lambda^2}). \tag{4.18}$$

Similar to the derivation of (4.18) we can show:

$$\ddot{\mathbf{P}}_{ij}(\lambda) = \frac{1}{\lambda}\mathbf{I}_{d_i} + \mathcal{O}(\frac{1}{\lambda^2}). \tag{4.19}$$

Defining $\mathbf{U}_{ij}(\lambda) = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}(\lambda)$ we see that:

$$\mathbf{U}_{ij}(\lambda) = -\frac{1}{\lambda}\mathbf{S}_{ji} + \mathcal{O}(\frac{1}{\lambda^2}). \tag{4.20}$$

In the next section, we show that, after the precision components of sGaBP have converged, the updates of the potential vector and mean components become linear. The linear update matrix is determined by the matrices $\mathbf{U}_{ij}(\lambda)$ and $\lambda\ddot{\mathbf{P}}_i(\lambda)$ for $i \in \mathcal{V}$ and $j \in \mathcal{N}_i$, and hence they play a crucial role in the convergence behaviour of sGaBP. Of particular interest is the behaviour of the spectral radius of the linear update matrix as $\lambda \to \infty$. The asymptotic expressions derived in this section allow us to show that this spectral radius will be less than 1 for large enough λ.

4.2.2 Convergence to Linear Updates

In the previous section we showed that if λ is sufficiently large, then $\ddot{\mathbf{P}}_{ij}^{(n)}(\lambda) \to \ddot{\mathbf{P}}_{ij}(\lambda)$ and $\ddot{\mathbf{P}}_i^{(n)}(\lambda) \to \ddot{\mathbf{P}}_i(\lambda)$ as $n \to \infty$. For the remainder of Section 4.2 we will write:

$$\ddot{\mathbf{P}}_i = \ddot{\mathbf{P}}_i(\lambda) \tag{4.21}$$

$$\mathbf{U}_{ij} = \mathbf{U}_{ij}(\lambda). \tag{4.22}$$

As in Chapter 2, we assume that this convergence has occurred. Let

$$\mathbf{v}_{ij}^{(n+1)} = \mathbf{U}_{ij}[\lambda\boldsymbol{\mu}_i^{(n-1)} + \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i\setminus j} \mathbf{v}_{ti}^{(n)}] \tag{4.23}$$

$$\boldsymbol{\mu}_i^{(n+1)} = \ddot{\mathbf{P}}_i[\lambda\boldsymbol{\mu}_i^{(n)} + \boldsymbol{b}_i + \sum_{t\in\mathcal{N}_i} \mathbf{v}_{ti}^{(n+1)}]. \tag{4.24}$$

We now show that these updates can be done through a linear transformation matrix. Define

$$\boldsymbol{\gamma}_i^{(n+1)} = (\mathbf{v}_{1i}^{(n+1)'}, \mathbf{v}_{2i}^{(n+1)'}, \ldots, \mathbf{v}_{(i-1);i}^{(n+1)'}, \mathbf{v}_{(i+1);i}^{(n+1)'}, \ldots, \mathbf{v}_{pi}^{(n+1)'})', \tag{4.25}$$

and set $\boldsymbol{\gamma}^{(n+1)} = (\boldsymbol{\gamma}_1^{(n+1)'}, \boldsymbol{\gamma}_2^{(n+1)'}, \ldots, \boldsymbol{\gamma}_p^{(n+1)'}, \boldsymbol{\mu}_1^{(n)'}, \ldots, \boldsymbol{\mu}_p^{(n)'})'$. Note that the size of $\boldsymbol{\gamma}_i^{(n+1)}$ is $(p-1)d_i$. Set $m_1 = (p-1)\sum_i d_i = k(p-1)$ and $m_2 = m_1 + k = kp$. We are going to show that there is a matrix $\mathbf{L} : m_2 \times m_2$, such that:

$$\boldsymbol{\gamma}^{(n+1)} = \boldsymbol{\gamma}_0 + \mathbf{L}\boldsymbol{\gamma}^{(n)}, \tag{4.26}$$

where $\boldsymbol{\gamma}_0$ is an $m_2 \times 1$ vector. Consider $\mathbf{U}_{ij} : d_j \times d_i$ and set

$$\boldsymbol{\gamma}_{0i} = (\boldsymbol{b}_1'\mathbf{U}_{1i}', \boldsymbol{b}_2'\mathbf{U}_{2i}', \ldots, \boldsymbol{b}_{i-1}'\mathbf{U}_{(i-1);i}', \boldsymbol{b}_{i+1}'\mathbf{U}_{(i+1);i}', \ldots, \boldsymbol{b}_p'\mathbf{U}_{pi}')'. \tag{4.27}$$

Let $\boldsymbol{\gamma}_0 = (\boldsymbol{\gamma}_{01}', \boldsymbol{\gamma}_{02}', \ldots, \boldsymbol{\gamma}_{0p}', \boldsymbol{b}_1'\ddot{\mathbf{P}}_1', \ldots, \boldsymbol{b}_p'\ddot{\mathbf{P}}_p')'$. For the linear update matrix, consider the decomposition

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_{11} : m_1 \times m_1 & \mathbf{L}_{12} : m_1 \times k \\ \mathbf{L}_{21} : k \times m_1 & \mathbf{L}_{22} : k \times k \end{bmatrix}. \tag{4.28}$$

The construction of $\mathbf{L}$ is as follows:

1. Consider first the matrix $\mathbf{L}_{11}$. We can decompose $\mathbf{L}_{11}$ into blocks, where each block corresponds to a row message and a column message. Consider block s, t of $\mathbf{L}_{11}$ and assign to this block a row index and column index, which are to be obtained from the first m_1 components of $\boldsymbol{\gamma}_0$. To obtain the row and column message indices of this block, we move to entry s and entry t of $\boldsymbol{\gamma}_0$. If entry s is $\mathbf{U}_{ji}\boldsymbol{b}_j$ and entry t is $\mathbf{U}_{ru}\boldsymbol{b}_r$, then the row and column indices of block s, t are (j, i) (message from j to i) and (r, u) (message from r to u) respectively.

2. Consider block s, t of $\mathbf{L}_{11}$ with row indices (j, i) and column indices (r, u). If $u = j$ and $r \in \mathcal{N}_j \setminus i$, then this block is $\mathbf{U}_{ji}$, otherwise the block is a matrix of zeros.

3. The matrix $\mathbf{L}_{22}$ has a block decomposition according to the last k components of γ_0. Block s, t is associated with $\boldsymbol{b}_s$ (row index is s) and $\boldsymbol{b}_t$ (column index is t). $\mathbf{L}_{22}$ is a block diagonal matrix with diagonal block i (row and column index is i) equal to $\lambda\ddot{\mathbf{P}}_i$.

4. The matrix $\mathbf{L}_{12}$ has a decomposition according to the row indices of $\mathbf{L}_{11}$ and the column indices of $\mathbf{L}_{22}$. Block s, t has a row index (j, i) and a column index u. This block is $\lambda\mathbf{U}_{ji}$ if $u = j$, and a matrix of zeros otherwise.

5. $\mathbf{L}_{21}$ has a block decomposition with row indices equal to the row indices of $\mathbf{L}_{22}$ and the column indices of $\mathbf{L}_{11}$. Block s, t has a row index u and a column index (j, i). This block is $\ddot{\mathbf{P}}_u$ if $i = u$ and $j \in \mathcal{N}_u$, otherwise it is a matrix of zeros.

Let us look at an example for $p = 3$. After convergence of the precision components, we have the matrices $\ddot{\mathbf{P}}_i : i = 1, 2, 3$ and the matrix

$$\begin{bmatrix} & \mathbf{U}_{12} & \mathbf{U}_{13} \\ \mathbf{U}_{21} & & \mathbf{U}_{23} \\ \mathbf{U}_{31} & \mathbf{U}_{32} & \end{bmatrix}, \tag{4.29}$$

where the diagonal blocks are irrelevant. The linear update matrix, with the row and column indices as discussed, is

$$\mathbf{L} = \begin{array}{c} \\ 2\to 1 \\ 3\to 1 \\ 1\to 2 \\ 3\to 2 \\ 1\to 3 \\ 2\to 3 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 2\to 1 & 3\to 1 & 1\to 2 & 3\to 2 & 1\to 3 & 2\to 3 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{21} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{31} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{31} \\ \mathbf{0} & \mathbf{U}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{12} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{U}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{32} \\ \mathbf{U}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{13} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{U}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\mathbf{U}_{23} & \mathbf{0} \\ \ddot{\mathbf{P}}_1 & \ddot{\mathbf{P}}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\ddot{\mathbf{P}}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddot{\mathbf{P}}_2 & \ddot{\mathbf{P}}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda\ddot{\mathbf{P}}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddot{\mathbf{P}}_3 & \ddot{\mathbf{P}}_3 & \mathbf{0} & \mathbf{0} & \lambda\ddot{\mathbf{P}}_3 \end{bmatrix} \end{array}. \tag{4.30}$$

Returning to the general case, the vector $\boldsymbol{\gamma}^{(n+1)} \to (\mathbf{I}-\mathbf{L})^{-1}\boldsymbol{\gamma}_0$ as $n \to \infty$ if, and only if, $\rho(\mathbf{L}) < 1$.

Let $\boldsymbol{\Delta} = \text{diag}(\mathbf{1}_{m_1}, \lambda\mathbf{1}_k)$, where $\mathbf{1}_i$ is an $i \times 1$ vector containing only ones. To study the spectral radius of $\mathbf{L}$ we consider the following matrix:

$$\tilde{\mathbf{L}} = \boldsymbol{\Delta}\mathbf{L}\boldsymbol{\Delta}^{-1} = \begin{bmatrix} \tilde{\mathbf{L}}_{11} : m_1 \times m_1 & \tilde{\mathbf{L}}_{12} : m_1 \times k \\ \tilde{\mathbf{L}}_{21} : k \times m_1 & \tilde{\mathbf{L}}_{22} : k \times k \end{bmatrix}, \tag{4.31}$$

where $\lambda > 0$, $\tilde{\mathbf{L}}_{11} = \mathbf{L}_{11}$, $\tilde{\mathbf{L}}_{12} = \frac{1}{\lambda}\mathbf{L}_{12}$, $\tilde{\mathbf{L}}_{21} = \lambda\mathbf{L}_{21}$ and $\tilde{\mathbf{L}}_{22} = \mathbf{L}_{22}$. Define $\mathbf{L}_{st}$ as the block of $\mathbf{L}$ corresponding to (s,t). A consequence of this scaling is that all non-zero blocks in a given row block are identical.

Our objective is to show that the spectral radius of $\tilde{\mathbf{L}}$ is less than 1 when λ is sufficiently large (this proves convergence of sGaBP). To this end, we consider the asymptotic behaviour of $\tilde{\mathbf{L}}$ as $\lambda \to \infty$ in the next section.

4.2.3 The Asymptotic Linear Update Matrix

Consider a matrix $\mathbf{H}$ defined as

$$\mathbf{H} : m_1 \times k = \begin{array}{c} \begin{array}{cccc} 1 & 2 & \dots & \mathrm{p} \end{array} \\ \begin{bmatrix} \mathbf{I}_{d_1} & \dots & \dots & \dots \\ \mathbf{I}_{d_1} & \dots & \dots & \dots \\ \vdots & \dots & \dots & \dots \\ \mathbf{I}_{d_1} & \dots & \dots & \dots \\ \dots & \mathbf{I}_{d_2} & \dots & \dots \\ \dots & \mathbf{I}_{d_2} & \dots & \dots \\ \dots & \vdots & \dots & \dots \\ \dots & \mathbf{I}_{d_2} & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & \mathbf{I}_{d_p} \\ \dots & \dots & \dots & \mathbf{I}_{d_p} \\ \dots & \dots & \dots & \mathbf{I}_{d_p} \\ \dots & \dots & \dots & \mathbf{I}_{d_p} \end{bmatrix} \end{array}, \tag{4.32}$$

where each column block (indexed by i) contains exactly $p-1$ $\mathbf{I}_{d_i}$ matrices stacked in consecutive rows; the remainder of the column block contains zeros. The way the stacking is done is clearly shown in Equation (4.32), where the $\dots$ and $\vdots$ indicate that the corresponding part of the matrix is filled with zeros. For our example we have

$$\mathbf{H} = \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} \end{bmatrix} \end{array}, \tag{4.33}$$

where each $\mathbf{0}$ is of a suitable dimension. We now have the following expressions:

$$\tilde{\mathbf{L}}_{12} = \frac{1}{p-2}\tilde{\mathbf{L}}_{11}\mathbf{H} \tag{4.34}$$

$$\tilde{\mathbf{L}}_{21} = \tilde{\mathbf{L}}_{22}\mathbf{H}'. \tag{4.35}$$

Set $\delta = \frac{1}{\lambda}$ and consider the following matrices:

1. $\mathbf{M}_{11}$ has an identical construction to $\mathbf{L}_{11}$, using $-\mathbf{S}_{ji}$ instead of $\mathbf{U}_{ij}$.
2. $\mathbf{M}_{22}$ has an identical construction to $\mathbf{L}_{22}$, using $-\mathbf{S}_{ii}$ instead of $\lambda\ddot{\mathbf{P}}_i$.

We have the following asymptotic expression for the linear update matrix:

$$\mathbf{L} = \begin{bmatrix} \delta\mathbf{M}_{11} & \frac{\delta}{p-2}\mathbf{M}_{11}\mathbf{H} \\ (\mathbf{I}_k - \delta\mathbf{M}_{22})\mathbf{H}' & (\mathbf{I}_k - \delta\mathbf{M}_{22}) \end{bmatrix} + \mathcal{O}(\delta^2). \tag{4.36}$$

The behaviour of the spectral radius of $\mathbf{L}$ as $\lambda \to \infty$ is given in the following theorem. A proof is contained in Appendix C.

Theorem 13 *Consider applying sGaBP on a precision matrix $\mathbf{S} : k \times k$ and a potential vector $\boldsymbol{b} : k \times 1$, where variables are assigned to nodes according to clusters $\mathcal{C}_i : i = 1, 2, \ldots, p$. Let $\mathbf{L}$ be the linear update matrix obtained after the precision components have converged. The following holds:*

$$\rho(\mathbf{L}) = 1 - \frac{\sigma_1}{\lambda} + \mathcal{O}(\frac{1}{\lambda^2}), \tag{4.37}$$

where σ_1 is the smallest eigenvalue of $\mathbf{S}$.

Theorem 13 shows that the spectral radius of the linear update matrix will be less than one given sufficiently large λ and, for this selection of the degree of regularisation, sGaBP will converge.

4.3 Adaptive Damping and Inference Quality

An important aspect of the sGaBP algorithm is the adjustment to the means performed at each iteration. Recalling that $\mathbf{z}_i^{(n)} = \sum_{t \in \mathcal{N}_i} \mathbf{v}_{ti}^{(n)}$ and $\hat{\mathbf{P}}_i^{(n)} = \mathbf{P}_i^{(n)}(\lambda) - \lambda\mathbf{I}_{d_i}$, the posterior means are damped through the following mechanism:

$$\begin{aligned} \boldsymbol{\mu}_i^{(n)} &= [\mathbf{P}_i^{(n)}(\lambda)]^{-1}[\lambda\boldsymbol{\mu}_i^{(n-1)} + \mathbf{z}_i^{(n)}] \\ &= \lambda[\mathbf{P}_i^{(n)}(\lambda)]^{-1}\boldsymbol{\mu}_i^{(n-1)} + [\mathbf{P}_i^{(n)}(\lambda)]^{-1}\mathbf{z}_i^{(n)} \\ &= \lambda[\mathbf{P}_i^{(n)}(\lambda)]^{-1}\boldsymbol{\mu}_i^{(n-1)} + [\mathbf{P}_i^{(n)}(\lambda)]^{-1}\hat{\mathbf{P}}_i^{(n)}[\hat{\mathbf{P}}_i^{(n)}]^{-1}\mathbf{z}_i^{(n)}. \end{aligned}$$

Note that $\mathbf{P}_i^{(n)}(\lambda) = \lambda\mathbf{I}_{d_i} + \hat{\mathbf{P}}_i^{(n)}$. If we set $\mathbf{S}_i^{(n)}(\lambda) = \lambda[\mathbf{P}_i^{(n)}(\lambda)]^{-1}$, then

$$\boldsymbol{\mu}_i^{(n)} = \mathbf{S}_i^{(n)}(\lambda)\boldsymbol{\mu}_i^{(n-1)} + (\mathbf{I}_{d_i} - \mathbf{S}_i^{(n)}(\lambda))[\hat{\mathbf{P}}_i^{(n)}]^{-1}\mathbf{z}_i^{(n)}. \tag{4.38}$$

We can interpret $[\hat{\mathbf{P}}_i^{(n)}]^{-1}\mathbf{z}_i^{(n)}$ as the posterior mean for iteration n that we would have computed if no damping was applied. Hence the posterior mean at iteration n can be interpreted as a combination of the posterior mean of the previous iteration and the mean suggested by the current messages. The damping is done through a matrix $\mathbf{S}_i^{(n)}(\lambda)$, which depends on λ and the current posterior precision. In contrast to methods such as RGaBP, sGaBP automatically computes damping matrices based on the regularisation parameter λ. This damping is essential to preserve the exactness of the converged posterior means as the true marginal means. We summarise this result in Theorem 14, which is proved in Appendix C.

Theorem 14 *Suppose the iterative updates given in (4.1) - (4.3) have converged to a fixed point. The converged posterior means solve the linear system of equations* $\mathbf{Sx} = \boldsymbol{b}$.

We analyse the accuracy of the posterior precisions as approximations to the true marginal precisions in the empirical section (Section 4.5) by selecting the value of the regularisation parameter yielding the fastest convergence (this value of the regularisation parameter is determined empirically). As we will show, sGaBP provides more accurate precision estimates than certain other competitors, and seems to improve inference quality overall.

4.4 Heuristic Regularisation

This section is dedicated to constructing a heuristic measure for the selection of the degree of regularisation. The heuristic is based on a recursive formula for the posterior means of sGaBP. The principle behind this formula is similar to the UWT formula for the posterior means of unregularised GaBP, in the sense that we propose determining the posterior means as a matrix-inversion problem involving a matrix based on a tree topology.

4.4.1 Tree Representation of sGaBP

Assume we want to use the UWT for node i to determine an analytical formula for $\boldsymbol{\mu}_i^{(n)}(\lambda)$, i.e. the posterior mean associated with cluster i after n iterations where the dependence on λ is emphasised. In order to do this, we need to adjust the way in which we assign a precision matrix and potential vector to the UWT, compared to the method of Chapter 3.

For the precision matrix associated with the UWT, we assign to a node with reference to cluster j the precision matrix $\lambda\mathbf{I}_{d_j} + \mathbf{S}_{jj}$. The precision matrices between nodes in the UWT remains as in Chapter 3. The precision matrix assigned to the UWT for cluster i is $\mathbf{T}_{ii}^{(n)}(\lambda)$ (precision matrix of UWT constructed for $\mathbf{S}(\lambda)$).

To assign a potential to a node in the UWT, we require the cluster reference of the node and the layer number in which it occurs. In addition, we require the history of the posterior means, that is $\boldsymbol{\mu}^{(l)}(\lambda)$ (posterior mean at iteration l) for all $l < n$. Consider a node in layer l with a reference to cluster j. To this node we assign the potential $\boldsymbol{b}_j + \lambda\boldsymbol{\mu}_j^{(n-l-1)}(\lambda)$. We do this with the understanding that $\boldsymbol{\mu}_j^{(-1)}(\lambda)$ is an initial value for the posterior mean associated with cluster i. In our application of sGaBP we use $\boldsymbol{\mu}_j^{(-1)}(\lambda) = \boldsymbol{b}_j$.

If we marginalise the UWT with the above precision matrix and potential

vector and extract the marginal mean at the root node, we obtain $\boldsymbol{\mu}_i^{(n)}(\lambda)$. An illustration of this procedure is given in Appendix D. In the next section, we introduce matrix notation for the tree representation of sGaBP.

4.4.2 Matrix Notation

Let $\mathbf{E}_{ii}^{(n)}$ be defined analogous to $\mathbf{E}_n$ in Chapter 3, for root node i and an UWT depth of n. Let the rows of $\mathbf{E}_{ii}^{(n)}$ be decomposed according to the different layers of the UWT:

$$\mathbf{E}_{ii}^{(n)} = \begin{bmatrix} \tilde{\mathbf{F}}_{1i} \\ \tilde{\mathbf{F}}_{2i} \\ \vdots \\ \tilde{\mathbf{F}}_{ni} \end{bmatrix}, \tag{4.39}$$

where $\tilde{\mathbf{F}}_{mi}$ contains all the row-extractor matrices of the nodes in layer m of the UWT for node i. If $\lambda = 0$, then we would have assigned $\mathbf{E}_{ii}^{(n)}\boldsymbol{b}$ to the potential of the UWT. In the $\lambda \neq 0$ case, different layers of the UWT correspond to different posterior means. To allow for this, we define the matrix

$$\mathbf{J}_{ii}^{(n)} = \begin{bmatrix} \tilde{\mathbf{F}}_{1i} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{F}}_{2i} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \tilde{\mathbf{F}}_{ni} \end{bmatrix} \tag{4.40}$$

and vector

$$\boldsymbol{\phi}_{n-1}(\lambda) = (\boldsymbol{\mu}^{(n-2)}(\lambda)', \boldsymbol{\mu}^{(n-3)}(\lambda)', \dots, \boldsymbol{\mu}^{(0)}(\lambda)', \boldsymbol{\mu}^{(-1)}(\lambda)')', \tag{4.41}$$

where $\boldsymbol{\mu}^{(-1)}(\lambda) = (1+\lambda)\boldsymbol{b}$. The potential assigned to the UWT becomes $\mathbf{E}_{ii}^{(n)}\boldsymbol{b} + \lambda\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1}(\lambda)$. Consider a node in layer l with a reference to cluster j. We state the following formula (it should be noted that this formula was validated empirically):

$$\boldsymbol{\mu}_i^{(n-1)}(\lambda) = (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda))^{-1}[\mathbf{E}_{ii}^{(n)}\boldsymbol{b} + \lambda\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1}(\lambda)]. \tag{4.42}$$

In the next section, we use the formula given in Equation (4.42) to derive a recursive representation of the posterior means.

4.4.3 Recursive Representation of the Posterior Means

Let $\mathbf{V}_n(\lambda) : k \times k$ and $\mathbf{W}_n(\lambda) : k \times nk$ be defined as:

$$\mathbf{V}_n(\lambda) = \begin{bmatrix} (\mathbf{G}_{11}^{(n)})'(\mathbf{T}_{11}^{(n)}(\lambda))^{-1}\mathbf{E}_{11}^{(n)} \\ (\mathbf{G}_{22}^{(n)})'(\mathbf{T}_{22}^{(n)}(\lambda))^{-1}\mathbf{E}_{22}^{(n)} \\ \vdots \\ (\mathbf{G}_{pp}^{(n)})'(\mathbf{T}_{pp}^{(n)}(\lambda))^{-1}\mathbf{E}_{pp}^{(n)} \end{bmatrix} \tag{4.43}$$

$$\mathbf{W}_n(\lambda) = \begin{bmatrix} (\mathbf{G}_{11}^{(n)})'(\mathbf{T}_{11}^{(n)}(\lambda))^{-1}\mathbf{J}_{11}^{(n)} \\ (\mathbf{G}_{22}^{(n)})'(\mathbf{T}_{22}^{(n)}(\lambda))^{-1}\mathbf{J}_{22}^{(n)} \\ \vdots \\ (\mathbf{G}_{pp}^{(n)})'(\mathbf{T}_{pp}^{(n)}(\lambda))^{-1}\mathbf{J}_{pp}^{(n)} \end{bmatrix}. \tag{4.44}$$

Equation (4.42) implies

$$\boldsymbol{\mu}^{(n-1)}(\lambda) = \mathbf{V}_n(\lambda)\boldsymbol{b} + \lambda\mathbf{W}_n(\lambda)\boldsymbol{\phi}_{n-1}(\lambda). \tag{4.45}$$

Define the following matrices,

$$\tilde{\mathbf{V}}_n(\lambda) : (n+1)k \times k = \begin{bmatrix} \mathbf{V}_n(\lambda) \\ \mathbf{0} : nk \times k \end{bmatrix} \tag{4.46}$$

$$\tilde{\mathbf{W}}_n(\lambda) = \begin{bmatrix} \mathbf{W}_n(\lambda) \\ \mathbf{I}_{kn} \end{bmatrix}, \tag{4.47}$$

which gives:

$$\boldsymbol{\phi}_n(\lambda) = \tilde{\mathbf{V}}_n(\lambda)\boldsymbol{b} + \lambda\tilde{\mathbf{W}}_n(\lambda)\boldsymbol{\phi}_{n-1}(\lambda). \tag{4.48}$$

Equation (4.48) can be used to obtain a recursive formula for $\boldsymbol{\mu}^{(n)}(\lambda)$ and can, in principle, be used to derive convergence conditions for sGaBP. However, this recursive formula is complicated due to the varying nature of $\tilde{\mathbf{V}}_n(\lambda)$ and $\tilde{\mathbf{W}}_n(\lambda)$ (in terms of varying dimensionality and its dependence on n), and we leave the study of this formula for further research. The results of this section can be useful in the derivation of heuristics for the selection of λ, which we discuss in the next section.

4.4.4 Heuristic Regularisation

In this section, we make an adjustment to the precision matrix of the unwrapped tree as discussed in Section 4.4.1. Instead of using a single tuning parameter for all blocks in the line topology, we vary λ from layer to layer (we call this a varying UWT). For the precision matrix associated with the varying UWT, we assign to a node, in layer l, with reference to cluster j the precision matrix $\lambda^{(n-l)}\mathbf{I}_{d_j} + \mathbf{S}_{jj}$. We call this precision matrix $\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)})$, where we only emphasise its dependence on $\lambda^{(n-1)}$ ($\lambda^{(n-l)}$ for $l \geq 2$ is assumed

to be fixed). The potential associated with a node, with reference to cluster j, is $\boldsymbol{b}_j + \lambda^{(n-l)}\boldsymbol{\mu}_j^{(n-l-1)}$. Note that we assume $\boldsymbol{\mu}_j^{(n-l)}$ to be fixed for $l \geq 2$ and hence no dependence on any regularisation is indicated. The precision matrix between nodes remains as in Chapter 3 and Section 4.4.1.

We now discuss a heuristic aimed at varying the regularisation between layers such that convergence is achieved at a faster rate. Let $\mathcal{D}_{ni}(\lambda^{(n-1)})$ denote a diagonal matrix in which the diagonal entries corresponding to layer l of the varying UWT are $\lambda^{(n-l)}$. Similar to Equation (4.42), we see that:

$$\boldsymbol{\mu}_i^{(n-1)}(\lambda^{(n-1)}) = (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}[\mathbf{E}_{ii}^{(n)}\boldsymbol{b} + \mathcal{D}_{ni}(\lambda^{(n-1)})\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1}], \quad (4.49)$$

where $\boldsymbol{\phi}_{n-1}$ contains all the posterior means until stage $n-2$ (similar to Equation (4.41)). We note that Equation (4.49) was validated empirically. Since we have completed the updates until stage $n-2$, the entries of $\boldsymbol{\phi}_{n-1}$ will be fixed. Furthermore,

$$\frac{\partial \mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)})}{\partial \lambda^{(n-1)}} = \frac{\partial \mathcal{D}_{ni}(\lambda^{(n-1)})}{\partial \lambda^{(n-1)}} = \mathbf{G}_{ii}^{(n)}(\mathbf{G}_{ii}^{(n)})' \quad (4.50)$$

$$\begin{aligned}\frac{\partial (\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}}{\partial \lambda^{(n-1)}} &= -(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\frac{\partial \mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)})}{\partial \lambda^{(n-1)}}(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1} \\ &= -(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}(\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}. \quad (4.51)\end{aligned}$$

Differentiating (4.49) with respect to $\lambda^{(n-1)}$, we obtain

$$\begin{aligned}\frac{\partial \boldsymbol{\mu}_i^{(n-1)}(\lambda^{(n-1)})}{\partial \lambda^{(n-1)}} &= -(\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}(\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{E}_{ii}^{(n)}\boldsymbol{b} \\ &+ (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}(\mathbf{G}_{ii}^{(n)})'\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1} \\ &- (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}(\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathcal{D}_{ni}(\lambda^{(n-1)})\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1} \\ &= (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}\Big[(\mathbf{G}_{ii}^{(n)})'\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1} \\ &- (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}(\mathbf{E}_{ii}^{(n)}\boldsymbol{b} + \mathcal{D}_{ni}(\lambda^{(n-1)})\mathbf{J}_{ii}^{(n)}\boldsymbol{\phi}_{n-1})\Big].\end{aligned}$$

Since $(\mathbf{G}_{ii}^{(n)})'\mathbf{J}_{ni}\boldsymbol{\phi}_{n-1} = \boldsymbol{\mu}_i^{(n-2)}$ (which is fixed), we have

$$\begin{aligned}\frac{\partial \boldsymbol{\mu}_i^{(n-1)}(\lambda^{(n-1)})}{\partial \lambda^{(n-1)}} &= (\mathbf{G}_{ii}^{(n)})'(\mathbf{T}_{ii}^{(n)}(\lambda^{(n-1)}))^{-1}\mathbf{G}_{ii}^{(n)}\Big[\boldsymbol{\mu}_i^{(n-2)} - \boldsymbol{\mu}_i^{(n-1)}(\lambda^{(n-1)})\Big] \\ &= (\mathbf{P}_i^{(n-1)}(\lambda^{(n-1)}))^{-1}\Big[\boldsymbol{\mu}_i^{(n-2)} - \boldsymbol{\mu}_i^{(n-1)}(\lambda^{(n-1)})\Big], \quad (4.52)\end{aligned}$$

where $\mathbf{P}_i^{(n-1)}(\lambda^{(n-1)})$ indicates the posterior precision at iteration $n-1$, and this only depends on $\lambda^{(n-1)}$. We can obtain the derivative of $\boldsymbol{\mu}^{(n)}(\lambda^{(n)})$ with

Algorithm 7 Synchronous RGaBP

1. Specify a tolerance ϵ, a maximum number of iterations m and a relaxation parameter τ.
2. Initialise $\mathbf{Q}_{ij}^{(0)} = \mathbf{0} : d_j \times d_j$, $\mathbf{v}_{ij}^{(0)} = \mathbf{0} : d_j \times 1$, $\boldsymbol{\mu}^{(-1)} = \boldsymbol{b}$ for all i and all $j \in \mathcal{N}_i$.
3. Set Err = Inf and $n = 0$.
4. While Err $> \epsilon$
   a) Compute $\mathbf{P}_i^{(n)}(0) = \mathbf{S}_{ii} + \sum_{j \in \mathcal{N}_i} \mathbf{Q}_{ji}^{(n)}$ and $\mathbf{z}_i^{(n)} = \boldsymbol{b}_i + \sum_{j \in \mathcal{N}_i} \mathbf{v}_{ji}^{(n)}$ for $i = 1, 2, \ldots, p$.
   b) Update $\mathbf{z}_i^{(n)} \leftarrow \tau \mathbf{z}_i^{(n)} + (1 - \tau)\mathbf{P}_i^{(n)}(0)\boldsymbol{\mu}_i^{(n-1)}$, set $\boldsymbol{\mu}_i^{(n)} = [\mathbf{P}_i^{(n)}(0)]^{-1}\mathbf{z}_i^{(n)}$, $\boldsymbol{e}_i^{(n)} = \sum_j \mathbf{S}_{ij}\boldsymbol{\mu}_j^{(n)} - \boldsymbol{b}_i$ and Err $= \max_i\{||\boldsymbol{e}_i^{(n)}||_\infty\}$.
   c) If Err $> \epsilon$, do for all $i \in \{1, 2, \ldots, p\}$ and all $j \in \mathcal{N}_i$: $\mathbf{Q}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)}(0) - \mathbf{Q}_{ji}^{(n)}]^{-1}\mathbf{S}_{ij}$ and $\mathbf{v}_{ij}^{(n+1)} = -\mathbf{S}_{ji}[\mathbf{P}_i^{(n)}(0) - \mathbf{Q}_{ji}^{(n)}]^{-1}[\mathbf{z}_i^{(n)} - \mathbf{v}_{ji}^{(n)}]$.
   d) Increment n.
   e) If $n = m$, break.
5. End.

respect to $\lambda^{(n)}$ by applying Equation (4.52) (where $n \leftarrow n + 1$) for all nodes i. Set $j = \text{argmax}_i\{|\mathbf{s}_i'\boldsymbol{\mu}^{(n)}(\lambda^{(n)}) - \boldsymbol{b}|\}$, where $\mathbf{s}_i'$ is the ith row of $\mathbf{S}$, and let $\text{div}(\lambda^{(n)}) = \mathbf{s}_j'\nabla\boldsymbol{\mu}^{(n)}(\lambda^{(n)})$, where $\nabla\boldsymbol{\mu}^{(n)}(\lambda^{(n)})$ is the gradient of $\boldsymbol{\mu}^{(n)}(\lambda^{(n)})$ with respect to $\lambda^{(n)}$. Consider λ_0 as a candidate for $\lambda^{(n)}$. To evaluate $\text{div}(\lambda_0)$, we need to perform the message updates using λ_0 as the value for the tuning parameter. After the message updates we see that using $\lambda_0 - \alpha\text{sign}(\text{div}(\lambda_0))$ instead of λ_0 would have been better (for sufficiently small α) in the sense that it would have given posterior means that are closer to solving the linear system $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$. If we assume that $\lambda^{(n)}$ was decided upon at iteration $n-1$, we can make the retrospective adjustment $\lambda^{(n+1)} = \lambda^{(n)} - \alpha\text{sign}(\text{div}(\lambda^{(n)}))$. We test this heuristic measure in the empirical section by varying α over different values.

4.5 Empirical Results

We conducted two empirical studies. In the first we compare sGaBP to the multivariate extensions of RGaBP and CFGaBP by considering both convergence speed and inference quality. We describe the RGaBP and CFGaBP for

Algorithm 8 Synchronous CFGaBP

1. Specify a tolerance ϵ, a maximum number of iterations m and a diagonal loading λ.
2. Initialise $\boldsymbol{\mu}_{\text{work}} = \mathbf{0}$.
3. Set Err = Inf.
4. While Err $> \epsilon$
   a) Compute $\mathbf{h} = \boldsymbol{b} - \mathbf{S}\boldsymbol{\mu}_{\text{work}}$.
   b) Apply ordinary GaBP using the precision matrix $\mathbf{S} + \lambda\mathbf{I}_k$ and the potential vector $\mathbf{h}$. This can be done by setting $\lambda = 0$ or $\tau = 1$ in Algorithm 3 or Algorithm 7 respectively.
   c) Let $\boldsymbol{\xi}$ be the posterior means supplied in Step (4b). Set $\boldsymbol{\mu}_{\text{work}} \leftarrow \boldsymbol{\mu}_{\text{work}} + \boldsymbol{\xi}$ and let Err $= ||\mathbf{S}\boldsymbol{\mu}_{\text{work}} - \boldsymbol{b}||_\infty$.
   d) Increment n by the number of iterations performed by GaBP in Step (4b).
   e) If $n \geq m$, break.
5. End.

nodes of any size in Algorithms 7 and 8 respectively. In the literature, these algorithms are formulated for univariate nodes, but can easily be extended to the multivariate case. The second study is dedicated to testing the heuristic described in the previous section.

4.5.1 Comparison of sGaBP with Other Methods

We simulated data using the following procedure:

1. Select a $\tilde{\rho}$ uniformly from the interval $[1; 1.3]$.
2. Using the method from Kamper *et al.* (2018*a*), we generate a 100×100 precision matrix $\mathbf{S}$ with zero-diagonal spectral radius equal to $\tilde{\rho}$, along with a 100×1 potential vector $\boldsymbol{b}$. Recall that the zero-diagonal spectral radius of $\mathbf{S}$ is defined as the spectral radius of $\mathbf{I}_k - \mathbf{S}$ after $\mathbf{S}$ has been scaled to have only ones along its diagonal.
3. The 100 variables are assigned randomly to 10 clusters each of size 10.
4. For each of sGaBP, RGaBP and CFGaBP, we determine the prior parameters yielding convergence in the minimum number of iterations using a line search with increments of 0.01. These parameters are then used to initialise the methods.

This process was repeated 1 000 times. For each simulation, we record the number of iterations required for convergence and the posterior precisions for each cluster supplied by each method. For sGaBP and RGaBP, the precision estimates are computed as

$$\hat{\mathbf{P}}_i = \mathbf{S}_{ii} + \sum_{t \in \mathcal{N}_i} \mathbf{Q}_{ti}. \tag{4.53}$$

Because we are supplying $\mathbf{S} + \lambda \mathbf{I}_k$ to the inner loop of CFGaBP, we propose computing the precision estimate of CFGaBP for cluster i as

$$\hat{\mathbf{P}}_i = \mathbf{S}_{ii} - \lambda \mathbf{I}_{d_i} + \sum_{t \in \mathcal{N}_i} \mathbf{Q}_{ti}. \tag{4.54}$$

Note that, if we use the same λ for sGaBP and CFGaBP, then the precision estimates will be the same. They are likely to differ in the simulations, since the λ yielding the convergence in the smallest number of iterations will differ between the methods. We note the following practical considerations for the CFGaBP algorithm:

1. The converged posterior precisions are the posterior precisions obtained from the first inner-loop application of GaBP in Algorithm 8.

2. This is because, in the later stages of the outer-loop of Algorithm 8, $\mathbf{h} = \boldsymbol{b} - \mathbf{S}\boldsymbol{\mu}_{\text{work}} \approx \mathbf{0}$ and this could cause the inner-loop application of GaBP to terminate before the convergence of the precision components.

To compare inference quality, we consider the KL divergence of the exact marginal of a cluster to its corresponding posterior distribution (recall Equation (2.13)). Because all the methods considered yield the exact marginal means at convergence, the KL divergence of the exact marginal of a cluster to its corresponding posterior distribution will only be influenced by the precisions of the respective distributions. For a specific simulation, each method is represented by the mean of all the KL divergences of the exact marginals to their corresponding posterior distributions.

The results for the convergence speed (as measured by the number of iterations required for convergence) and inference quality (as measured by the mean KL divergence) are summarised in Figures 4.1 and 4.2 respectively. The convergence speed of CFGaBP is slow compared to that of the other methods. This is caused by the double-loop implementation in Algorithm 8, Step (b). sGaBP tends to converge faster than RGaBP. In terms of inference quality, the performance of RGaBP is poor compared to that of the other methods. The best inference quality is provided by sGaBP. Clearly, sGaBP outperformed the competitors in our simulations.

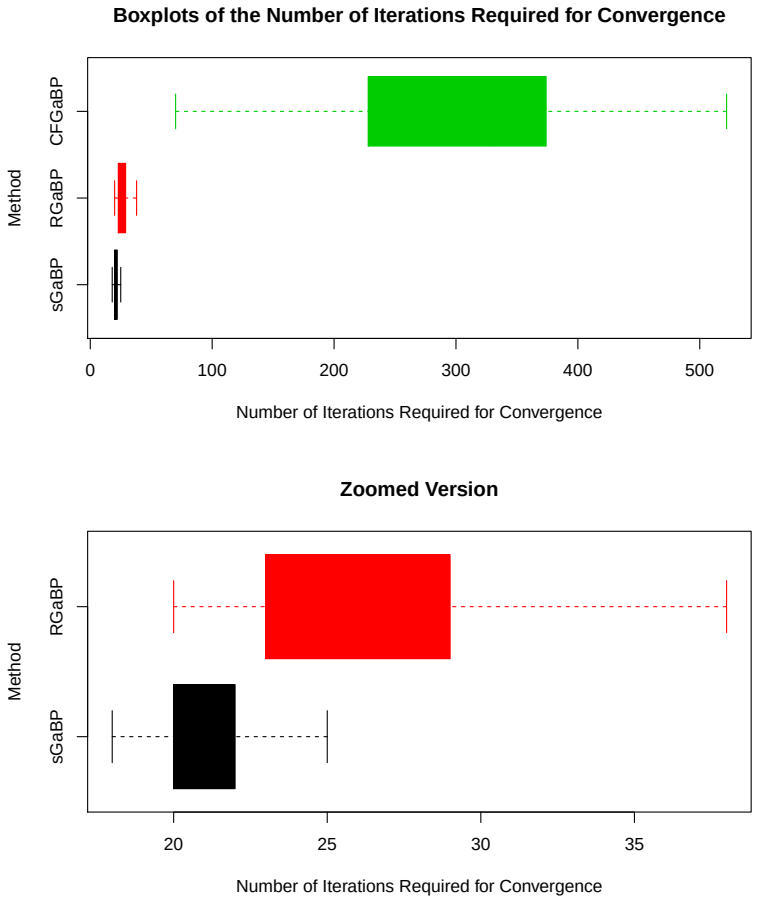


Figure 4.1: Visualisation of the results of our simulations comparing the number of iterations required for convergence by sGaBP, RGaBP and CFGaBP. The bottom panel zooms in on the boxplots corresponding to sGaBP and RGaBP. CFGaBP required the most number of iterations to converge. The number of iterations required for convergence by sGaBP and RGaBP are more comparable, with sGaBP tending to require a smaller number of iterations to converge.

It is possible to measure inference quality by computing the KL divergence of the posterior distribution to its corresponding exact marginal (this is the opposite direction used in the simulations). We note that changing the order of the KL divergence does not effect the conclusions drawn from Figure 4.2.

4.5.2 Performance of Heuristic Regularisation

In this section we investigate how well the heuristic regularisation tuning approaches optimal regularisation. For this purpose, we use the data from the previous section and compare optimal sGaBP with different initialisations of the heuristic. For each application of the heuristic, we start with $\lambda = 0$ and

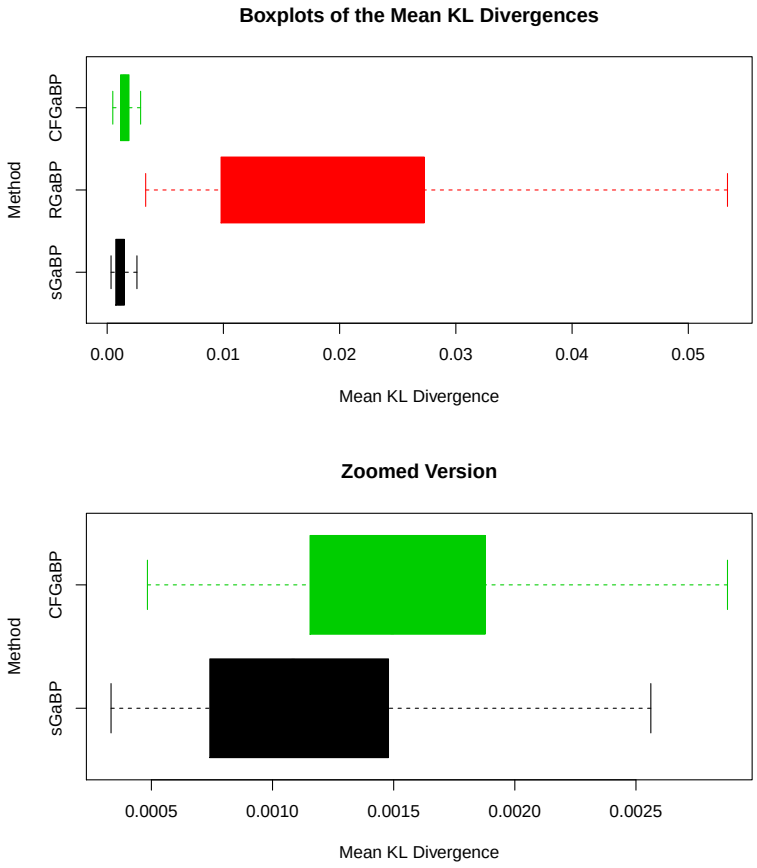


Figure 4.2: Visualisation of the results of our simulations comparing the inference quality of sGaBP, RGaBP and CFGaBP. The bottom panel zooms in on the boxplots corresponding to sGaBP and CFGaBP. The inference quality of RGaBP is poor compared to that of the other methods. This is because RGaBP computes precision estimates in the same manner as ordinary GaBP. Clearly, sGaBP performed the best of the methods in terms of inference quality.

consider using $\alpha = 0.01, 0.05$ and 0.1. The different methods are compared in terms of the number of iterations required for convergence. The results are given in Figure 4.3. We see that the heuristics with $\alpha = 0.05$ and $\alpha = 0.1$ compares well with the optimal method (sGaBP initialised to have fastest convergence), but they tend to converge at a slower speed. The heuristic with $\alpha = 0.01$ does not compare well with the other methods. We see that the heuristic makes some progress in shifting the regularisation towards the optimal level, but it is sensitive to the selection of α. This simulation study shows that our heuristic can play a role in the selection of the regularisation parameter, given appropriate initialisation.

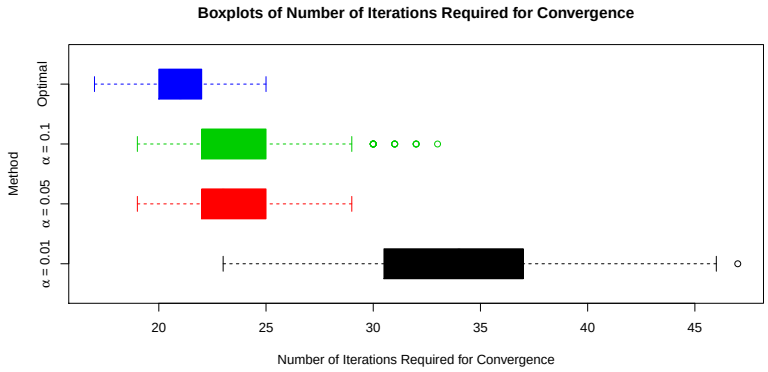


Figure 4.3: Visualisation of the results of our simulations of the convergence speed of the heuristic method, with different initialisations, compared to optimal regularisation. We see that the heuristics with $\alpha = 0.05$ or $\alpha = 0.1$ compares well with the optimal method, although they tend to provide slower convergence. The $\alpha = 0.01$ heuristic does not compare well with the other methods, indicating that the adjustments are done too slowly.

4.5.3 Further Empirical Considerations

In this section, we investigate the effect of selecting different error thresholds to define convergence on the performance of sGaBP and RGaBP (similar to Section 2.5.4). We generate four 100×100 precision matrices (and potential vectors), each associated with a different spectral radius in $\{1, 1.1, 1.2, 1.3\}$. For each of these matrices, the sGaBP and RGaBP methods were initialised to yield convergence to a 10^{-10} error threshold in as few iterations as possible (this was determined by a line search in increments of 0.01).

Figure 4.4 displays the log convergence error as a function of the iteration number over the four applications of sGaBP and RGaBP. In all the graphs, we see that the convergence error associated with both methods are reasonably comparable for smaller iteration numbers, but starts to deviate significantly for higher iteration numbers, where sGaBP performs the better of the two methods. Moreover, as the zero-diagonal spectral radius increases, we observe that the performance of sGaBP relative to RGaBP improves.

In short, Figure 4.4 shows that sGaBP outperforms RGaBP in terms of convergence speed, with the relative performance of sGaBP improving for smaller error thresholds.

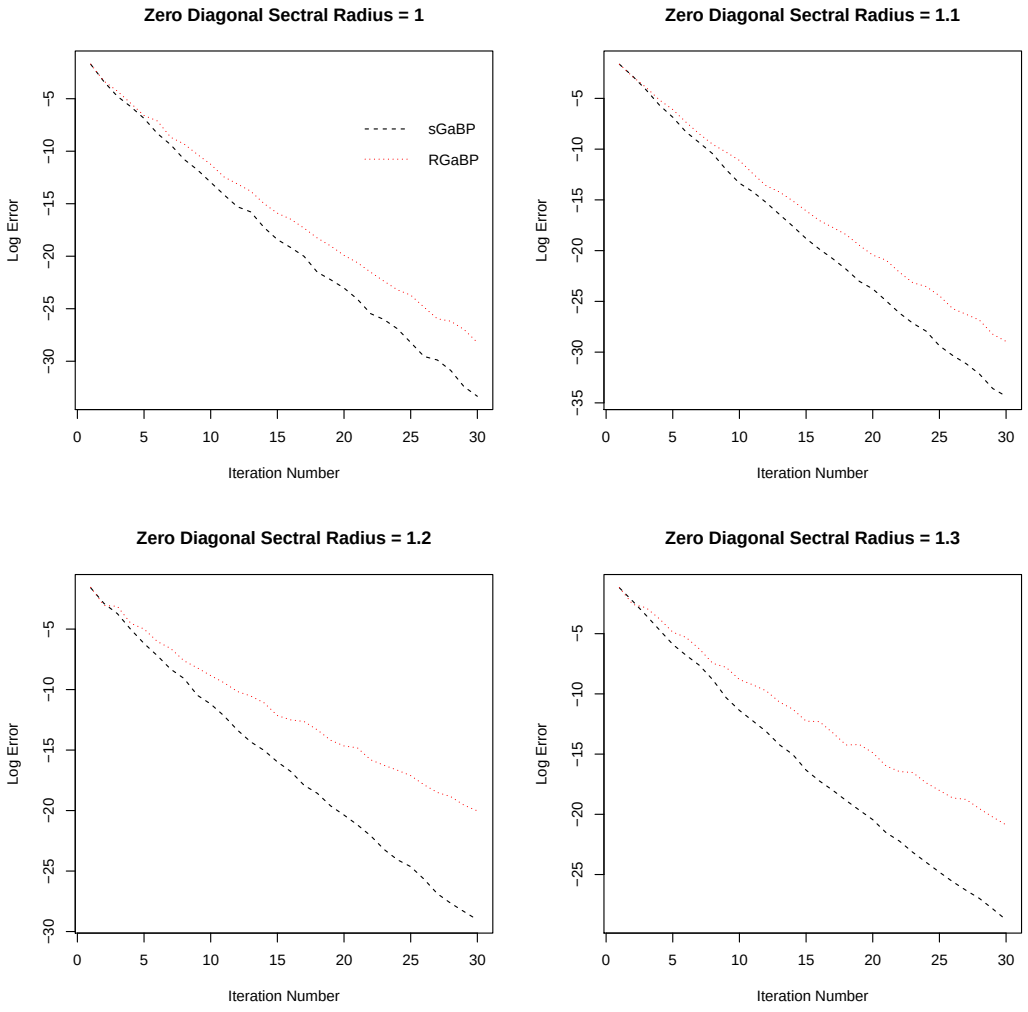


Figure 4.4: Comparison of the convergence error as a function of the iteration number for RGaBP (red dotted line) and sGaBP (black striped line) over four different data structures with different spectral radii.

4.6 Conclusion

In this chapter we have proven convergence of the sGaBP algorithm where nodes are allowed to be of any size, given sufficient regularisation. In the empirical section, we showed that sGaBP tends to converge faster compared to certain competing algorithms based on optimal convergence speed initialisations. When the algorithms involved in the empirical section are initialised to yield optimal convergence speed, the empirical results show that sGaBP provides more accurate approximations for the posterior precisions compared to its competitors. In particular, we see that the observations made for the univariate case (Chapter 2) also extend to multivariate nodes.

Chapter 5

Conclusion

In this dissertation, we have considered the problem of the convergence of BP on MGs where the graph is allowed to be of any topology. In general, BP on loopy graphs is not guaranteed to converge and does not necessarily provide exact inference if convergence occurs. We proposed applying the principle of node regularisation to MGs as a tool to address the problems associated with BP on loopy graphs. This principle aims to regularise the potentials used in the forming of messages, thereby affecting the way nodes communicate. To test this principle, we considered BP on a MG induced by a multivariate Gaussian distribution in canonical parameterisation (GaBP). The main aim of this dissertation was to show that the principle of node regularisation has the potential to improve BP on pairwise MGs in the sense that it guarantees convergence and tends to provide more accurate inference in cases where basic BP converges. This aim was achieved by proving that one can always apply regularisation, such that sGaBP will converge regardless of the way in which variables are assigned to nodes. We also discussed the implementation of node regularisation in such a way that the converged posterior means supplied by sGaBP are the exact marginal means. Hence sGaBP can be considered as an iterative solver of linear systems involving symmetric and positive definite matrices. An empirical approach was taken to investigate the accuracy of the posterior precisions supplied by sGaBP as approximations for the true marginal precisions, where it was found that sGaBP compared favourably to certain competing algorithms. When selecting the degree of regularisation, such that sGaBP converges at the fastest speed, we showed that sGaBP can improve inference quality when compared to some of the other variants of GaBP available in the literature.

Although progress was made in this dissertation in addressing the convergence problems of BP on loopy graphs, there are several unanswered questions that need to be explored. In the final part of this dissertation we list some of these unanswered questions and discuss how they could be answered in further research.

5.1 Selection of the Degree of Regularisation

To apply sGaBP we need to specify a regularisation parameter a priori. The approach taken in this dissertation regarding the selection of the regularisation parameter has largely been through the use of heuristics, and no theoretical guarantees were offered. Within the context of basic GaBP, convergence is guaranteed when the precision matrix is walk-summable (this is implied by diagonal dominance). In Chapter 3, we extended walk-summability to preconditioned walk-summability for nodes with more than one variable. A logical suggestion for the selection of the degree of regularisation would be to select λ such that $\mathbf{S} + \lambda\mathbf{I}$ is walk-summable or diagonally-dominant. Consider, for example, the following precision matrix and potential vector:

$$\mathbf{S} = \begin{bmatrix} 1.0000000 & 0.2435805 & 0.9642933 \\ 0.2435805 & 1.0000000 & 0.3026115 \\ 0.9642933 & 0.3026115 & 1.0000000 \end{bmatrix}$$

$$\boldsymbol{b} = \begin{bmatrix} -0.4490656 \\ -0.9778466 \\ -0.5050349 \end{bmatrix}.$$

It can be validated that $\mathbf{S}$ is not walk-summable. If we run sGaBP (univariate nodes) with $\lambda = 0.11$, we observe divergent behaviour. Since $\mathbf{S} + \lambda\mathbf{I}$ is walk-summable, we therefore have a counter-example for selecting λ large enough for walk-summability as sufficient for convergence. sGaBP starts to converge at values higher than $\lambda = 0.14$, and also for λ such that $\mathbf{S} + \lambda\mathbf{I}$ is diagonally dominant. We have not been able to generate an example where selecting λ large enough for $\mathbf{S} + \lambda\mathbf{I}$ to be diagonally dominant causes sGaBP to diverge.

There are also examples where sGaBP converges for values of λ where $\mathbf{S} + \lambda\mathbf{I}$ is not walk-summable (see, for instance, the diabetes data). In these cases, the posterior precisions supplied by sGaBP may be negative definite and it could be beneficial to sacrifice some convergence speed for better approximations of the marginal precisions.

Clearly, more detail regarding the convergence behaviour of sGaBP as a function of λ is necessary, along with practical suggestions on how to select λ for both fast convergence and good inference quality.

5.2 Multiple Regularisation Parameters

A natural way to improve on the convergence behaviour of sGaBP is to use more regularisation parameters. The extreme case would be to assign a unique parameter to every node. Consider applying sGaBP (with univariate nodes) to the diabetes data, as in chapter 2. There we saw that $\lambda = 1.29$ yielded

the fastest convergence and, for this selection of λ, convergence occurred after 574 iterations. Using 2.4771650, 1.3293856, 2.0373875, 1.8105460, 0.8841369, 1.0011064, 2.9310134, 0.8063211, 2.4593143 and 3.1425276 (these values were obtained by minimising the spectral radius of the linear update matrix using general-purpose optimisation) as regularisation parameters for the respective nodes, we see that the number of iterations required for convergence can be decreased to 141 (speed-up factor of more than 4). Using multiple regularisation parameters can also improve the inference quality of the posterior distributions of sGaBP.

In this dissertation, we did not cover the use of multiple regularisation parameters. It is not difficult to adjust our heuristic method, presented in Section 4.4.4, to allow for multiple parameters. In our experience, the heuristic struggles to find good regularisation parameters and quite often does worse than the single-parameter heuristic. A compromise would be to constrain certain (but not all) clusters to use the same regularisation parameter. Dividing clusters into groups, where clusters in a group use the same regularisation parameter, would need to be considered.

Another consideration relates to the automatic preconditioning done by basic GaBP. Using a single regularisation parameter tends to void this property. Finding a regularisation scheme that preserves this type of preconditioning is an interesting topic for further research.

5.3 Linear Systems and Distributive Application

GaBP can be used as an iterative solver of linear systems. Within this context, our contribution is the development of a BP-based variant that can solve any symmetric positive definite linear system of equations. Within the context of solving linear systems, this dissertation does not establish sGaBP as a competitor for methods such as the CG and PCG solvers (although some tentative comparisons were made). The literature does contain some examples where GaBP compares favourably with the PCG method within the context of large, sparse, diagonally dominant linear systems (El-Kurdi *et al.*, 2012*b*).

The favourable convergence speed comparisons of sGaBP with other GaBP variants suggest that this algorithm should be researched for the purpose of solving linear systems. This research is related to the discussion surrounding the selection of regularisation parameters, since these play a crucial role in the convergence speed of the algorithm.

Also related to solving linear systems is the ease of distributed implementation of BP algorithms. Like ordinary GaBP, sGaBP is convenient to implement in distributed settings and can also benefit from sparsity in the precision matrix. The performance of sGaBP in solving large and sparse linear systems (not necessarily diagonally dominant or walk-summable) should be researched within a distributed context and compared to that of the CG and PCG methods.

5.4 Extensions to Other Graph Types

We have discussed the application of our high-level approach specifically for a MG constructed from a multivariate Gaussian distribution in canonical parameterisation. A next step would be to generalise our high-level approach for other (non-Gaussian) MGs. One could attempt to apply the principle of node regularisation on the exact messages. Another approach would be to construct an approximate message-passing scheme using Gaussian approximations. Essentially, we propose applying node regularisation within the context of expectation propagation (Minka, 2001). Consider Equation (1.11) from Chapter 1:

$$\begin{aligned} m_{ij}^{(n+1)}(\mathbf{x}_j) &= \max_{\mathbf{x}_i} \left\{ \log(\psi_{ii}(\mathbf{x}_i)) + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) + \sum_{t \in \mathcal{N}_i \backslash j} m_{ti}^{(n)}(\mathbf{x}_i) \right. \\ &\quad \left. - \frac{\lambda}{q} ||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_q^q \right\} \\ &= \max_{\mathbf{x}_i} \left\{ \log(\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)) - \frac{\lambda}{q} ||\mathbf{x}_i - \boldsymbol{\theta}_i^{(n-1)}||_q^q + \log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) \right\}. \end{aligned}$$

Let us assume that $\log(\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j)) = -\mathbf{x}_i' \mathbf{S}_{ij} \mathbf{x}_j$ as in the Gaussian case, but we leave $\psi_{ii}(\mathbf{x}_i)$ unspecified. Furthermore, suppose that $m_{ij}^{(n)}(\mathbf{x}_j) \propto -\frac{1}{2}\mathbf{x}_j' \mathbf{Q}_{ij}^{(n)} \mathbf{x}_j + \mathbf{x}_j' \mathbf{v}_{ij}^{(n)}$ for certain matrices $\mathbf{Q}_{ij}^{(n)}$ and vectors $\mathbf{v}_{ij}^{(n)}$. Consider

$$\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i) = \psi_{ii}(\mathbf{x}_i) \times \prod_{t \in \mathcal{N}_i \backslash j} \exp\left[m_{ti}^{(n)}(\mathbf{x}_i) \right], \tag{5.1}$$

where $\psi_{ii}(\mathbf{x}_i)$ may cause $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ to be non-Gaussian. In all likelihood, solving Equation (5.1) will cause the messages to lose their conjugacy. However, if we approximate $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ with some Gaussian distribution, and use this approximation for propagation purposes, we can preserve the conjugacy of the messages. Let $\widehat{\text{post}}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ be the Gaussian approximation for $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$. A natural way to select $\widehat{\text{post}}_{i\backslash j}^{(n)}(\mathbf{x}_i)$, is such that the KL divergence of $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ to $\widehat{\text{post}}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ is minimum (note that both these posterior distributions must be

normalised in the computation of the KL divergence). This is obtained by setting $\widehat{\text{post}}_{i\backslash j}^{(n)}(\mathbf{x}_i)$ as a Gaussian with the same mean and variance as $\text{post}_{i\backslash j}^{(n)}(\mathbf{x}_i)$. Unfortunately, analytical expressions for the expected value and variance will not always be available, and we could be forced to use methods such as numerical integration to obtain these quantities. This would increase the computational burden of the algorithm. Another issue is how much the Gaussian approximations will influence inference quality.

Another important type of graphical model is obtained through the following representation of a density function:

$$f(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{P}\psi_i(\mathbf{x}_i), \tag{5.2}$$

where Z is a normalisation constant and $\mathbf{x}_i$ are subvectors of $\mathbf{x}$ that are allowed to overlap. The problem posed in this context is to estimate the marginal of each $\mathbf{x}_i$. The density function given in Equation (5.2) can be represented as a cluster graph or a factor graph, and BP on these graphs can be used to estimate the marginals. Again, we are confronted with the loopy-graph problem, i.e. convergence is not guaranteed and, if convergence occurs, the inference provided is not exact. To achieve the convergence of BP, these graphs are usually constructed to have the running intersection property (RIP). BP on graphs with the RIP tends to converge, although this is not guaranteed. As an alternative to constructing graphs to have the RIP, one can attempt to apply node regularisation to achieve convergence. A natural extension of our work would be to assume that the $\psi_i(\mathbf{x}_i)$ factors in Equation (5.2) are proportional to Gaussian densities. If node regularisation proves effective in this setting, one can attempt the marginalisation of Equation (5.2) when the factors involved are discrete tables.

5.5 Novel Applications of GaBP

There are novel ways of utilising the GaBP algorithm. Consider, for instance, the problem of linear regression on the data points $(\mathbf{x}_i, y_i)$ for $i = 1, 2, \ldots, n$. The least-squares estimates of the linear regression coefficients are $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, where $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \ldots & \mathbf{x}_n\end{bmatrix}'$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)'$. One can pose the problem of least-squares estimation as a marginalisation problem by considering a Gaussian distribution with precision $\mathbf{X}'\mathbf{X}$ and potential $\mathbf{X}'\mathbf{y}$. The marginal means in this context equal the least-squares coefficients. If we apply sGaBP (with sufficient regularisation), our message-passing algorithm will converge to $\hat{\boldsymbol{\beta}}$. Since each round of message passing yields a posterior mean, we can interpret sGaBP as a path algorithm for least-squares estimation. In particular, setting λ to be large will relate sGaBP to

slow-learning algorithms (Efron *et al.*, 2004) and sGaBP should be researched within this context.

Another possibility is to use sGaBP to perform clustering of a set of data points $\mathbf{x}_i : i = 1, 2, \ldots, n$. Consider an association matrix $\mathbf{S}$, where S_{ij} denotes the association between $\mathbf{x}_i$ and $\mathbf{x}_j$. We assume that $\mathbf{S}$ is constructed to be symmetric and positive definite. The basic idea is to perform sGaBP on a Gaussian distribution with precision matrix $\mathbf{S}$, and to utilise the strength of the messages to cluster the points. An example of this type of clustering can be found in the detection of F-formations in free-standing conversational groups (Kamper, 2017). The problem addressed in this work was detecting F-formations in the Salsa data, among 18 participants who spent over 60 minutes in a poster presentation and cocktail party, using certain measures of social interactions (Alameda-Pineda *et al.*, 2015). An F-formation arises whenever two or more people sustain a spatial and orientational relationship in which the space between them is one to which they have equal, direct and exclusive access (Kendon, 1990). The approach was to use the measures of social interactions to create an association matrix between individuals. BP was performed on the association matrix, and graph cuts were performed on the messages for the purpose of forming F-formations. Based on optimal comparisons, the BP approach outperformed other state-of-the-art methods (Kamper, 2017). One possible reason for this is that message-passing emulates the way in which individuals communicate in a social environment. It is therefore natural to suggest that GaBP should be considered for the purpose of performing clustering on social data.

Appendices

Appendix A

Proofs for Chapter 2

A.1 Proof of Theorem 1

The proof is contained in the following list.

1. From 1, all the precision components are negative at stage n, hence $Q_{ij}^{(n+1)} = \frac{-S_{ij}^2}{1+\lambda+\sum_{t\in\mathcal{N}_i/j} Q_{ti}^{(n)}} = \frac{-S_{ij}^2}{1+\lambda-\sum_{t\in\mathcal{N}_i/j}|Q_{ti}^{(n)}|}$. From 3 we have that $\sum_{t\in\mathcal{N}_i/j}|Q_{ti}^{(n)}| \leq \sum_{t\in\mathcal{N}_i}|Q_{ti}^{(n)}| < \delta_i^{(n)} < 1+\lambda$ and $1+\lambda-\sum_{t\in\mathcal{N}_i/j}|Q_{ti}^{(n)}| > 0$, from which 1 follows for iteration $n+1$.

2. $|Q_{ij}^{(n+1)}| = \frac{S_{ij}^2}{1+\lambda-\sum_{t\in\mathcal{N}_i/j}|Q_{ti}^{(n)}|} > \frac{S_{ij}^2}{1+\lambda-\sum_{t\in\mathcal{N}_i/j}|Q_{ti}^{(n-1)}|} = |Q_{ij}^{(n)}|$ since $|Q_{ti}^{(n)}| > |Q_{ti}^{(n-1)}|, t \in \mathcal{N}_i$ from 2 for iteration n, and hence 2 is also true for $n+1$.

3. $\delta_i^{(n+1)} = \sum_{t\in\mathcal{N}_i}|Q_{ti}^{(n+1)}| = \sum_{t\in\mathcal{N}_i}\frac{S_{ti}^2}{1+\lambda-\delta_t^{(n)}+|Q_{it}^{(n)}|} \leq \sum_{t\in\mathcal{N}_i}\frac{S_{ti}^2}{1+\lambda-\delta_t+|Q_{it}^{(n)}|} \leq \delta_i < 1+\lambda$ by 4, and therefore 3 is true for $n+1$.

4. From the above, we have $\sum_{t\in\mathcal{N}_i}\frac{S_{ti}^2}{1+\lambda-\delta_t+|Q_{it}^{(n+1)}|}$ $\leq \sum_{t\in\mathcal{N}_i}\frac{S_{ti}^2}{1+\lambda-\delta_t+|Q_{it}^{(n)}|} \leq \delta_i$, hence 4 holds for $n+1$.

A.2 Proof of Theorem 2

Let $\mathbf{S}$ be a symmetric, positive definite matrix with diagonal entries equal to 1, and let its entries be denoted by S_{ij}. Values $Q_{ij}(\lambda)$ are characterised by the system

$$Q_{ij} = Q_{ij}(\lambda) = -\frac{S_{ij}^2}{1+\lambda+\sum_{t\in\mathcal{N}_i/j} Q_{ti}(\lambda)}, \qquad 1 \leq i,j \in \mathcal{N}_i.$$

We are particularly interested in the behaviour of Q_{ij} as $\lambda \to \infty$. A consequence of Theorem 1 is that $\lim_{\lambda\to\infty} Q_{ij} = 0$. For convenience, set $\delta = \lambda^{-1}$, so

that $\delta \to 0$. The system can be rewritten as

$$Q_{ij}\Big(-\delta \sum_{t\in\mathcal{N}_i/j} Q_{ti} - 1 - \delta\Big) - \delta S_{ij}^2 = 0, \qquad 1 \le i,j \in \mathcal{N}_i.$$

Note that

$$\frac{\partial}{\partial Q_{kl}}\left(Q_{ij}\Big(-\delta \sum_{t\in\mathcal{N}_i/j} Q_{ti} - 1 - \delta\Big) - \delta S_{ij}^2\right)$$
$$= \begin{cases} -\delta \sum_{t\in\mathcal{N}_i/j} Q_{ti} - 1 - \delta & (k,l) = (i,j), \\ -\delta Q_{ij} & l = i, k \in \mathcal{N}_i/j, \\ 0 & \text{otherwise.} \end{cases}$$

As $\delta \to 0$, we see that the Jacobian of the system tends to a negative identity matrix, so in particular it is invertible. This means that the Q_{ij} are analytic functions of δ if δ is in a suitable neighbourhood of 0. Consequently, the Q_{ij} have power series expansions in δ:

$$Q_{ij} = a_{ij}\delta + b_{ij}\delta^2 + \cdots$$

Plugging this back into the system, we see that $a_{ij} = -S_{ij}^2$. Defining $C_{ij} = \frac{Q_{ij}}{S_{ij}}$ we see that

$$C_{ij} = -S_{ij}\delta + \mathcal{O}(\delta^2).$$

Consider again the matrix $\mathbf{L}$ given in (2.10). Let $l = p^2 - p$, we now define a $l \times p$ matrix $\mathbf{G} = \begin{bmatrix}\mathbf{g}_1 & \mathbf{g}_2 & \dots & \mathbf{g}_p\end{bmatrix}$. The vector $\mathbf{g}_i$ has entries 1 in positions $(p-1)(i-1)+1, \dots, (p-1)i$. It can be shown that

$$\mathbf{L} : p^2 \times p^2 = \begin{bmatrix} \mathbf{L}_{11} : l \times l & \frac{\lambda}{p-2}\mathbf{L}_{11}\mathbf{G} \\ \frac{1}{\lambda}\mathbf{G}'\mathbf{L}_{22} & \mathbf{L}_{22} : p \times p \end{bmatrix}, \tag{A.1}$$

with the understanding that $\lambda > 0$. Let

$$\mathbf{D} = \begin{bmatrix} \mathbf{I} : l \times l & \mathbf{0} : l \times p \\ \mathbf{0} : p \times l & \lambda[\mathbf{I} : p \times p] \end{bmatrix} \tag{A.2}$$

and set $\tilde{\mathbf{L}} = \mathbf{D}\mathbf{L}\mathbf{D}^{-1}$. It is easy to see that

$$\tilde{\mathbf{L}} : p^2 \times p^2 = \begin{bmatrix} \mathbf{L}_{11} : l \times l & \frac{1}{p-2}\mathbf{L}_{11}\mathbf{G} \\ \mathbf{L}_{22}\mathbf{G}' & \mathbf{L}_{22} : p \times p \end{bmatrix}, \tag{A.3}$$

and that $\mathbf{L}$ and $\tilde{\mathbf{L}}$ will have the same eigenvalues. As a first step, we show that the eigenvalues of $\tilde{\mathbf{L}}$ are all clustered around 0 and 1 as $\delta \to 0$. We have already discussed the construction of $\mathbf{L}_{11}$ from the elements C_{ij}. Using the fact that $C_{ij} = -\delta S_{ij} + \mathcal{O}(\delta^2)$, we see that $\mathbf{L}_{11} = \delta\mathbf{A} + \mathcal{O}(\delta^2)$ where $\mathbf{A}$ does not depend on λ, and $\mathcal{O}(\delta^2)$ is of a suitable dimension, with each entry

being $\mathcal{O}(\delta^2)$. The matrix $\mathbf{A}$ is constructed exactly as $\mathbf{L}_{11}$, however $-S_{ij}$'s are used instead of $-C_{ij}$'s. As discussed, the matrix $\mathbf{L}_{22}$ is diagonal, with entries $\frac{\lambda}{1+\lambda+\sum_{t\neq i} Q_{ti}} = \frac{\lambda}{1+\lambda+\mathcal{O}(\delta)} = 1-\delta+\mathcal{O}(\delta^2)$, and therefore $\mathbf{L}_{22} = \mathbf{I} - \delta\mathbf{I} + \mathcal{O}(\delta^2)$. We now consider the matrix

$$\tilde{\mathbf{L}} = \begin{bmatrix} \delta\mathbf{A} & \frac{\delta}{p-2}\mathbf{A}\mathbf{G} \\ (1-\delta)\mathbf{G}' & (1-\delta)\mathbf{I} \end{bmatrix} + \mathcal{O}(\delta^2) \tag{A.4}$$

and the following lemma.

Lemma 1 *Let* $\mathbf{M}$ *be a square matrix, and let* c *be a positive constant that satisfies* $c > \|\mathbf{M}\|_\infty$ *(*$\|\mathbf{M}\|_\infty$ *is the* ∞*-norm of* $\mathbf{M}$*, which can be obtained by calculating the row sums of the absolute values of entries in* $\mathbf{M}$ *and taking the maximum of these sums). For every* x *with* $|x| \geq c$*, the matrices* $x\mathbf{I} - \mathbf{M}$ *and* $\mathbf{I} - \frac{1}{x}\mathbf{M}$ *are invertible, and the entries of* $(\mathbf{I} - \frac{1}{x}\mathbf{M})^{-1}$ *are bounded by constants that only depend on* c *and* $\mathbf{M}$*.*

The invertibility follows directly from the fact that the matrix $x\mathbf{I}-\mathbf{M}$ is strictly diagonally dominant by our assumptions. For the second statement, let $|\mathbf{M}|$ be obtained from $\mathbf{M}$ by replacing all entries with their absolute values. Note that $|\mathbf{M}|$ has the same ∞-norm as $\mathbf{M}$. Clearly, the entries of

$$\left(\mathbf{I} - \frac{1}{x}\mathbf{M}\right)^{-1} = \sum_{j=0}^{\infty} x^{-j}\mathbf{M}^j$$

are bounded by the entries of

$$\left(\mathbf{I} - \frac{1}{c}|\mathbf{M}|\right)^{-1} = \sum_{j=0}^{\infty} c^{-j}|\mathbf{M}|^j,$$

which readily proves the desired statement.

Lemma 2 *There exists a constant* $K > 0$ *such that, for sufficiently small* δ*, each eigenvalue* x *of* $\tilde{\mathbf{L}}$ *either satisfies* $|x| \leq K\delta$ *or* $|x-1| \leq K\delta$*.*

We reason by contradiction and assume that there is an eigenvalue for which $|x| > K\delta$ and $|x-1| > K\delta$. Consider first $||\mathbf{L}_{11}||_\infty = ||\delta\mathbf{A} + \mathcal{O}(\delta^2)||_\infty \leq \delta||\mathbf{A}||_\infty + \mathcal{O}(\delta^2)$. If we choose K large enough (e.g., $K \geq \|\mathbf{A}\|_\infty + 1$), then the matrix $x\mathbf{I} - \mathbf{L}_{11} = x\mathbf{I} - \delta\mathbf{A} + \mathcal{O}(\delta^2)$ is invertible by the previous lemma for sufficiently small δ, and the entries of $(\mathbf{I} - \frac{1}{x}\mathbf{L}_{11})^{-1}$ are bounded by absolute constants. Now we use the Schur complement on A.4:

$$\begin{aligned} &\det(x\mathbf{I} - \tilde{\mathbf{L}}) = \det(x\mathbf{I} - \mathbf{L}_{11}) \\ &\times \det\left(x\mathbf{I} - \mathbf{L}_{22} - \frac{1}{p-2}\mathbf{L}_{22}\mathbf{G}'(x\mathbf{I} - \mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{G}\right). \end{aligned} \tag{A.5}$$

It remains to show that the second determinant is not equal to 0. We rewrite the matrix as follows:

$$\begin{aligned} &x\mathbf{I} - \mathbf{L}_{22} - \frac{1}{p-2}\mathbf{L}_{22}\mathbf{G}'(x\mathbf{I} - \mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{G} \\ &= (x-1)\mathbf{I} - (\mathbf{L}_{22} - \mathbf{I}) \\ &- \frac{1}{x(p-2)}\mathbf{L}_{22}\mathbf{G}'\Big(\mathbf{I} - \frac{1}{x}\mathbf{L}_{11}\Big)^{-1}\mathbf{L}_{11}\mathbf{G} \\ &= (x-1)\mathbf{I} + \mathbf{H}_1 + \frac{1}{x}\mathbf{H}_2. \end{aligned} \tag{A.6}$$

Consider $(\mathbf{L}_{22} - \mathbf{I}) = -\delta\mathbf{I} + \mathcal{O}(\delta^2)$ and $||\mathbf{L}_{22} - \mathbf{I}||_\infty = || - \delta\mathbf{I} + \mathcal{O}(\delta^2)||_\infty \leq \delta||\mathbf{I}||_\infty + ||\mathcal{O}(\delta^2)||_\infty = \delta + \mathcal{O}(\delta^2)$. Therefore $||\mathbf{H}_1||_\infty \leq \kappa_1\delta$ for a constant κ_1 and sufficiently small δ. The entries of $(\mathbf{I} - \frac{1}{x}\mathbf{L}_{11})^{-1}$ are bounded by $(\mathbf{I} - \frac{1}{K\delta}|\mathbf{L}_{11}|)^{-1} = \mathbf{I} + \frac{|\mathbf{L}_{11}|}{K\delta} + \sum_{j=2}^{\infty}\frac{|\mathbf{L}_{11}|^j}{K\delta}$ for sufficiently small δ by Lemma 1. Since $\mathbf{L}_{11} = \delta\mathbf{A} + \mathcal{O}(\delta^2)$, we have that $(\mathbf{I} - \frac{1}{x}\mathbf{L}_{11})^{-1} = \mathbf{I} + \frac{|\mathbf{A}|}{K} + \mathcal{O}(\delta) = \mathcal{O}(1)$. Furthermore, $\mathbf{L}_{22} = \mathcal{O}(1)$ and $\mathbf{L}_{11} = \mathcal{O}(\delta)$, from which we have that $\mathbf{H}_2 = \mathcal{O}(\delta) + \mathcal{O}(\delta^2)$ and $||\mathbf{H}_2||_\infty \leq \kappa_2\delta$ for a constant κ_2 and sufficiently small δ. If $|x| \geq \frac{1}{2}$, we find that

$$\Big\|\mathbf{H}_1 + \mathbf{H}_2\Big\|_\infty \leq \kappa_1\delta + \frac{\kappa_2\delta}{|x|} \leq (\kappa_1 + 2\kappa_2)\delta < K\delta \leq |x-1|$$

if K is chosen large enough (greater than $\kappa_1 + 2\kappa_2$). If $|x| \leq \frac{1}{2}$, we get

$$\Big\|\mathbf{H}_1 + \mathbf{H}_2\Big\|_\infty \leq \kappa_1\delta + \frac{\kappa_2\delta}{|x|} \leq \kappa_1\delta + \frac{\kappa_2}{K} < \frac{1}{2} \leq |x-1|$$

if K is chosen large enough and δ is sufficiently small. In either case, we can apply the previous lemma to see that the matrix in (A.6) is in fact invertible.

Now we focus on the eigenvalues that are close to 1, setting $x = 1 - \delta t$ for some t with $|t| \leq K$. Returning to (A.5), we observe that $x\mathbf{I} - \mathbf{L}_{11}$ is invertible for sufficiently small δ, again by Lemma 1. Hence we consider the second matrix:

$$\begin{aligned} &x\mathbf{I} - \mathbf{L}_{22} - \frac{1}{p-2}\mathbf{L}_{22}\mathbf{G}'(x\mathbf{I} - \mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{G} \\ &= (1-t)\delta\mathbf{I} - \frac{\delta}{p-2}\mathbf{G}'\mathbf{A}\mathbf{G} + \mathcal{O}(\delta^2). \end{aligned}$$

The entries of the matrix hidden by the $\mathcal{O}(\delta^2)$ term are in fact analytic in δ and t, since we proved earlier that the entries of $\tilde{\mathbf{L}}$ are analytic functions. We can take out a factor δ, to be left with the equation

$$\det\Big((1-t)\mathbf{I} - \frac{1}{p-2}\mathbf{G}'\mathbf{A}\mathbf{G} + \mathbf{M}\Big) = 0, \tag{A.7}$$

where the matrix $\mathbf{M}$ has entries that are analytic functions of δ and t (if δ is restricted to a sufficiently small neighbourhood of 0 and $|t| \leq K$). Moreover, $\mathbf{M} = \mathcal{O}(\delta)$. As $\delta \to 0$, we obtain (up to a change of variable $1 - t = u$) the characteristic equation of the matrix $\frac{1}{p-2}\mathbf{G}'\mathbf{A}\mathbf{G}$ (we show later that this matrix is in fact equal to $\mathbf{I} - \mathbf{S}$). Its p solutions (counted with multiplicity) give rise to p branches $t_1(\delta), t_2(\delta), \ldots, t_p(\delta)$ that solve the implicit equation (A.7). We can treat the "small" eigenvalues that are close to 0 in the same way. We set $x = \delta t$ for some t with $|t| \leq K$, and use the Schur complement with respect to the other diagonal block:

$$\begin{aligned}\det(x\mathbf{I} - \tilde{\mathbf{L}}) &= \det(x\mathbf{I} - \mathbf{L}_{22}) \\ &\quad \times \det\left(x\mathbf{I} - \mathbf{L}_{11} - \frac{1}{p-2}\mathbf{L}_{11}\mathbf{G}(x\mathbf{I} - \mathbf{L}_{22})^{-1}\mathbf{L}_{22}\mathbf{G}'\right). \qquad \text{(A.8)}\end{aligned}$$

Since

$$x\mathbf{I} - \mathbf{L}_{22} = (x - 1 + \delta)\mathbf{I} + \mathcal{O}(\delta^2) = -\mathbf{I} + \mathcal{O}(\delta),$$

this matrix is invertible for sufficiently small δ, again by Lemma 1. Moreover, we have

$$\begin{aligned}&x\mathbf{I} - \mathbf{L}_{11} - \mathbf{L}_{11}\mathbf{G}(x\mathbf{I} - \mathbf{L}_{22})^{-1}\mathbf{L}_{22}\mathbf{G}' \\ &= \delta(t\mathbf{I} - \mathbf{A} + \frac{1}{p-2}\mathbf{A}\mathbf{G}\mathbf{G}') + \mathcal{O}(\delta^2),\end{aligned}$$

so we can repeat the argument for the "large" eigenvalues. We obtain $p^2 - p$ branches $\bar{t}_1(\delta), \bar{t}_2(\delta), \ldots, \bar{t}_{p^2-p}(\delta)$ that correspond to the eigenvalues of $\mathbf{A} - \frac{1}{p-2}\mathbf{A}\mathbf{G}\mathbf{G}'$.

Returning to the large eigenvalues, we consider the product $\mathbf{G}'\mathbf{A}\mathbf{G}$. The matrix $\mathbf{A}$ is constructed by taking the first l rows and columns of $\mathbf{L}$ and replacing the C_{ij} elements with $-S_{ij}$. The rows $(p-1)(j-1)+1, \ldots, (p-1)j$ correspond to messages received by node j (in order), and hence $\mathbf{g}_j$ contains ones at the rows corresponding to messages received by node j, and zeros otherwise. Consider a row corresponding to a message from node i to node j that requires communication from other nodes (excluding j) to node i, this row will therefore contain $-S_{ij}$ where $\mathbf{g}_i$ is equal to 1, except the element corresponding to the message from j to i. Now $\mathbf{A}\mathbf{g}_i$ will be equal to $-(p-2)S_{ij}$ in the rows corresponding to the message from i to j, and zero otherwise. The vector $\mathbf{g}_j$ contains references to rows corresponding to messages received by node j, and since there is only one message from i to j, the non-zero elements of $\mathbf{g}_j$ will overlap with the non-zero elements of $\mathbf{A}\mathbf{g}_i$ at one element, and hence $\mathbf{g}_j'\mathbf{A}\mathbf{g}_i = -(p-2)S_{ij}$ for $j \neq i$. Furthermore, since there is no message from node i to node i, we have that $\mathbf{g}_i'\mathbf{A}\mathbf{g}_i = 0$. We see that $\frac{1}{p-2}\mathbf{G}'\mathbf{A}\mathbf{G} = \mathbf{I} - \mathbf{S}$. Equation (A.7) becomes

$$\det\left(\mathbf{S} - t\mathbf{I} + \mathbf{M}\right) = 0.$$

Since $\mathbf{S}$ is symmetric, it is diagonalisable. There exists an orthogonal matrix $\mathbf{U}$ such that $\mathbf{U}^{-1}\mathbf{S}\mathbf{U} = \mathbf{D}$ is a diagonal matrix. We have

$$\begin{aligned}\det\big(\mathbf{S} - t\mathbf{I} + \mathbf{M}\big) &= \det\big(\mathbf{U}^{-1}(\mathbf{S} - t\mathbf{I} + \mathbf{M})\mathbf{U}\big) \\ &= \det\big(\mathbf{D} - t\mathbf{I} + \mathbf{U}^{-1}\mathbf{M}\mathbf{U}\big).\end{aligned}$$

Recall that $\mathbf{M} = \mathcal{O}(\delta)$, uniformly in t (for $|t| \leq K$), so we also have $\mathbf{U}^{-1}\mathbf{M}\mathbf{U} = \mathcal{O}(\delta)$. Let κ be a constant such that $\|\mathbf{U}^{-1}\mathbf{M}\mathbf{U}\|_\infty \leq \kappa\delta$ (for sufficiently small δ and $|t| \leq K$). If

$$\det\big(\mathbf{D} - t\mathbf{I} + \mathbf{U}^{-1}\mathbf{M}\mathbf{U}\big) = 0,$$

then we must have $|t - d_{ii}| \leq \kappa\delta$ for one of the diagonal entries d_{ii} of $\mathbf{D}$, for otherwise the matrix $\mathbf{D} - t\mathbf{I} + \mathbf{U}^{-1}\mathbf{M}\mathbf{U}$ will be strictly diagonally dominant and thus invertible. The diagonal entries of $\mathbf{D}$ are the eigenvalues $\sigma_1, \sigma_2, \ldots, \sigma_p$ of $\mathbf{S}$, so it follows that $t = \sigma_i + \mathcal{O}(\delta)$. We can deal with the small eigenvalues in the same way and it only remains to determine the entries of $\mathbf{A} - \frac{1}{p-2}\mathbf{A}\mathbf{G}'\mathbf{G}$ (thereby verifying that this matrix is also symmetric and thus diagonalisable). It is easy to verify that $\mathbf{G}'\mathbf{G}$ is a block-diagonal matrix where the blocks are of dimension $(p-1) \times (p-1)$ with all entries equal to 1; in fact, $\mathbf{G}'\mathbf{G} = [\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_p]$ where $\mathbf{B}_i$ is $\mathbf{g}_i$ appended $p-1$ times as columns. Consider a row in $\mathbf{A}$ corresponding to a message from i to j, say $\mathbf{a}'_{ij}$. We have already verified that this row contains $-S_{ij}$ where $\mathbf{g}_i$ is 1, except for the 1 corresponding to the message from j to i. Now $\mathbf{a}'_{ij}\mathbf{G}'\mathbf{G}$ will contain non-zero elements in $\mathbf{a}'_{ij}\mathbf{B}_i$, and these will all equal $-(p-2)S_{ij}$. A row, $\mathbf{b}_{ij}$, of $\frac{1}{p-2}\mathbf{A}\mathbf{G}'\mathbf{G}$ corresponding to a message from i to j will contain $-S_{ij}$ where $\mathbf{g}_i$ equals 1 (even for the message from j to i). Hence, $\mathbf{a}_{ij}$ and $\mathbf{b}_{ij}$ will be identical except for the element corresponding to the message from i to j, where $\mathbf{a}_{ij}$ is zero and $\mathbf{b}_{ij}$ is $-S_{ij}$. A row of $\mathbf{A} - \frac{1}{p-2}\mathbf{A}\mathbf{G}'\mathbf{G}$ corresponding to a message from i to j hence will have one element (at the message from j to i) equal to S_{ij}, and the rest are zero. Furthermore, the row corresponding to the message from j to i will have $S_{ji} = S_{ij}$ as an element in the position corresponding to the message from i to j. Hence $\mathbf{A} - \frac{1}{p-2}\mathbf{A}\mathbf{G}'\mathbf{G}$ is symmetric. In conclusion, the eigenvalues of $\tilde{\mathbf{L}}$ are

- $1 - \sigma_i\delta + \mathcal{O}(\delta^2)$, where $\sigma_1, \sigma_2, \ldots, \sigma_p$ are the eigenvalues of $\mathbf{S}$, and
- $\pm S_{ij}\delta + \mathcal{O}(\delta^2)$, $1 \leq i < j \leq p$.

In particular, the largest eigenvalue of $\tilde{\mathbf{L}}$ is connected to the least eigenvalue $\sigma_{\min}$ of $\mathbf{S}$ by

$$\max\{\mu : \mu \text{ is an eigenvalue of } \tilde{\mathbf{L}}\} = 1 - \sigma_{\min}\delta + \mathcal{O}(\delta^2).$$

Since $\mathbf{S}$ is a positive definite matrix, we know that $\sigma_{\min} > 0$. It follows that

$$\max\{\mu : \mu \text{ is an eigenvalue of } \tilde{\mathbf{L}}\} < 1$$

for sufficiently small δ.

A.3 Proof of Theorem 3

In Theorems 1 and 2 we proved convergence to the following stationary equations:

$$\begin{aligned} Q_{ij} &= \frac{-S_{ij}^2}{\lambda + q_i - Q_{ji}} \\ V_{ij} &= \frac{Q_{ij}}{S_{ij}}(\lambda\mu_i + z_i - V_{ji}) \\ \mu_i &= \frac{\lambda\mu_i + z_i}{\lambda + q_i}, \end{aligned} \tag{A.9}$$

for all i and $j \in \mathcal{N}_i$. Furthermore, $q_i = 1 + \sum_{t\in\mathcal{N}_i} Q_{ti}$ and $z_i = b_i + \sum_{t\in\mathcal{N}_i} V_{ti}$. Using (A.9):

$$z_i + \lambda\mu_i = V_{ji} + \frac{S_{ij}}{Q_{ij}}V_{ij}$$

for all i and $j \in \mathcal{N}_i$. For any $k \in \mathcal{N}_i$ we can write

$$S_{ki}(z_i + \lambda\mu_i) = S_{ki}V_{ji} + S_{ki}\frac{S_{ij}}{Q_{ij}}V_{ij} = S_{ki}V_{ki} + \frac{S_{ik}^2}{Q_{ik}}V_{ik}.$$

Furthermore, since $\frac{S_{ik}^2}{Q_{ik}} = Q_{ki} - (q_i + \lambda_i)$, we have

$$S_{ki}(z_i + \lambda\mu_i) = S_{ki}V_{ki} + (Q_{ki} - (q_i + \lambda_i))V_{ik}. \tag{A.10}$$

Dividing (A.10) by $q_i + \lambda$ gives

$$S_{ki}\mu_i = \frac{1}{q_i + \lambda}S_{ki}V_{ki} + \frac{Q_{ki}}{q_i + \lambda}V_{ik} - V_{ik}. \tag{A.11}$$

Further simplification can be done by noting that $S_{ki}V_{ki} = Q_{ki}(\lambda u_k + z_k - V_{ik})$, and substituting into (A.11):

$$\begin{aligned} S_{ki}\mu_i &= \frac{Q_{ki}}{q_i + \lambda}(\lambda\mu_k + z_k - V_{ik}) + \frac{Q_{ki}}{q_i + \lambda}V_{ik} - V_{ik} \\ &= \frac{Q_{ki}}{q_i + \lambda}(\lambda\mu_k + z_k) - \frac{Q_{ki}}{q_i + \lambda}V_{ik} + \frac{Q_{ki}}{q_i + \lambda}V_{ik} - V_{ik} \\ &= \frac{Q_{ki}}{q_i + \lambda}(\lambda\mu_k + z_k) - V_{ik}. \end{aligned} \tag{A.12}$$

Summing $S_{ki}\mu_i$ over i, substituting (A.12) for $i \in \mathcal{N}_k$, gives

$$\sum_{i\in\mathcal{N}_k\cup k} S_{ki}\mu_i = \mu_k + (\lambda\mu_k + z_k)\sum_{i\in\mathcal{N}_k}\frac{Q_{ki}}{q_i + \lambda} - \sum_{i\in\mathcal{N}_k} V_{ik}. \tag{A.13}$$

Since $Q_{ik}(\lambda + q_i - Q_{ki}) = -S_{ik}^2 = -S_{ki}^2 = Q_{ki}(\lambda + q_k - Q_{ik})$,

$$\frac{Q_{ik}}{\lambda + q_k} = \frac{Q_{ki}}{\lambda + q_i}. \tag{A.14}$$

Substituting (A.14) into (A.13):

$$\begin{aligned}
\sum_{i \in \mathcal{N}_k \cup k} S_{ki}\mu_i &= \mu_k + (\lambda\mu_k + z_k) \sum_{i \in \mathcal{N}_k} \frac{Q_{ik}}{\lambda + q_k} - \sum_{i \in \mathcal{N}_k} V_{ik} \\
&= \mu_k + \mu_k \sum_{i \in \mathcal{N}_k} Q_{ik} - \sum_{i \in \mathcal{N}_k} V_{ik} \\
&= \mu_k + \mu_k(q_k - 1) - (z_k - b_k) \\
&= q_k\mu_k - z_k + b_k. \qquad \text{(A.15)}
\end{aligned}$$

Finally, since $\mu_k = \frac{\lambda\mu_k + z_k}{\lambda + q_k}$, we have that $\mu_k(\lambda + q_k) = \lambda\mu_k + z_k$ and

$$q_k\mu_k = z_k. \tag{A.16}$$

Substituting (A.16) into (A.15) completes the proof.

A.4 Simulation Information

A.4.1 Simulation Scheme

We briefly describe the simulation scheme (Bach *et al.*, 2011) used in our empirical work. This simulation scheme also relates GaBP to least-squares estimation under the linear model. In order to apply sGaBP we need to generate a positive definite symmetric precision matrix ($\mathbf{S}$) and potential vector ($\boldsymbol{b}$). One way to do this is to generate a data structure according to the linear model with n observations and p inputs. This yields a design matrix $\mathbf{X} : n \times p$ and a response vector $\mathbf{y} : n \times 1$. We then form the sample correlation matrix $\mathbf{S} = \mathbf{X}'\mathbf{X}$, where we assume the columns of $\mathbf{X}$ are standardised to have zero mean and unity L_2 norm. We assume the same for $\mathbf{y}$ and form the sample correlation vector, $\boldsymbol{b} = \mathbf{X}'\mathbf{y}$. Explanatory variables are generated from $N(\mathbf{0}, \frac{1}{n}\mathbf{I}_p)$, where n is the number of observations and p the number of explanatory variables. The generated explanatory variables are stored in $\mathbf{X}$. Coefficients are generated, $\beta_i \sim$ iid $N(0, 1)$, and sparsity is introduced by randomly selecting half of the β_i's and setting these equal to zero. Observations of the response are generated, $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\epsilon_i \sim$ iid $N(0, \sigma^2)$ and $\sigma^2 = 0.01 \times \frac{||\mathbf{X}\boldsymbol{\beta}||^2}{n}$. All variables were standardised to have zero mean and unit L_2 norm, and we form $\mathbf{S} = \mathbf{X}'\mathbf{X}$ and $\boldsymbol{b} = \mathbf{X}'\mathbf{y}$. We used $n = p$ throughout the empirical section. The matrix, $\mathbf{S}$, was ensured to be positive definite by regulating its zero-diagonal spectral radius using the method discussed in the next section. The potential vector $\boldsymbol{b}$ remains unchanged. We then apply sGaBP on a multivariate Gaussian with precision matrix $\mathbf{S}$ and potential vector $\boldsymbol{b}$.

A.4.2 Regulating the Zero-diagonal Spectral Radius

Suppose we have a precision matrix, $\mathbf{S} : p \times p$, normalised to have ones along its diagonal. Set $\mathbf{R} = \mathbf{I}_p - \mathbf{S}$ and let $\tau_i : i = 1, 2, \ldots, p$ be the eigenvalues of $\mathbf{R}$. Suppose we wish to find a new precision matrix, $\mathbf{S}^*$, with zero-diagonal spectral radius set to a specified value (say α). First we compute the eigen-decomposition of $\mathbf{R}$,

$$\mathbf{R} = \mathbf{V}\mathbf{D}\mathbf{V}',$$

where $\mathbf{V}'\mathbf{V} = \mathbf{V}\mathbf{V}' = \mathbf{I}$ and $\mathbf{D} = \text{diag}(\tau_1, \ldots, \tau_p)$. We form a new diagonal matrix, $\mathbf{D}^* = \frac{\alpha}{\tilde{\rho}(\mathbf{S})}\mathbf{D}$, and set $\mathbf{R}^* = \mathbf{V}\mathbf{D}^*\mathbf{V}'$ and $\mathbf{S}^* = \mathbf{I} - \mathbf{R}^*$. We now show that $\mathbf{S}^*$ is a valid precision matrix with diagonal entries equal to one if $\alpha < 1$. Since $\mathbf{S}$ is a normalised precision matrix, the diagonal of $\mathbf{R}$ will contain only zeros; the same is true for $\mathbf{R}^*$ (being a scalar multiple of $\mathbf{R}$), and therefore the diagonal of $\mathbf{S}^*$ will contain only ones. Suppose $\lambda_i, i = 1, 2, \ldots, p$ and $\lambda_i^*, i = 1, 2, \ldots, p$ represent the eigenvalues of $\mathbf{S}$ and $\mathbf{S}^*$ respectively. The following holds:

$$\begin{aligned}
\lambda_i^* &= 1 - \frac{\alpha}{\tilde{\rho}(\mathbf{S})}(1 - \lambda_i) \\
&= 1 - \alpha\frac{1 - \lambda_i}{\max_j\{|1 - \lambda_j|\}} \\
&= 1 - \alpha \times \text{sign}(1 - \lambda_i) \times \frac{|1 - \lambda_i|}{\max_j\{|1 - \lambda_j|\}}.
\end{aligned}$$

Since $\frac{|1-\lambda_i|}{\max_j\{|1-\lambda_j|\}} \leq 1$, we have that $1 - \alpha \leq \lambda_i^* \leq 1 + \alpha$. If $0 \leq \alpha < 1$, then $\mathbf{S}^*$ will be positive definite. In our simulations, when $\alpha > 1$, we check that $\mathbf{S}^*$ is positive definite by investigating its eigenvalues. If $\mathbf{S}^*$ is not positive definite, we simulate a new $\mathbf{S}$. We continue this procedure until a positive definite $\mathbf{S}^*$, with the given zero-diagonal spectral radius, is generated. The value of α can be varied according to the simulation specifics. The matrix $\mathbf{S}^*$ is supplied to an algorithm as a precision matrix.

Appendix B

Proofs for Chapter 3

We start by proving three lemmas that are used in the proofs contained in this appendix.

Lemma 3 *Consider any (possibly non-symmetric) matrix* $\mathbf{A} : k \times k$. *Let* $\mathbf{T}_n(\mathbf{A})$ *be the precision matrix of the UWT, of depth* n, *constructed for any cluster of* $\mathbf{A}$. *For all* n *we have that:*

$$||\mathbf{T}_n(\mathbf{A})||_\infty \leq ||\mathbf{A}||_\infty, \tag{B.1}$$

with equality if the UWT is of sufficient depth such that all clusters are referenced before the terminal later.

Proof.

Consider a node of the UWT and suppose that this node has a reference to cluster s:

1. If this node is before the terminal layer, then its neighbourhood in the UWT will reference each cluster in $\mathcal{N}_s$ exactly once. Hence, the infinity norm of the row-block of $\mathbf{T}_n(\mathbf{A})$ corresponding to this node will be equal to the infinity norm of the row-block of $\mathbf{A}$ corresponding to the clusters in $\mathcal{N}_s$.

2. If this node is in the terminal layer, then its neighbourhood in the UWT will reference each cluster in $\mathcal{N}_s$ at most once (certain clusters will not be referenced). Hence, the infinity norm of the row-block of $\mathbf{T}_n(\mathbf{A})$ corresponding to this node will be at most equal to the infinity norm of the row-block of $\mathbf{A}$ corresponding to the clusters in $\mathcal{N}_s$.

The result follows directly from these points.

Lemma 4 *Consider an irreducible matrix* $\mathbf{A} : t \times t$ *consisting of non-negative elements. There exist an eigenvector* $\mathbf{v}$, *containing only positive elements,*

such that $\mathbf{Av} = \rho(\mathbf{A})\mathbf{v}$ *(the spectral radius of* $\mathbf{A}$ *is also an eigenvalue of* $\mathbf{A}$*). Furthermore, if* $\mathbf{D} = diag(\mathbf{v})$*, then:*

$$\rho(\mathbf{A}) = ||\mathbf{D}^{-1}\mathbf{AD}||_{\infty}. \tag{B.2}$$

Proof.

First we note that the Perron-Frobenius theorem guarantees the existence of a vector $\mathbf{v}$, consisting only of positive elements, such that $\mathbf{Av} = \rho(\mathbf{A})\mathbf{v}$. Let $\mathbf{A} = [a_{st}]$ and $\mathbf{v} = (v_1, v_2, \ldots, v_t)'$. Note that,

$$\mathbf{D}^{-1}\mathbf{AD} = \left[\frac{v_s a_{ts}}{v_t}\right], \tag{B.3}$$

and all elements of $\mathbf{D}^{-1}\mathbf{AD}$ are non-negative. The consequence is that the sum of the absolute elements of row t of $\mathbf{D}^{-1}\mathbf{AD}$ is,

$$\sum_{s=1}^{t} \frac{v_s a_{ts}}{v_t} = \frac{1}{v_t}\sum_{s=1}^{t} v_s a_{ts} = \frac{v_t \rho(\mathbf{A})}{v_t} = \rho(\mathbf{A}), \tag{B.4}$$

since $\mathbf{v}$ is the eigenvector of $\mathbf{A}$ corresponding to the eigenvalue $\rho(\mathbf{A})$. The row-sums of the absolute elements of $\mathbf{D}^{-1}\mathbf{AD}$ are all equal to $\rho(\mathbf{A})$ and hence, $||\mathbf{D}^{-1}\mathbf{AD}||_{\infty} = \rho(\mathbf{A})$.

Lemma 5 *Consider a matrix* $\mathbf{A} : t \times t$ *and let* $\mathbf{B} : t \times t$ *be any invertible matrix. We have that*

$$\rho(\mathbf{A}) = \rho(\mathbf{B}^{-1}\mathbf{AB}). \tag{B.5}$$

Proof.

This follows from the well known result that $\mathbf{A}$ and $\mathbf{B}^{-1}\mathbf{AB}$ have the same set of eigenvalues.

Proof of Theorem 7

We assume that the matrix $\mathbf{S}$ is irreducible – if it is not, then GaBP can be applied to separate irreducible matrices – and scaled to have ones along its diagonal. Set $\mathbf{S} = \mathbf{I} - \mathbf{R}$, and therefore $|\mathbf{R}|$ is also irreducible. By Lemma 4, we can find a vector $\mathbf{v}$, consisting only of positive elements, such that $|\mathbf{R}|\mathbf{v} = \rho(|\mathbf{R}|)\mathbf{v}$. Set $\mathbf{D_v} = \text{diag}(\mathbf{v})$ and consider constructing a UWT for $\mathbf{D_v}^{-1}|\mathbf{R}|\mathbf{D_v}$ according to cluster 1 (similar results hold for other clusters). Define $\tilde{\mathbf{v}}_n = \mathbf{E}_n\mathbf{v}$ and $\mathbf{D}_{\tilde{\mathbf{v}}_n} = \text{diag}(\tilde{\mathbf{v}}_n)$. We note the following important facts:

1. By Lemma 3 we have that $||\mathbf{T}_n(\mathbf{D_v}^{-1}|\mathbf{R}|\mathbf{D_v})||_{\infty} \leq ||\mathbf{D_v}^{-1}|\mathbf{R}|\mathbf{D_v}||_{\infty}$.

2. By Lemma 4 we have $\rho(|\mathbf{R}|) = ||\mathbf{D}_{\mathbf{v}}^{-1}|\mathbf{R}|\mathbf{D}_{\mathbf{v}}||_\infty$ and therefore $||\mathbf{T}_n(\mathbf{D}_{\mathbf{v}}^{-1}|\mathbf{R}|\mathbf{D}_{\mathbf{v}})||_\infty \leq \rho(|\mathbf{R}|)$.

3. Recall that $\mathbf{R} = \mathbf{I} - \mathbf{S}$. We see that the UWT for cluster 1, with a depth of n, constructed from $|\mathbf{R}|$ is $|\mathbf{R}_n|$, i.e. $|\mathbf{R}_n| = \mathbf{T}_n(|\mathbf{R}|)$ (this can be validated using our running example).

4. It can be shown that $\mathbf{T}_n(\mathbf{D}_{\mathbf{v}}^{-1}|\mathbf{R}|\mathbf{D}_{\mathbf{v}}) = \mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}\mathbf{T}_n(|\mathbf{R}|)\mathbf{D}_{\tilde{\mathbf{v}}_n} = \mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}$ (this can be validated using our running example).

In conclusion, we find that:

$$||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \leq \rho(|\mathbf{R}|). \tag{B.6}$$

Finally we obtain,

$$\rho(|\mathbf{R}_n|) = \rho(\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}) \leq ||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \leq \rho(|\mathbf{R}|),$$

by Lemma 5 and the fact that the eigenvalues of a matrix are bounded by its infinity norm.

Proof of Theorem 8

Consider

$$\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^k\mathbf{D}_{\tilde{\mathbf{v}}_n}\mathbf{E}_n = \mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^{k-1}\mathbf{D}_{\tilde{\mathbf{v}}_n}\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}. \tag{B.7}$$

From Equation (B.7),

$$\begin{aligned} ||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^k\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty &\leq ||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^{k-1}\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \\ &\leq \rho(|\mathbf{R}|)||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^{k-1}\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \end{aligned} \tag{B.8}$$

If we apply Inequality (B.8) recursively, we see that

$$||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^k\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \leq [\rho(|\mathbf{R}|)]^k$$

and

$$\begin{aligned} |||\mathbf{R}_n|^k||_\infty &= ||\mathbf{D}_{\tilde{\mathbf{v}}_n}\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^k\mathbf{D}_{\tilde{\mathbf{v}}_n}\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}||_\infty \\ &\leq ||\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}||_\infty||\mathbf{D}_{\tilde{\mathbf{v}}_n}^{-1}|\mathbf{R}_n|^k\mathbf{D}_{\tilde{\mathbf{v}}_n}||_\infty \\ &\leq \frac{\max_i\{v_i\}}{\min_j\{v_j\}}[\rho(|\mathbf{R}|)]^k. \end{aligned}$$

Setting $\kappa_1 = \frac{\max_i\{v_i\}}{\min_j\{v_j\}}$,

$$||\mathbf{R}_n^k||_\infty \leq |||\mathbf{R}_n|^k||_\infty \leq \kappa_1[\rho(|\mathbf{R}|)]^k.$$

If we assume that $\rho(|\mathbf{R}|) < 1$, then we can write $\mathbf{T}_n^{-1} = \mathbf{T}_n^{-1}(\mathbf{S})$ as a Neumann power series:

$$\mathbf{T}_n^{-1} = \sum_{i=0}^{\infty} \mathbf{R}_n^i. \tag{B.9}$$

The series in Equation (B.9) implies

$$\begin{aligned} ||\mathbf{T}_n^{-1}||_\infty &\leq \sum_{i=0}^{\infty} ||\mathbf{R}_n^i||_\infty \\ &\leq \kappa_1 \sum_{i=0}^{\infty} [\rho(|\mathbf{R}|)]^i \\ &= \frac{\kappa_1}{1 - \rho(|\mathbf{R}|)} = \kappa_2 \end{aligned}$$

Proof of Theorem 9

Consider the following recursive expansions:

$$\mathbf{T}_{n+1} = \begin{bmatrix} \mathbf{T}_n & \mathbf{T}_{1n} \\ \mathbf{T}_{n1} & \mathbf{U}_{nn} \end{bmatrix}$$

$$\tilde{\mathbf{E}}_{n+1} = \begin{bmatrix} \tilde{\mathbf{E}}_n \\ \mathbf{W}_n \end{bmatrix}$$

for certain matrices $\mathbf{T}_{1n}$, $\mathbf{U}_{nn}$ and $\mathbf{W}_n$. We have

$$\mathbf{T}_{n+1}^{-1} = \begin{bmatrix} \tilde{\mathbf{T}}_n^{-1} & -\tilde{\mathbf{T}}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1} \\ \cdots & \cdots \end{bmatrix}$$

where the $\ldots$ represents an irrelevant part of the matrix and $\tilde{\mathbf{T}}_n = \mathbf{T}_n - \mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{T}_{n1}$. Consider

$$\begin{aligned} \mathbf{Z}_{n+1} &= \mathbf{G}'_{n+1}\mathbf{T}_{n+1}^{-1}\tilde{\mathbf{E}}_{n+1} \\ &= \begin{bmatrix} \mathbf{G}'_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{T}}_n^{-1} & -\tilde{\mathbf{T}}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1} \\ \cdots & \cdots \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{E}}_n \\ \mathbf{W}_n \end{bmatrix} \\ &= \mathbf{G}'_n\tilde{\mathbf{T}}_n^{-1}\tilde{\mathbf{E}}_n - \mathbf{G}'_n\tilde{\mathbf{T}}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n. \end{aligned} \tag{B.10}$$

It can be shown that

$$\begin{aligned} \tilde{\mathbf{T}}_n^{-1} &= \mathbf{T}_n^{-1} + \mathbf{T}_n^{-1}\mathbf{T}_{1n}[\mathbf{U}_{nn} - \mathbf{T}_{n1}\mathbf{T}_n^{-1}\mathbf{T}_{1n}]^{-1}\mathbf{T}_{n1}\mathbf{T}_n^{-1} \\ &= \mathbf{T}_n^{-1} + \mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1} \end{aligned}$$

where $\mathbf{V}_n = [\mathbf{U}_{nn} - \mathbf{T}_{n1}\mathbf{T}_n^{-1}\mathbf{T}_{1n}]^{-1}\mathbf{T}_{n1}$. Furthermore,

$$\begin{aligned} \mathbf{G}'_n\tilde{\mathbf{T}}_n^{-1}\tilde{\mathbf{E}}_n &= \mathbf{G}'_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n + \mathbf{G}'_n\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n \\ &= \mathbf{Z}_n + \mathbf{G}'_n\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n. \end{aligned}$$

Considering again Equation (B.10),

$$\begin{aligned}\mathbf{Z}_{n+1} &= \mathbf{Z}_n + \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n - \mathbf{G}_n'\tilde{\mathbf{T}}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n \\ &= \mathbf{Z}_n + \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n - \mathbf{G}_n'[\mathbf{T}_n^{-1} + \mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}]\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n \\ &= \mathbf{Z}_n + \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n - \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n - \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{V}_n\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n \\ &= \mathbf{Z}_n + \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}[\mathbf{V}_n\mathbf{T}_n^{-1}\tilde{\mathbf{E}}_n - \mathbf{V}_n\mathbf{T}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n - \mathbf{U}_{nn}^{-1}\mathbf{W}_n] \\ &= \mathbf{Z}_n + \mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}[\mathbf{V}_n\mathbf{T}_n^{-1}(\tilde{\mathbf{E}}_n - \mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n) - \mathbf{U}_{nn}^{-1}\mathbf{W}_n]. \qquad \text{(B.11)}\end{aligned}$$

Note that the sparsity pattern of $\mathbf{T}_{1n}$ is similar to that of $\mathbf{L}_n$ in the sense that it contains zeros in all the rows except for the rows corresponding to the terminal layer of the UWT (of depth n). Similar to Equation (3.37) we can show that,

$$\mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n} = \mathbf{G}_n'\mathbf{R}_n^{n-1}\mathbf{T}_n^{-1}\mathbf{T}_{1n}, \qquad \text{(B.12)}$$

and hence:

$$||\mathbf{G}_n'\mathbf{T}_n^{-1}\mathbf{T}_{1n}||_\infty \leq \kappa_1\kappa_2||\mathbf{T}_{1n}||_\infty[\rho(|\mathbf{R}|)]^{n-1}. \qquad \text{(B.13)}$$

By Equations (B.11) and (B.13) we see that:

$$||\mathbf{Z}_{n+1}-\mathbf{Z}_n||_\infty \leq \kappa_1\kappa_2||\mathbf{T}_{1n}||_\infty||\mathbf{V}_n\mathbf{T}_n^{-1}(\tilde{\mathbf{E}}_n-\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n)-\mathbf{U}_{nn}^{-1}\mathbf{W}_n||_\infty\times[\rho(|\mathbf{R}|)]^{n-1} \qquad \text{(B.14)}$$

We now make some comments on the matrices present in Equation (B.14).

1. The matrix $\mathbf{T}_{1n}$ represents the links between parents in layer n and their children in layer $n+1$. Note that a parent in layer n has links to nodes in layer $n+1$ corresponding to its neighbours only, hence $||\mathbf{T}_{1n}||_\infty \leq \max_{i\neq j}\{||\mathbf{S}_{ij}||_\infty\}$.

2. The matrix $\mathbf{U}_{nn}$ is block-diagonal, with diagonal blocks corresponding to the submatrices of $\mathbf{S}$ as defined by their clusters. Therefore, $||\mathbf{U}_{nn}^{-1}||_\infty \leq \max_i\{||\mathbf{S}_{ii}^{-1}||_\infty\}$ and $||\mathbf{U}_{nn}||_\infty \leq \max_i\{||\mathbf{S}_{ii}||_\infty\}$.

3. Because $||\mathbf{T}_s^{-1}||_\infty \leq \kappa_2$ for all s, we have $||\tilde{\mathbf{T}}_n^{-1}||_\infty \leq \kappa_2$ since $\tilde{\mathbf{T}}_n^{-1}$ is a block of $\mathbf{T}_{n+1}^{-1}$.

4. Consider $\mathbf{V}_n = \mathbf{U}_{nn}^{-1}\mathbf{T}_{n1} + \mathbf{U}_{nn}^{-1}\mathbf{T}_{n1}[\mathbf{T}_n - \mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{T}_{n1}]^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{T}_{n1} = \mathbf{U}_{nn}^{-1}\mathbf{T}_{n1} + \mathbf{U}_{nn}^{-1}\mathbf{T}_{n1}\tilde{\mathbf{T}}_n^{-1}\mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{T}_{n1}$. Since all the matrices in this expression have an infinity norm bounded by some constant which does not depend on n, there will also be a constant (independent of n) that bounds the infinity norm of $\mathbf{V}_n$.

5. Note that $||\tilde{\mathbf{E}}_s||_\infty \leq ||\mathbf{E}_s||_\infty = 1$ (see Section 3.5.2) for all s, and therefore $||\tilde{\mathbf{E}}_n||_\infty \leq 1$. Since $\mathbf{W}_n$ is a row-block of $\tilde{\mathbf{E}}_{n+1}$ we see that $||\mathbf{W}_n||_\infty \leq 1$.

The above considerations imply the existence of a constant K (independent of n) such that $\kappa_1\kappa_2||\mathbf{T}_{1n}||_\infty||\mathbf{V}_n\mathbf{T}_n^{-1}(\tilde{\mathbf{E}}_n - \mathbf{T}_{1n}\mathbf{U}_{nn}^{-1}\mathbf{W}_n) - \mathbf{U}_{nn}^{-1}\mathbf{W}_n||_\infty \leq K$. Therefore, $||\mathbf{Z}_{n+1} - \mathbf{Z}_n||_\infty \leq K[\rho(|\mathbf{R}|)]^{n-1}$, hence $\mathbf{Z}_n$ is a Cauchy sequence and will converge to a specified matrix $\mathbf{Z}_\infty$.

Appendix C

Proofs for Chapter 4

Proof of Theorem 13

We start with the following:

Lemma 6 *There exists a constant $K > 0$ such that, for sufficiently small δ, each eigenvalue x of $\tilde{\mathbf{L}}$ either satisfies $|x| \leq K\delta$ or $|x-1| \leq K\delta$.*

We reason by contradiction and assume that there is an eigenvalue for which $|x| > K\delta$ and $|x-1| > K\delta$. Consider $||\tilde{\mathbf{L}}_{11}||_\infty = ||\mathbf{L}_{11}||_\infty = \delta||\mathbf{M}_{11}||_\infty + \mathcal{O}(\delta^2)$. If we choose K large enough (e.g. $K \geq 1 + ||\mathbf{M}_{11}||_\infty$), then $||\mathbf{L}_{11}||_\infty < K\delta + \mathcal{O}(\delta^2)$ and $||\mathbf{L}_{11}||_\infty < |x|$ for sufficiently small δ. Therefore, x is not an eigenvalue of $\mathbf{L}_{11}$ and $x\mathbf{I}_{m_1} - \mathbf{L}_{11}$ will be invertible. We can now apply the Schur complement on $\tilde{\mathbf{L}}$ to obtain:

$$\begin{aligned}\det(x\mathbf{I}_{m_2} - \tilde{\mathbf{L}}) &= \det(x\mathbf{I}_{m_1} - \mathbf{L}_{11}) \\ &\quad \times \det(x\mathbf{I}_k - \mathbf{L}_{22} - \frac{1}{p-2}\mathbf{L}_{22}\mathbf{H}'(x\mathbf{I}_{m_1} - \mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{H}). \quad \text{(C.1)}\end{aligned}$$

It remains to show that the second determinant is not equal to zero.

$$\begin{aligned}&x\mathbf{I}_k - \mathbf{L}_{22} - \frac{1}{p-2}\mathbf{L}_{22}\mathbf{H}'(x\mathbf{I}_{m_1} - \mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{H} \\ &= (x-1)\mathbf{I}_k - (\mathbf{L}_{22} - \mathbf{I}_k) - \frac{1}{x(p-2)}\mathbf{L}_{22}\mathbf{H}'(\mathbf{I}_{m_1} - \frac{1}{x}\mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{H} \\ &= (x-1)\mathbf{I}_k + \mathbf{H}_1 + \frac{1}{x}\mathbf{H}_2, \quad \text{(C.2)}\end{aligned}$$

where $\mathbf{H}_1 = -(\mathbf{L}_{22} - \mathbf{I}_k)$ and $\mathbf{H}_2 = -\frac{1}{(p-2)}\mathbf{L}_{22}\mathbf{H}'(\mathbf{I}_{m_1} - \frac{1}{x}\mathbf{L}_{11})^{-1}\mathbf{L}_{11}\mathbf{H}$. Consider

$$\begin{aligned}||\mathbf{H}_1||_\infty &= ||\mathbf{L}_{22} - \mathbf{I}_k||_\infty \\ &= ||\delta\mathbf{M}_{22} + \mathcal{O}(\delta^2)||_\infty \\ &= \delta||\mathbf{M}_{22}||_\infty + \mathcal{O}(\delta^2) \\ &\leq \delta\kappa_1, \quad \text{(C.3)}\end{aligned}$$

where δ is sufficiently small and, e.g. $\kappa_1 \geq ||\mathbf{M}_{22}||_\infty + 1$. Furthermore,

$$\begin{aligned}(\mathbf{I} - \frac{1}{x}\mathbf{L}_{11})^{-1}\mathbf{L}_{11} &= (\mathbf{I} - \frac{\delta}{x}\mathbf{M}_{11} + \mathcal{O}(\delta^2))^{-1}(\delta\mathbf{M}_{11} + \mathcal{O}(\delta^2)) \\ &= \delta(\mathbf{I} - \frac{\delta}{x}\mathbf{M}_{11})^{-1}\mathbf{M}_{11} + \mathcal{O}(\delta^2) \\ &= \delta(\mathbf{I} + \mathcal{O}(\delta))\mathbf{M}_{11} + \mathcal{O}(\delta^2) \\ &= \delta\mathbf{M}_{11} + \mathcal{O}(\delta^2). \end{aligned} \tag{C.4}$$

Since $\mathbf{L}_{22} = \mathbf{I}_k + \mathcal{O}(\delta)$, we see that

$$-\mathbf{H}_2 = \frac{\delta}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H} + \mathcal{O}(\delta^2), \tag{C.5}$$

and $||\mathbf{H}_2||_\infty \leq \kappa_2\delta$ for sufficiently small δ and, e.g., $\kappa_2 \geq \frac{1}{p-2}||\mathbf{H}'\mathbf{M}_{11}\mathbf{H}||_\infty + 1$. Suppose that $|x| \geq 0.5$, then

$$||\mathbf{H}_1 + \frac{1}{x}\mathbf{H}_2||_\infty \leq \delta\kappa_1 + \delta\frac{\kappa_2}{|x|} \leq \delta(\kappa_1 + 2\kappa_2) < K\delta < |x-1| \tag{C.6}$$

for δ sufficiently small and $K > \kappa_1 + 2\kappa_2$. If $|x| < 0.5$,

$$||\mathbf{H}_1 + \frac{1}{x}\mathbf{H}_2||_\infty \leq \delta\kappa_1 + \delta\frac{\kappa_2}{|x|} \leq \delta\kappa_1 + \frac{\delta\kappa_2}{\delta K} = \delta\kappa_1 + \frac{\kappa_2}{K} < 0.5 \leq |x-1| \tag{C.7}$$

for δ sufficiently small and K sufficiently large (e.g. $K > 2\kappa_2$). We have that, for sufficiently small δ and sufficiently large K, $x-1$ will not be an eigenvalue of $\mathbf{H}_1 + \mathbf{H}_2$, and the second determinant in Equation (C.1) will not be zero. Therefore x cannot be an eigenvalue of $\tilde{\mathbf{L}}$ and, by contradiction, the statement is proved.

We first consider the eigenvalues that are close to one. Set $x = 1 - \delta t$ for some t where $|t| < K$. For sufficiently small δ, x will not be an eigenvalue of $\mathbf{L}_{11}$, and therefore $x\mathbf{I}_{m_1} - \mathbf{L}_{11}$ will be invertible. Application of the Schur complement on $\tilde{\mathbf{L}}$ yields

$$\det(\tilde{\mathbf{L}}) = \det(x\mathbf{I}_{m_1} - \mathbf{L}_{11})\det(-\delta t + \mathbf{H}_1 + \frac{1}{x}\mathbf{H}_2) \tag{C.8}$$

(see Equation (C.2)). Since $\mathbf{H}_1 = \delta\mathbf{M}_{22} + \mathcal{O}(\delta^2)$ and $\mathbf{H}_2 = -\frac{\delta}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H} + \mathcal{O}(\delta^2)$, we see that the second determinant of Equation (C.8) becomes

$$\begin{aligned}&\det(\delta(\mathbf{M}_{22} - t\mathbf{I}_k) - \frac{\delta}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H} + \mathcal{O}(\delta^2)) \\ &= \det((\mathbf{M}_{22} - t\mathbf{I}_k) - \frac{1}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H} + \mathcal{O}(\delta)) \\ &= \det(-t\mathbf{I}_k + \mathbf{M}_{22} - \frac{1}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H} + \mathbf{M}) \\ &= \det(t\mathbf{I}_k - \left[\mathbf{M}_{22} - \frac{1}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H}\right] + \mathbf{M}), \end{aligned} \tag{C.9}$$

where $\mathbf{M} = \mathcal{O}(\delta)$. As $\delta \to 0$ we obtain the characteristic equation of $\left[\mathbf{M}_{22} - \frac{1}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H}\right]$. Its k solutions (counted with multiplicity) give rise to k branches $t_1(\delta), t_2(\delta), \ldots, t_k(\delta)$ that solve the implicit equation in Equation (C.9).

Considering the eigenvalues that are close to zero, we set $x = \delta t$ for some t with $|t| < K$. For δ sufficiently small, x will not be an eigenvalue of $\mathbf{L}_{22}$, and hence $x\mathbf{I}_k - \mathbf{L}_{22}$ will be invertible. A second application of the Schur complement (with respect to the diagonal block $\mathbf{L}_{22}$) gives

$$\det(\tilde{\mathbf{L}}) = \det(x\mathbf{I}_k - \mathbf{L}_{22})\det(x\mathbf{I}_{m_1} - \mathbf{L}_{11} - \frac{1}{p-2}\mathbf{L}_{11}\mathbf{H}(x\mathbf{I}_k - \mathbf{L}_{22})^{-1}\mathbf{L}_{22}\mathbf{H}'). \tag{C.10}$$

First note that $x\mathbf{I}_{m_1} - \mathbf{L}_{11} = \delta t\mathbf{I}_{m_1} - \delta\mathbf{M}_{11} + \mathcal{O}(\delta^2)$ and $(x\mathbf{I}_k - \mathbf{L}_{22})^{-1} = -\mathbf{I}_k + \mathcal{O}(\delta)$. Hence $\mathbf{L}_{11}\mathbf{H}(x\mathbf{I}_k - \mathbf{L}_{22})^{-1} = \delta\mathbf{M}_{11}\mathbf{H} + \mathcal{O}(\delta^2)$ and

$$\frac{1}{p-2}\mathbf{L}_{11}\mathbf{H}(x\mathbf{I}_k - \mathbf{L}_{22})^{-1}\mathbf{L}_{22}\mathbf{H}' = -\frac{\delta}{p-2}\mathbf{M}_{11}\mathbf{H}\mathbf{H}' + \mathcal{O}(\delta^2). \tag{C.11}$$

The second determinant of Equation (C.10) becomes

$$\det(t\mathbf{I}_{m_1} - \mathbf{M}_{11}[\mathbf{I}_{m_1} - \frac{1}{p-2}\mathbf{H}\mathbf{H}'] + \tilde{\mathbf{M}}), \tag{C.12}$$

where $\tilde{\mathbf{M}} = \mathcal{O}(\delta)$. As $\delta \to 0$, Equation C.12 converges to the characteristic function of $\mathbf{M}_{11}[\mathbf{I}_{m_1} - \frac{1}{p-2}\mathbf{H}\mathbf{H}']$. This gives rise to $k^2 - k$ branches $\bar{t}_i(\delta) : i = 1, 2, \ldots, k^2 - k$, which solve the implicit equation in Equation (C.12).

For our example, the matrix $\left[\mathbf{M}_{22} - \frac{1}{p-2}\mathbf{H}'\mathbf{M}_{11}\mathbf{H}\right]$ can easily be seen to be equal to $\mathbf{S}$ and this holds for the general case as well. The matrix $\mathbf{W} = \mathbf{M}_{11}[\mathbf{I}_{m_1} - \frac{1}{p-2}\mathbf{H}\mathbf{H}']$ has a more complicated construction. For our example, we see that $\mathbf{W}$ is symmetric and that $\mathbf{W}'\mathbf{W}$ is a block-diagonal matrix with a diagonal block for each of $\mathbf{S}_{ij}\mathbf{S}_{ji}$ where $i \neq j$. Again, this holds in general.

Consider Equation (C.9) set equal to zero,

$$\det(t\mathbf{I}_k - \mathbf{S} + \mathbf{M}) = 0, \tag{C.13}$$

where $\mathbf{M} = \mathcal{O}(\delta)$. Let $\mathbf{V}$ be the matrix of eigenvectors that diagonalises $\mathbf{S}$. Pre- and post-multiplying Equation (C.13) by $\mathbf{V}$ and $\mathbf{V}'$ respectively, we obtain

$$\det(t\mathbf{I}_k - \mathbf{\Lambda} + \mathbf{V}\mathbf{M}\mathbf{V}') = 0, \tag{C.14}$$

where $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\mathbf{S}$ on its diagonal. Since $\mathbf{M} = \mathcal{O}(\delta)$, we have that $\mathbf{V}\mathbf{M}\mathbf{V}' = \mathcal{O}(\delta)$. This guarantees the existence

of a constant κ_3 such that $||\mathbf{VMV}'||_\infty \leq \kappa_3\delta$ for sufficiently small δ. In order for the matrix in the determinant of Equation (C.13) to be singular, we must have that $|t - \Lambda_{ii}| \leq \kappa_3\delta$, otherwise this matrix will be strictly diagonally dominant. The diagonal entries of $\mathbf{\Lambda}$ are the eigenvalues $\sigma_i : i = 1, 2, \ldots, k$ of $\mathbf{S}$, and therefore $t = \sigma_i + \mathcal{O}(\delta)$.

The smaller eigenvalues can be dealt with in a similar way where we need to find the eigenvalues of $\mathbf{W}$. Since $\mathbf{W}'\mathbf{W}$ is a block-diagonal matrix and symmetric, we can find the squared values of the eigenvalues of $\mathbf{W}$ (recall that $\mathbf{W}$ is symmetric) by computing the eigenvalues of $\mathbf{S}_{ij}\mathbf{S}_{ji}$ for all $i \neq j$. Suppose that these squared eigenvalues are represented in the set $\{v_i^2 : i = 1, 2, \ldots, m_1\}$. We have the following asymptotic expressions for the eigenvalues:

$$1 - \sigma_i\delta + \mathcal{O}(\delta^2) : i = 1, 2, \ldots, k. \tag{C.15}$$

$$s_j|v_j|\delta + \mathcal{O}(\delta^2) : j = 1, 2, \ldots, m_1, \tag{C.16}$$

where $s_j \in \{-1, 1\}$. Clearly, the eigenvalue with the largest absolute value approaches one from below as $\delta \to 0$. For sufficiently large λ (small δ), the spectral radius of the linear update matrix will be less than one, and convergence will occur with this level of regularisation.

Proof of Theorem 14

We have proven, for sufficiently large λ, convergence to

$$\mathbf{Q}_{ij} = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ji} \tag{C.17}$$

$$\mathbf{v}_{ij} = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}[\lambda\boldsymbol{\mu}_i + \mathbf{z}_i - \mathbf{v}_{ji}] \tag{C.18}$$

$$\boldsymbol{\mu}_i = \ddot{\mathbf{P}}_i[\lambda\boldsymbol{\mu}_i + \mathbf{z}_i], \tag{C.19}$$

where $\ddot{\mathbf{P}}_{ij}^{-1} = \lambda\mathbf{I}_{d_i} + \mathbf{S}_{ii} + \sum_{t\neq i,j}\mathbf{Q}_{ti}$, $\ddot{\mathbf{P}}_i^{-1} = \ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{Q}_{ji}$ and $\mathbf{z}_i = \boldsymbol{b}_i + \sum_{t\neq i}\mathbf{v}_{ti}$. Note that these formulas are valid even when $\mathbf{S}_{ij} = \mathbf{0}$. We now show that these equations imply that $\sum_i \mathbf{S}_{ji}\boldsymbol{\mu}_i = \boldsymbol{b}_j$ for all j.

Note the following for all $i \neq j$,

$$\begin{aligned}
\ddot{\mathbf{P}}_{ji} &= [\ddot{\mathbf{P}}_j^{-1} - \mathbf{Q}_{ij}]^{-1} \\
&= [\ddot{\mathbf{P}}_j^{-1} + \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ji}]^{-1} \\
&= \ddot{\mathbf{P}}_j - \ddot{\mathbf{P}}_j\mathbf{S}_{ji}[\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}]^{-1}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j,
\end{aligned} \tag{C.20}$$

and,

$$\begin{aligned}\ddot{\mathbf{P}}_i &= [\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{Q}_{ji}]^{-1} \\ &= [\ddot{\mathbf{P}}_{ij}^{-1} - \mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji}\mathbf{S}_{ji}]^{-1} \\ &= \ddot{\mathbf{P}}_{ij} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}[\ddot{\mathbf{P}}_{ji}^{-1} - \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij} \\ &= \ddot{\mathbf{P}}_{ij} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}[\ddot{\mathbf{P}}_{ji}^{-1} + \mathbf{Q}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij} \\ &= \ddot{\mathbf{P}}_{ij} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}. \end{aligned} \tag{C.21}$$

Consider the following lemma.

Lemma 7 *For all $i \neq j$ we have that $\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} = -\mathbf{Q}_{ij}\ddot{\mathbf{P}}_j$.*

Proof. Consider

$$\begin{aligned}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} &= \mathbf{S}_{ji}[\ddot{\mathbf{P}}_{ij} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}]\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} \text{ (see Equation (C.21))} \\ &= \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} + \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} \\ &= \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}[\ddot{\mathbf{P}}_j - \ddot{\mathbf{P}}_j\mathbf{S}_{ji}[\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}]^{-1}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j] \\ &+ \mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}[\ddot{\mathbf{P}}_j - \ddot{\mathbf{P}}_j\mathbf{S}_{ji}[\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}]^{-1}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j]. \end{aligned}$$

Setting $\mathbf{O}_{ij} = [\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}]^{-1}$, we obtain (after some simplification)

$$\begin{aligned}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji} &= -\mathbf{Q}_{ij}\ddot{\mathbf{P}}_j \\ &+ \mathbf{Q}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}[\mathbf{I}_{d_i} - \ddot{\mathbf{P}}_{ij}\mathbf{O}_{ij}^{-1} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}]\mathbf{O}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j. \end{aligned} \tag{C.22}$$

Consider

$$\begin{aligned}\mathbf{I}_{d_i} - \ddot{\mathbf{P}}_{ij}\mathbf{O}_{ij}^{-1} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji} &= \mathbf{I}_{d_i} - \ddot{\mathbf{P}}_{ij}[\ddot{\mathbf{P}}_{ij}^{-1} + \mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji}] + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji} \\ &= \mathbf{I}_{d_i} - \mathbf{I}_{d_i} - \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji} + \ddot{\mathbf{P}}_{ij}\mathbf{S}_{ij}\ddot{\mathbf{P}}_j\mathbf{S}_{ji} \\ &= \mathbf{0} : d_i \times d_i, \end{aligned}$$

which completes the proof.

Setting $\ddot{\mathbf{Q}}_{ij} = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{S}_{ij}$, we see that $\ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji} = \mathbf{Q}_{ij}\ddot{\mathbf{P}}_j$ and $\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij} = \mathbf{S}_{ji}\ddot{\mathbf{P}}_i + \ddot{\mathbf{Q}}_{ij}[\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i$ by Lemma 7 and Equation (C.20) respectively. Furthermore,

$$\begin{aligned}\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij} &= [\mathbf{I}_{d_j} + \ddot{\mathbf{Q}}_{ij}[\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]^{-1}]\mathbf{S}_{ji}\ddot{\mathbf{P}}_i \\ &= [\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij} + \ddot{\mathbf{Q}}_{ij}][\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i \\ &= \ddot{\mathbf{P}}_{ji}^{-1}[\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i. \end{aligned} \tag{C.23}$$

Consider

$$\begin{aligned}\mathbf{v}_{ij} &= -\mathbf{S}_{ji}\ddot{\mathbf{P}}_{ij}[\lambda\boldsymbol{\mu}_i + \mathbf{z}_i - \mathbf{v}_{ji}] \\ &= -\ddot{\mathbf{P}}_{ji}^{-1}[\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]^{-1}\mathbf{S}_{ji}\ddot{\mathbf{P}}_i[\lambda\boldsymbol{\mu}_i + \mathbf{z}_i - \mathbf{v}_{ji}], \end{aligned}$$

which implies that

$$-[\ddot{\mathbf{P}}_{ji}^{-1} - \ddot{\mathbf{Q}}_{ij}]\ddot{\mathbf{P}}_{ji}\mathbf{v}_{ij} = \mathbf{S}_{ji}\boldsymbol{\mu}_i - \mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{v}_{ji}, \tag{C.24}$$

since $\boldsymbol{\mu}_i = \ddot{\mathbf{P}}_i[\lambda\boldsymbol{\mu}_i + \mathbf{z}_i]$. From Equation (C.24) we see that

$$\mathbf{S}_{ji}\boldsymbol{\mu}_i = -\mathbf{v}_{ij} + \ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}\mathbf{v}_{ij} + \mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{v}_{ji}. \tag{C.25}$$

Since $\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{v}_{ji} = -\mathbf{S}_{ji}\ddot{\mathbf{P}}_i\mathbf{S}_{ij}\ddot{\mathbf{P}}_{ji}[\lambda\boldsymbol{\mu}_j + \mathbf{z}_j - \mathbf{v}_{ji}] = \ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}[\lambda\boldsymbol{\mu}_j + \mathbf{z}_j - \mathbf{v}_{ji}] = \ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}[\lambda\boldsymbol{\mu}_j+\mathbf{z}_j]-\ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}\mathbf{v}_{ij} = \mathbf{Q}_{ij}\ddot{\mathbf{P}}_j[\lambda\boldsymbol{\mu}_j+\mathbf{z}_j]-\ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}\mathbf{v}_{ij} = \mathbf{Q}_{ij}\boldsymbol{\mu}_j-\ddot{\mathbf{Q}}_{ij}\ddot{\mathbf{P}}_{ji}\mathbf{v}_{ij}$, we have

$$\mathbf{S}_{ji}\boldsymbol{\mu}_i = -\mathbf{v}_{ij} + \mathbf{Q}_{ij}\boldsymbol{\mu}_j \tag{C.26}$$

for all $i \neq j$. From Equation (C.26),

$$\begin{aligned}
\sum_i \mathbf{S}_{ji}\boldsymbol{\mu}_i &= \mathbf{S}_{jj}\boldsymbol{\mu}_j - \sum_{i\neq j}\mathbf{v}_{ij} + \left[\sum_{i\neq j}\mathbf{Q}_{ij}\right]\boldsymbol{\mu}_j \\
&= \mathbf{S}_{jj}\boldsymbol{\mu}_j + \boldsymbol{b}_j - \mathbf{z}_j + \left[\ddot{\mathbf{P}}_j^{-1} - \lambda\mathbf{I}_{d_j} - \mathbf{S}_{jj}\right]\boldsymbol{\mu}_j \\
&= \boldsymbol{b}_j - \mathbf{z}_j + \ddot{\mathbf{P}}_j^{-1}\boldsymbol{\mu}_j - \lambda\boldsymbol{\mu}_j \\
&= \boldsymbol{b}_j - \mathbf{z}_j + \lambda\boldsymbol{\mu}_j + \mathbf{z}_j - \lambda\boldsymbol{\mu}_j \\
&= \boldsymbol{b}_j
\end{aligned} \tag{C.27}$$

for all j. In particular, we see that the means implied by the stationary conditions satisfy $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$ and are therefore the correct marginal means.

Appendix D

Chapter 4 - Additional Notes

We start this Appendix by proving Lemma 8, which was used in Chapter 4 to derive the asymptotic behaviour of $\ddot{\mathbf{P}}_i(\lambda)$ and $\ddot{\mathbf{P}}_{ij}(\lambda)$.

Lemma 8 *Consider a sequence of matrices* $\mathbf{A}_n(\lambda)$ *with the following properties:*

1. *There exists a constant* λ_0 *such that* $lim_{n\to\infty}\mathbf{A}_n(\lambda) = \mathbf{A}(\lambda)$ *for all* $\lambda > \lambda_0$.
2. $||\mathbf{A}_n(\lambda)||_\infty \leq g(\lambda) = \mathcal{O}(\frac{1}{\lambda^3})$.

These properties imply that $\mathbf{A}(\lambda) = \mathcal{O}(\frac{1}{\lambda^2})$.

Proof.

Consider $A_{ij}^{(n)}(\lambda)$ and $\lim_{n\to\infty} A_{ij}^{(n)}(\lambda) = A_{ij}(\lambda)$ for any i, j. Since $||\mathbf{A}_n(\lambda)||_\infty \leq g(\lambda) = \mathcal{O}(\frac{1}{\lambda^3})$, we also have:

$$|A_{ij}^{(n)}(\lambda)| \leq g(\lambda) \tag{D.1}$$

Pre-multiplying (D.1) by λ^2 and taking the limit as $n \to \infty$ we see that:

$$|\lambda^2 A_{ij}(\lambda)| \leq \lambda^2 g(\lambda). \tag{D.2}$$

Since $\lim_{\lambda\to\infty} \lambda^2 g(\lambda) = 0$, we see from the squeeze theorem:

$$\lim_{\lambda\to\infty} \lambda^2 A_{ij}(\lambda) = 0. \tag{D.3}$$

(D.3) implies that $A_{ij}(\lambda) = \mathcal{O}(\frac{1}{\lambda^2})$.

The remainder of this appendix briefly describes UWTs for sGaBP. Consider applying sGaBP on the loopy MG given in the top panel of Figure D.1. There are four different UWTs for this graph, one for each of the clusters as the root node. The UWT for cluster 4 (with a depth of $n = 4$) is shown in the bottom

panel of Figure D.1.

The matrix $\mathbf{T}_{ii}^{(n)}(\lambda)$ can be obtained by moving along the UWT and assigning to a node, with reference to cluster t, the matrix $\lambda\mathbf{I}_{d_t} + \mathbf{S}_{tt}$. If two nodes in the UWT are linked, we need to determine the references of both nodes (say s and t), and they are linked by the matrix $\mathbf{S}_{ts}$. As an example,

$$\mathbf{T}_{44}^{(3)}(\lambda) = \begin{bmatrix} \mathbf{S}_{44}+\lambda\mathbf{I}_{d_4} & \mathbf{S}_{42} & \mathbf{S}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{24} & \mathbf{S}_{22}+\lambda\mathbf{I}_{d_2} & \mathbf{0} & \mathbf{S}_{21} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{34} & \mathbf{0} & \mathbf{S}_{33}+\lambda\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{31} & \mathbf{S}_{32} \\ \mathbf{0} & \mathbf{S}_{12} & \mathbf{0} & \mathbf{S}_{11}+\lambda\mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{33}+\lambda\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{11}+\lambda\mathbf{I}_{d_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{22}+\lambda\mathbf{I}_{d_2} \end{bmatrix}.$$

The matrix $\mathbf{T}_{ji}^{(n)}(\lambda)$, $j \neq i$ can be obtained in a similar fashion. However, we only consider the subtree rooted at the node in the second layer corresponding to cluster j of the original graph. For example,

$$\mathbf{T}_{34}^{(3)}(\lambda) = \begin{bmatrix} \mathbf{S}_{33}+\lambda\mathbf{I}_{d_3} & \mathbf{S}_{31} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{13} & \mathbf{S}_{11}+\lambda\mathbf{I}_{d_1} & \mathbf{0} & \mathbf{S}_{12} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{23} & \mathbf{0} & \mathbf{S}_{22}+\lambda\mathbf{I}_{d_2} & \mathbf{0} & \mathbf{S}_{21} & \mathbf{S}_{24} \\ \mathbf{0} & \mathbf{S}_{21} & \mathbf{0} & \mathbf{S}_{22}+\lambda\mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{12} & \mathbf{0} & \mathbf{S}_{11}+\lambda\mathbf{I}_{d_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{44}+\lambda\mathbf{I}_{d_4} \end{bmatrix}.$$

Let us consider constructing a potential vector $\mathbf{t}_4^{(3)}(\lambda)$ to use alongside $\mathbf{T}_{44}^{(3)}(\lambda)$ for the computation of $\boldsymbol{\mu}_4^{(2)}(\lambda)$. Suppose we have already obtained $\boldsymbol{\mu}^{(0)}(\lambda)$ and $\boldsymbol{\mu}^{(1)}(\lambda)$. The potential is:

$$\mathbf{t}_4^{(3)}(\lambda) = \begin{bmatrix} \boldsymbol{b}_4 + \lambda\boldsymbol{\mu}_4^{(1)}(\lambda) \\ \boldsymbol{b}_2 + \lambda\boldsymbol{\mu}_2^{(0)}(\lambda) \\ \boldsymbol{b}_3 + \lambda\boldsymbol{\mu}_3^{(0)}(\lambda) \\ \boldsymbol{b}_1 + \lambda\boldsymbol{b}_1 \\ \boldsymbol{b}_3 + \lambda\boldsymbol{b}_3 \\ \boldsymbol{b}_1 + \lambda\boldsymbol{b}_1 \\ \boldsymbol{b}_2 + \lambda\boldsymbol{b}_2 \end{bmatrix}.$$

Consider the following matrices:

$$[\mathbf{G}_{44}^{(3)}]' = \begin{bmatrix} \mathbf{I}_{d_4} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{E}_{44}^{(3)} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_4} \\ \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} & \mathbf{0} \\ \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} & \mathbf{0} \\ \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{J}_{44}^{(3)} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_4} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

From these definitions we see that $\mathbf{t}_4^{(3)}(\lambda) = \mathbf{E}_{44}^{(3)}\boldsymbol{b} + \lambda\mathbf{J}_{44}^{(3)}\boldsymbol{\phi}_2(\lambda)$ where $\boldsymbol{\phi}_2(\lambda) = (\boldsymbol{\mu}^{(1)}(\lambda), \boldsymbol{\mu}^{(0)}(\lambda), \boldsymbol{b})'$. We can obtain $\boldsymbol{\mu}_4^{(2)}(\lambda)$ by computing $[\mathbf{G}_{44}^{(3)}]'[\mathbf{T}_{44}^{(3)}(\lambda)]^{-1}\mathbf{t}_4^{(3)}(\lambda)$, which is what is given by Equation (4.42).

Let us consider an example for the change of Equation (4.42) to Equation (4.49) for the heuristic measure. For our example we have $n = 3$ and $i = 4$. The following changes are made:

$$\mathbf{T}_{44}^{(3)}(\lambda^{(2)}) = \begin{bmatrix} \mathbf{S}_{44} + \lambda^{(2)}\mathbf{I}_{d_4} & \mathbf{S}_{42} & \mathbf{S}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{24} & \mathbf{S}_{22} + \lambda^{(1)}\mathbf{I}_{d_2} & \mathbf{0} & \mathbf{S}_{21} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{34} & \mathbf{0} & \mathbf{S}_{33} + \lambda^{(1)}\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{31} & \mathbf{S}_{32} \\ \mathbf{0} & \mathbf{S}_{12} & \mathbf{0} & \mathbf{S}_{11} + \lambda^{(0)}\mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{33} + \lambda^{(0)}\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{11} + \lambda^{(0)}\mathbf{I}_{d_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_{23} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{22} + \lambda^{(0)}\mathbf{I}_{d_2} \end{bmatrix}.$$

$$\mathcal{D}_{34}(\lambda^{(2)}) = \begin{bmatrix} \lambda^{(2)}\mathbf{I}_{d_4} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \lambda^{(1)}\mathbf{I}_{d_2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \lambda^{(1)}\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda^{(0)}\mathbf{I}_{d_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda^{(0)}\mathbf{I}_{d_3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda^{(0)}\mathbf{I}_{d_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \lambda^{(0)}\mathbf{I}_{d_2} \end{bmatrix}.$$

Here, $\lambda^{(2)}$, $\lambda^{(1)}$ and $\lambda^{(0)}$ denote the regularisation used for layers 1, 2 and 3 of the UWT in Figure D.1. For the purpose of our heuristic, we are interested in tuning $\lambda^{(2)}$ to a level where the posterior means at iteration 2 are closer to solving the system of linear equations, $\mathbf{S}\boldsymbol{\mu} = \boldsymbol{b}$.

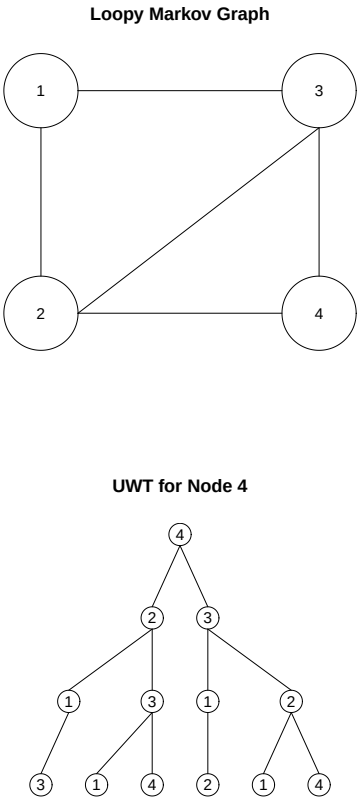


Figure D.1: Loopy Markov graph and the UWT for cluster 4 with a depth of $n = 4$.

List of References

Aji, S. and McEliece, R. (2000 Mar). The generalized distributive law. *IEEE Transactions on Information Theory*, vol. 46, pp. 325–343.

Alameda-Pineda, X., Staiano, J., Subramanian, R., Batrinca, L., Ricci, E., Lepri, B., Lanz, O. and Sebe, N. (2015 June). Salsa: A novel dataset for multimodal group behavior analysis. `1506.06882`.

Bach, F., Jenatton, R., Mairal, J. and Obozinski, G. (2011). Convex optimization with sparsity-inducing norms. In: Sra, S., Nowozin, S. and Wright, J. (eds.), *Optimization for Machine Learning*. MIT Press.

Bickson, D. (2008 October). *Gaussian Belief Propagation: Theory and Application*. Ph.D. thesis, The Hebrew University of Jerusalem.

Clifford, P. (1990). Markov random fields in statistics. In: Grimmett, G.R. and Welsh, D.J.A. (eds.), *Disorder in Physical Systems. A Volume in Honour of John M. Hammersley*, pp. 19–32. Clarendon Press, Oxford.

Efron, B., Hastie, T., Johnstone, I. and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499.

El-Kurdi, Y., Giannacopoulos, D. and Gross, W.J. (2012 September*a*). Relaxed Gaussian belief propagation. In: *Proc. IEEE International Symposium on Information Theory*.

El-Kurdi, Y., Gross, W.J. and Giannacopoulos, D. (2012 February*b*). Efficient implementation of Gaussian belief propagation solver for large sparse diagonally dominant linear systems. *IEEE Transactions on Magnetics*, vol. 48, pp. 471–474.

Frey, B. and Kschischang, F. (1996 Oct). Probability propagation and iterative decoding. In: *Proc. 34th Annual Allerton Conference on Communication, Control, and Computing*. Allerton House, Monticello, Illinois.

Gallager, R.G. (1963). *Low-Density Parity-Check Codes*. MA: MIT Press, Cambridge.

Hammersley, J.M. and Clifford, P. (1971). Markov field on finite graphs and lattices. Unpublished.

Johnson, J.K., Bickson, D. and Dolev, D. (2009). Fixing convergence of Gaussian belief propagation. In: *Proc. IEEE International Symposium on Information Theory*. Seoul, South Korea.

Kamper, F. (2017 October). An Empirical Study of Gaussian Belief Propagation and Application in the Detection of F-formations. In: *Proc. ACM Multimedia 2017 Workshop on South African Academic Participation*.

Kamper, F., du Preez, J., Steel, S. and Wagner, S. (2018 May*a*). Regularized Gaussian belief propagation. *Statistics and Computing*, vol. 28, no. 3, pp. 653–672.

Kamper, F., Steel, S. and du Preez, J. (2018*b*). On the convergence of Gaussian belief propagation with nodes of arbitrary size. Working paper, Stellenbosch University.

Kamper, F., Steel, S. and du Preez, J. (2018*c*). Regularized Gaussian belief propagation with nodes of arbitrary size. Working paper, Stellenbosch University.

Kendon, A. (1990). *Conducting Interaction: Patterns of Behavior in Focused Encounters*. Cambridge University Press, Cambridge.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models Principles and Techniques*. The MIT Press, Cambridge, MA.

Lauritzen, S. and Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B*, pp. 157–224.

Liu, Y. (2010 June). *Feedback Message Passing for Inference in Gaussian Graphical Models*. Master's thesis, Massachusetts Institute of Technology.

Malioutov, D.M., Johnson, J.K. and Willsky, A.S. (2006 October). Walk-Sums and Belief Propagation in Gaussian Graphical Models. *Journal of Machine Learning Research*, vol. 7, pp. 2031–2064.

Minka, T. (2001). Expectation propagation for approximate Bayesian inference. In: *Proc. 17th Conference on Uncertainty in Artificial Intelligence*, pp. 362–369. Morgan Kaufmann Publishers Inc.

Murphy, K.P., Weiss, Y. and Jordan, M.I. (1999). Loopy belief propagation for approximate inference: An empirical study. In: *Proc. 15th Conference on Uncertainty in Artificial Intelligence*, pp. 467–475. Morgan Kaufmann Publishers Inc.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Francisco, CA, USA.

Ruozzi, N. and Tatikonda, S. (2013 August). Message-passing algorithms for quadratic minimization. *Journal of Machine Learning Research*, vol. 14, pp. 2287–2314.

Shachter, R. (1988). Probabilistic inference and influence diagrams. *Operations Research*, vol. 36, pp. 589–605.

Shafer, G. and Shenoy, P. (1990). Probability propagation. *Annals and Mathematics and Artificial Intelligence*, vol. 2, pp. 327–351.

Shental, O., Siegel, P.H., Wolf, J.K., Bickson, D. and Dolev, D. (2008). Gaussian belief propagation solver for systems of linear equations. In: *Proc. IEEE International Symposium on Information Theory*, pp. 1863–1867.

Shewchuk, J.R. (1994 August 4). *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Su, Q. and Wu, Y. (2015 March). On convergence conditions of Gaussian belief propagation. *IEEE International Transactions on Signal Processing*, vol. 63, pp. 1144–1155.

Sui, T., Marelli, D. and Fu, M. (2015 Dec). Convergence analysis of Gaussian belief propagation for distributed state estimation. In: *Proc. IEEE Annual Conference on Decision and Control*. Osaka, Japan.

Weiss, Y. and Freeman, W.T. (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, vol. 13, no. 10, pp. 2173–2200.