Delft University of Technology

Thesis Project
WI5000COSSE

Backward Filtering Forward Guiding for Finite-State Space Models with Expectation Propagation

Author:
Daniël Brus (4379179)

Supervisors TU Delft:
Prof.dr.ir. F.H. van der Meulen
Prof.dr.ir. C. Vuik
Dr. J. Söhl

June 25, 2023

Abstract

In many fields we are interested in inference for a complex stochastic process given limited observations regarding its state over time. This thesis therefore introduces an expectation propagation approach to backward filtering forward guiding for high-dimensional finite-state space models. The backward filtering forward guiding method is first derived for such models with a specific emphasis on the temporal dynamics, after which factorised guiding terms which exploit the inherent structure of the latent state space are introduced.

Performance of the method is assessed by comparing numerical results for statistical inference of a Susceptible-Infected-Recovered example problem. The expectation propagation approach performs comparably with existing methods in a particularly simple setting where state variables are observed individually, and performs very well in a more difficult setting which the familiar methods can not deal with. We conclude that a more advanced treatment of the approximate likelihood filtering phase may be warranted in such complex settings.

The research summarised in this work provides a first effort towards the development of more general expectation propagation based backward filtering procedures for other types of high-dimensional sequential data models. The work additionally elucidates connections between backward filtering with backward marginalisation and alternative approximate likelihood filtering procedures, suggesting multiple avenues for future research.

Keywords: parameter inference, backward filtering forward guiding, expectation propagation, M-projection, state space model, dynamic Bayesian Network, guided process, guiding functions

Contents

1 Introduction

In a large number of scientific fields and engineering disciplines we are interested in the latent dynamics of a complex stochastic process given limited observations regarding its state over time. The development of high-dimensional sequential data modelling has achieved remarkable success in particular, with numerous applications in fields such as natural language processing, computer vision, climate forecasting and financial econometrics. Inference for latent states and model parameters in this setting is of often challenging. Van der Meulen & Schauer (2022) propose a general paradigm for this task, which they refer to as *backward filtering forward guiding*.

The method consists of two distinct phases. In the *backward filtering* phase information provided by the observations is first propagated through a model. The model is thereby transformed into a *guided model* featuring a distribution which we can tractably sample from and which approximates the exact conditional distribution, with known likelihood ratio between the two. In the *forward guiding* phase the conditional distribution can then be targeted through weighted sampling, allowing for a variety of inferential methods.

For a discrete-time finite-state space model the backward filtering phase requires us to compute for each time step a guiding term which approximates the conditional likelihood of 'later' observations. Such guiding terms must necessarily include an entry for each possible state of the process, and because the number of possible states grows exponentially with respect to the number of state variables such terms may prove to be intractably large even in only moderately complex settings. Van der Meulen & Schauer describe one approach to dealing with this issue, and following a literature review we have noted similarities between the proposed approach and other approximate likelihood filtering procedures.

In this thesis I propose the use of guiding terms which exploit the natural factorisation of the state space via the set of conditionally independent state variables as an alternative solution to this problem. With this approach we restrict guiding terms to be members of some tractable family of functions, and for each time step we compute the 'optimal' guiding term through a recursive procedure by minimising a KL-divergence. The procedure therefore consists of a series or recurse- and project-steps and may be seen as an *expectation propagation* approach to backward filtering.

We compare the expectation propagation approach to the backward filtering procedures originally described by Van der Meulen & Schauer (2020) through an example problem. In particular, we are interested in parameter inference and state estimation for a stochastic Susceptible-Infected-Recovered (SIR) process of a collection of individuals featuring the following dynamics: susceptible individuals may be infected either spontaneously or by their infected neighbours, infected individuals randomly recover (at which point they are resistant to reinfection), and recovered individuals randomly return to the susceptible state. We seek to infer the rates by which individuals transition between states and reconstruct the realisation of the process which produced the observations.

Descriptions of the general problem we are interested in and the SIR-process specifically may be found in section 2. In section 3 the backward filtering forward guiding method itself is described for the finite-state space models considered in this work specifically. The expectation propagation approach to backward filtering is described in section 4, and numerical results are presented in section 5. A description of the computational data structure used to represent the involved objects is provided in Appendix A.

2 Problem description

In this section we first describe the structure of the finite-state space models considered in this work. We then formulate the problem of inferring latent states and model parameters for such a model given observations regarding its state over time and finally we present an example problem which corresponds to this setting.

2.1 The generative model

We wish to model the evolution of a finite-state stochastic process $\mathcal{X}$ over discrete time steps $\{1, \dots, T\}$, which we refer to as the *forward model*. Denote the state of the process $\mathcal{X}$ at time step t with $\boldsymbol{X}^{(t)}$ and denote states between two subsequent time steps collectively with

$$\boldsymbol{X}^{(t:t')} = \{\boldsymbol{X}^{(t)}, \dots, \boldsymbol{X}^{(t')}\}, \quad (t \leq t'). \tag{1}$$

By the basic rules of probability the joint distribution of $\mathcal{X} \equiv \boldsymbol{X}^{(1:T)}$ factorises as

$$P\left(\boldsymbol{X}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1)}\right) \prod_{t=2}^{T} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(1:t-1)}\right). \tag{2}$$

In the preceding display the conditional probability distribution for each time step depends on all of the previous states the process has attained. This 'remembering' property results in unwieldy inference procedures in general and therefore we will assume that the process is *memoryless*.

Assumption 2.1. Throughout this work we assume that $\mathcal{X}$ satisfies the **Markov property**:

$$\left(\boldsymbol{X}^{(t+1)} \perp \boldsymbol{X}^{(1:t-1)} \mid \boldsymbol{X}^{(t)}\right). \tag{3}$$

Given this assumption each time step evolution depends on only the 'present' state of the process, and it follows that now

$$P\left(\boldsymbol{X}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1)}\right) \prod_{t=2}^{T} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right). \tag{4}$$

The joint distribution is now fully defined by a distribution describing the state of the first time step - which we refer to as the *initial state* - and a description of the incremental evolution of the process for all other time steps. To further simplify the process we will additionally assume that its evolution is *time-invariant*.

Assumption 2.2. Throughout this work we assume that $\mathcal{X}$ is **time-invariant**, i.e. for all t:

$$P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{X}' \mid \boldsymbol{X}\right). \tag{5}$$

The term on the right hand side of the preceding display fully encodes the process dynamics of $\mathcal{X}$ and we refer to is as the *transition model*. The transition model describes the probability of transitioning to any possible assignment $\boldsymbol{X}' = \boldsymbol{x}'$ in the following time step given the current state.

For models featuring a large state space it may prove difficult (or even intractable) to adequately specify the transition model as a singular entity, for computational reasons or other. We therefore assume that the process consists of a number of interacting *sub-processes* which we can instead model separately, resulting in a compact representation of the overall process dynamics.

Assumption 2.3. Throughout this work we assume that the state space of $\mathcal{X}$ can be decomposed into disjoint subspaces corresponding to interacting sub-processes which transition independently of one another given the current state. We identify the states of the conditionally independent sub-processes with **state variables** $\{X_1^{(t)}, \ldots, X_N^{(t)}\}$, such that for all pairs (i, j):

$$\left(X_i^{(t+1)} \perp X_j^{(t+1)} \mid \boldsymbol{X}^{(t)}\right). \tag{6}$$

We refer to the set of sub-processes that the transition of any given sub-process $\mathcal{X}_i \equiv X_i^{(1:T)}$ depends on (including itself) as its *neighbourhood* and denote the indices corresponding to this set collectively with $\mathrm{ne}(i)$.

It follows that the transition model is now given by product of factors for individual state variables:

$$P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} P\left(X_i^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t)}\right). \tag{7}$$

A sub-process $\mathcal{X}_i$ need not interact with all other sub-processes and for computational reasons it would be beneficial if its neighbourhood were small: If it is difficult to specify a transition probability between all possible pairs of states $(\boldsymbol{x}, \boldsymbol{x}')$ directly, then having to specify a transition probability between pairs $(\boldsymbol{x}, x_i')$ for all N sub-processes instead would be only marginally less so.

In this thesis I propose an extension of the backward filtering forward guiding method to state space models which exploits small neighbourhood sets. We will therefore require that all sub-processes of $\mathcal{X}$ interact with relatively few other sub-processes, a phenomena we refer to as *local interaction*. Further motivation for this requirement is provided in section 4.

Assumption 2.4. Throughout this work we assume that the transition of any given sub-process $\mathcal{X}_i$ depends on only a small subset of current state $\boldsymbol{X}_{\mathrm{ne}(i)}^{(t)} \subset \boldsymbol{X}^{(t)}$, such that for all i:

$$\mid \mathrm{ne}(i) \mid << N. \tag{8}$$

Finally, we will extend the assumption of conditional independence in later time steps to the first by assuming independence between the states of sub-processes in the initial state.

Assumption 2.5. Throughout this work we assume that the initial states of sub-processes of $\mathcal{X}$ are independent of one another, such that:

$$P\left(\boldsymbol{X}^{(1)}\right) = \prod_{i=1}^{N} P\left(X_i^{(1)}\right). \tag{9}$$

Having fully defined a process $\mathcal{X}$ in terms of its initial state and transition model we now consider the observations which we require for the task of inferring latent states and model parameters. We introduce an *observation model* and will assume that in each time step an observation depends on only the current state $\boldsymbol{X}^{(t)}$. Denote the state of the observation model at time step t with $\boldsymbol{Y}^{(t)}$.

Assumption 2.6. Throughout this work we assume that for all time steps t an observation is conditionally independent of all other observations given the latent state, such that:

$$\left(\boldsymbol{Y}^{(t)} \perp \boldsymbol{Y}^{(1:t-1)}, \boldsymbol{Y}^{(t+1:T)} \mid \boldsymbol{X}^{(t)}\right). \tag{10}$$

Similar to the transition model we require that the observations in a given time step are conditionally independent of one another, though the observation model is not assumed to be time time-invariant, allowing for incomplete observations and other variations.

Assumption 2.7. Throughout this work we assume that an observation model consists of a number of conditionally independent **observation variables** $\{Y_1^{(t)}, \ldots, Y_M^{(t)}\}$ which each measure some subset of the latent state $\boldsymbol{X}_{\text{obs}(i)}^{(t)} \subset \boldsymbol{X}^{(t)}$, such that:

$$P\left(\boldsymbol{Y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{M} P\left(Y_i^{(t)} \mid \boldsymbol{X}_{\text{obs}(i)}^{(t)}\right). \tag{11}$$

Note that not all state variables need necessarily be observed, i.e. $\prod_{i=1}^{M} \boldsymbol{X}_{\text{obs}(i)}^{(t)} \subseteq \boldsymbol{X}^{(t)}$.

Finally, we require that each conditionally independent transition model factor may only measure a set of variables which is a subset of the neighbourhood of at least one sub-process.

Assumption 2.8. Throughout this work we assume that for any set of observed state variables $\boldsymbol{X}_{\text{obs}(i)}^{(t)}$ there exists at least one sub-process $\mathcal{X}_j$ featuring a neighbourhood which encompasses that set, such that

$$\boldsymbol{X}_{\text{obs}(i)}^{(t)} \subseteq \boldsymbol{X}_{\text{ne}(j)}^{(t)}. \tag{12}$$

Motivation for this final requirement is again provided in section 4.

From the preceding assumptions it follows that the joint distribution of the entire process and its observations factorises as

$$P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{Y}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1)}\right) P\left(\boldsymbol{Y}^{(1)} \mid \boldsymbol{X}^{(1)}\right) \prod_{t=2}^{T} \left(P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) P\left(\boldsymbol{Y}^{(t)} \mid \boldsymbol{X}^{(t)}\right)\right), \tag{13}$$

where the transition model and observation model terms in turn factorise as shown in equation (7) and equation (11). The distribution in the preceding display equates to a model for all latent states and observations jointly, and is often referred to as a *generative model* in related literature. A general graphical representation of the generative models described in this subsection is shown in Figure 1.

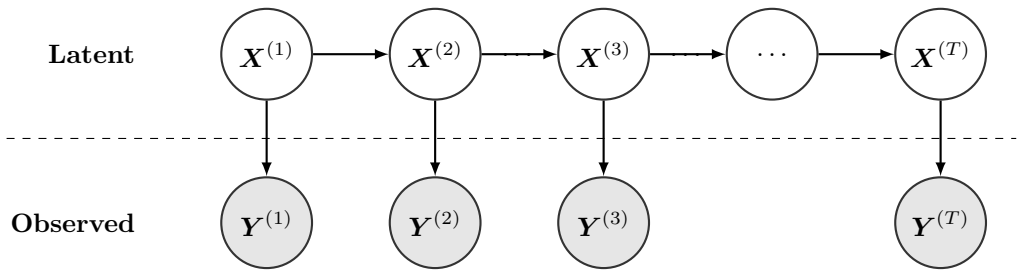

Figure 1: A representation of the generative models encompassed by the form shown in equation (13). Circles indicate states of either the forward model or observation model and arrows indicate the conditional dependence between states.

2.2 Statistical inference

In the previous subsection we developed a generative model for $\mathcal{X}$ and its observations consisting of the triplet

$$\left(P\left(\boldsymbol{X}^{(1)}\right),\ P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right),\ P\left(\boldsymbol{Y}^{(t)} \mid \boldsymbol{X}^{(t)}\right)\right). \tag{14}$$

In practice we often do not know the true values of certain *model parameters* which encode the objects featuring in equation (14): Instead we have available only a parametric representation of the objects and a limited set of observations $\boldsymbol{y}^{(1:T)}$, and we would like to estimate the parameters which best explain the observations. Denote the unknown model parameters collectively with $\boldsymbol{\theta}$.

For ease of exposition we will assume that only the initial state or transition model may contain unknown model parameters, and that the observation model does not.

Assumption 2.9. Throughout this work we assume that only the forward model contains unknown model parameters, such that:

$$P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{Y}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}\right) = P\left(\boldsymbol{Y}^{(1:T)} \mid \boldsymbol{X}^{(1:T)}\right) P\left(\boldsymbol{X}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}\right). \tag{15}$$

Model parameters for the initial state generally encode the probability for each state $x_i^{(1)}$ explicitly, whereas transition model parameters could encode the transition probability between pairs $(\boldsymbol{x}_{\mathrm{ne}(i)}, x_i')$ in a more compact way. For example, if sub-processes interact with one another only when in certain states then many transition pairs may be encoded by the same few parameters.

We may additionally be interested in the posterior distribution of states of $\mathcal{X}$ given all its observations, and because $\boldsymbol{\theta}$ is unknown we choose to adhere to a fully Bayesian setting, i.e. we model $\boldsymbol{\theta}$ itself as a random quantity and concern ourselves with the joint posterior distribution:

$$P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right). \tag{16}$$

A graphical representation of the resulting statistical model is shown in Figure 2.

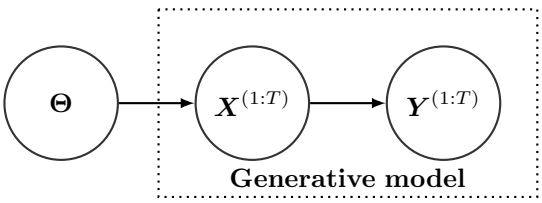

Figure 2: A representation of the statistical model shown in equation (16).

Sampling from this distribution is a highly challenging inference problem because the latent states and model parameters can be highly correlated. The posterior distribution does not admit a closed form representation and appropriate inference methods are required to tractably interact with it.

In the following section we introduce one such method, referred to as *backward filtering forward guiding*. We consider the method from the perspective of entire time steps in section 3 and derive an approach which exploits the conditionally independent state variables specifically in section 4.

2.3 Example problem

In this subsection we discuss an example problem originally presented by Van der Meulen & Schauer (2020). Consider a population of N individuals in which at any time each individual is either **S**usceptible, **I**nfected or **R**ecovered. Suppose that an individual transitions from **S** to **I** at a rate proportional to its number of infected neighbours and that it transitions from **I** to **R** and from **R** back to **S** at differing constant rates.

We first model each individual as one of N interacting continuous-time Markov chains. Denote the state of individual i at time step t with $x_i^{(t)}$ and denote the collection of states of individuals jointly with $\boldsymbol{x}^{(t)}$. In summary we have that:

- if $x_i^{(t)} = \mathbf{S}$ the individual transitions to **I** with intensity $\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)$,
- if $x_i^{(t)} = \mathbf{I}$ the individual transitions to **R** with intensity μ, and
- if $x_i^{(t)} = \mathbf{R}$ the individual transitions to **S** with intensity ν,

where

$$N_i\left(\boldsymbol{x}^{(t)}\right) = \{\text{number of infected neighbours of individual } i \text{ in state } \boldsymbol{x}^{(t)}\}.$$

Denote the categorical state space of each individual with $E = \{e_1 = \mathbf{S}, e_2 = \mathbf{I}, e_3 = \mathbf{R}\}$, such that the transition rate matrix of individual i is given by

$$Q_i\left(\boldsymbol{x}^{(t)}\right) = \begin{bmatrix} -\left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & \left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & 0 \\ 0 & -\mu & \mu \\ \nu & 0 & -\nu \end{bmatrix}. \tag{17}$$

A transition rate matrix Q represents the intensities with which a continuous-time Markov chain transitions between states, such that the matrix element $q_{i,j}$ describes the transition rate from e_i to e_j and $q_{i,i}$ describes the total transition rate 'away' from e_i. Note that the transition rate matrix itself is not of immediate interest but rather that it is used to describe the discrete-time forward model considered next.

2.3.1 Time discretisation

We wish to consider the stochastic process as a state space model and therefore now introduce a time discretisation: Time steps are multiples of $\tau > 0$ and in each time interval $[t, t+\tau)$, conditional on the present state $\boldsymbol{x}^{(t)}$, each individual independently either transitions or remains in its present state.[1] We assume that in a time interval an individual can transition only once, and may therefore approximate the transition probabilities as exponentially distributed waiting times.

In a given interval the transition matrix of individual i is therefore given by

$$K_i\left(\boldsymbol{x}^{(t)}\right) = \begin{bmatrix} \psi\left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & 1 - \psi\left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & 0 \\ 0 & \psi(\mu) & 1 - \psi(\mu) \\ 1 - \psi(\nu) & 0 & \psi(\nu) \end{bmatrix}, \tag{18}$$

where

$$\psi(u) = \exp\left(-\tau u\right).$$

A transition matrix K represents the probabilities with which a discrete-time Markov chain transitions between states, such that the matrix element $k_{i,j}$ describes the probability of transitioning from e_i to e_j in a time interval and $k_{i,i}$ describes the probability of 'remaining' in the current state.

[1] Such an approximation should be seen as a forward Euler approach to discretisation of the collection of continuous-time Markov processes described previously, as in actuality the number of infected neighbours (and thus the rate of transitioning to the infected state) need not remain constant throughout an interval.

2.3.2 The generative model

Let $\mathcal{X}_i$ denote the sub-process corresponding to the states of individual i, such that $\mathcal{X}$ denotes the process corresponding to the population as a whole. Each sub-process $\mathcal{X}_i$ interacts with the two neighbours to its immediate 'left' and 'right' $\mathcal{X}_{i-2}$, $\mathcal{X}_{i-1}$, $\mathcal{X}_{i+1}$, $\mathcal{X}_{i+2}$, with suitable restrictions for the 'edge' sub-processes $\mathcal{X}_1$, $\mathcal{X}_2$, $\mathcal{X}_{N-1}$, $\mathcal{X}_N$.

Recall that the forward model is fully described in terms of an initial state and a transition model. We may assume some fixed initial state for each sub-process and identify the conditionally independent factors of the transition model with the transition matrices K_i, as these fully described the transition probabilities for a given individual. Two realisations of such a SIR process consisting of 50 individuals are displayed in Figure 3. In this example the individuals $\{1, \ldots, 7\}$ are initialised as **I**nfected and the remaining individuals as **S**usceptible.

Note that although realisations of the process are random, some semblance of structure is preserved: Because infection is primarily transmitted between individuals in close proximity to one another the spread of infection throughout the population exhibits wavelike behaviour. Independently of their neighbours, infected individuals recover and become susceptible again, and when enough individuals have returned to the susceptible state another wave of infection may propagate through the population.

Figure 4 shows an example in which the two realisations displayed in Figure 3 are observed through distinct observation schemes: For both realisations the entire population is observed every 50 time steps, though in the left figure the states of individuals are observed directly whereas in the right figure individuals are observed to be either **S** or {**I** or **R**}.

2.3.3 Statistical inference

One may reasonably expect that because of the apparent structure featuring in realisations of the SIR process that it should be possible to infer the transition rates of the process dynamics if the process is observed at 'enough' points. This forms the problem we concern ourselves with in section 5: statistical inference of the model parameters $\boldsymbol{\theta} = (\lambda, \mu, \nu)$, given limited observations regarding the state of $\mathcal{X}$ over time.

We may additionally be interested in reconstructing the realisation of $\mathcal{X}$ which produced the observations. Figure 5 shows two alternative realisations of the SIR process described previously which agree with the observed states displayed in Figure 4. Note that in the right figure the states of individuals need not agree with those shown in Figure 3 exactly, as **I**nfected and **R**ecovered individuals are mapped to the same observation state.

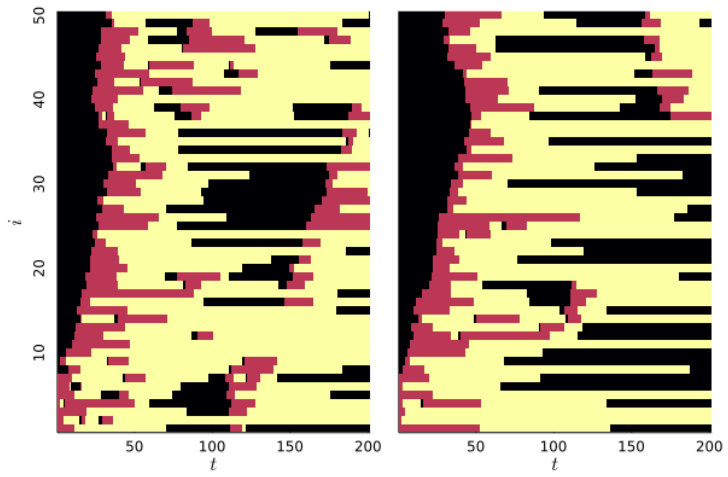

Figure 3: Two realisations of a discrete-time SIR process of 50 individuals and 200 time steps. **S**usceptible, **I**nfected and **R**ecovered individuals are respectively denoted as black, red and yellow. Note the spontaneous emergence of infection at certain points because of the 'base' intensity λ_0.

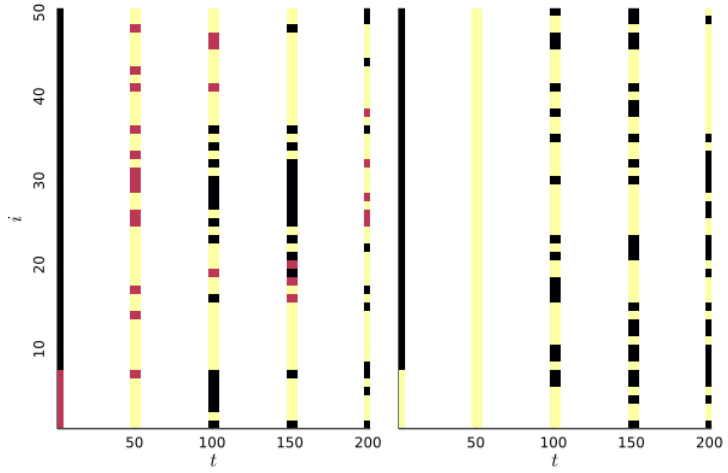

Figure 4: The realisations displayed in Figure 3 observed every 50 time steps through distinct observation models. In the right figure **S**usceptible and {**I**nfected or **R**ecovered} individuals are respectively denoted as black and yellow. The width of observed states is increased for clarity.

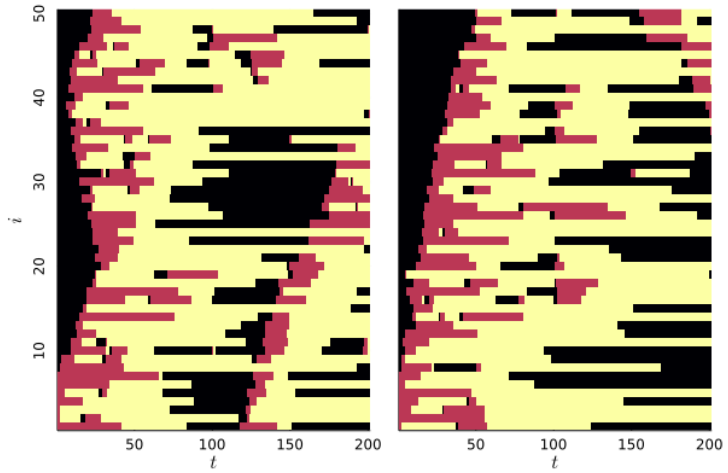

Figure 5: Two alternative realisations of a discrete-time SIR process of 50 individuals and 200 time steps which agree with the observed states displayed in Figure 4. Note that in the right figure the states of individuals need not agree with those shown in Figure 3 exactly, as **I**nfected and **R**ecovered individuals are mapped to the same observation state.

3 The backward filtering forward guiding method

In this section we introduce the backward filtering forward guiding method for finite-state space models specifically. We take a high-level approach and for a more measure-theoretic exposition we refer the reader to the original work provided by Van der Meulen & Schauer (2022). Van der Meulen (2022) additionally provides an introduction to the method which is more accessible and encompasses the setting considered in this work specifically.

3.1 Approach

Recall that the distribution of the generative model is encoded by a set of model parameters through its initial state and transition model. Given a suitable prior the joint distribution of the model parameters and the generative model therefore factorises as

$$P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{Y}^{(1:T)}, \boldsymbol{\Theta}\right) = P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{Y}^{(1:T)} \mid \boldsymbol{\Theta}\right) P\Big(\boldsymbol{\Theta}\Big), \tag{19}$$

where we explicitly denote the dependence of the distribution of latent states of $\mathcal{X}$ on the model parameters $\boldsymbol{\theta}$ as a conditional distribution. We are interested in the posterior distribution of latent states and model parameters given an assignment for the observations, as shown in equation (16).

In its simplest form the backward filtering forward guiding method decomposes this task into two parts: inference for model parameters and inference for latent states. It follows from equation (19) that the posterior distribution of model parameters and latent states jointly factorises as

$$P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1:T)} \mid \boldsymbol{\Theta}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) P\left(\boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right). \tag{20}$$

The second term on the right hand side of the preceding display represents the posterior distribution of model parameters given all available observations, whereas the first term represents the conditional distribution of latent states given the observations, further conditioned on the model parameters. By analysing equation (20) we see that we can target the joint distribution of interest by first obtaining a sample $\boldsymbol{\theta}$ from the posterior distribution of the model parameters given the observations, and subsequently sampling from the conditional distribution of $\mathcal{X}$ which is instantiated by that value $\boldsymbol{\theta}$ and conditioned on the observations $\boldsymbol{y}^{(1:T)}$.

Van der Meulen & Schauer (2022) propose a method for statistical inference based on this insight, focusing their efforts on the construction of the first term for a variety of Markov processes and graphical models. In its simplest form the backward filtering phase of the method consists of propagating information provided by the observation backwards in the form of conditional likelihood terms, after which the filtered information is used in the forward guiding phase to construct a secondary process which is conditioned on the observations.

Only in certain situations can the information from the observations be propagated backwards exactly (i.e. in closed-form), in which case the resulting constructed process allows us to sample from the forward model conditioned on the observations directly. The key insight is that performing the forward guiding phase with approximate *guiding terms* results in a process which is *guided* by the observations, featuring a distribution which approximates that of the exact conditioned model and with known likelihood ratio between the two. In that case we do not simply target the distribution of interest as shown in equation (20) but instead through *weighted sampling*, a result we derive at a later point. The finite-state setting considered in this work allows for exact closed-form filtering in theory, but in practice this proves intractable for processes featuring a large state space for computational reasons. The contribution provided by this work addresses this issue specifically.

In the following subsection we first consider the exact filtering and guiding procedures before introducing the more general framework in the subsection after it. We describe the method in isolation assuming that we have available a correctly distributed sample of the model parameters $\boldsymbol{\theta}$, and we consider the method as part of a larger statistical framework at a later point.

3.2 Exact conditioned processes for finite-state space models

Setting: We have available a parametric description of the generative model in the form of the triplet shown in equation (14), a correctly distributed sample of the posterior of model parameters $\boldsymbol{\theta}$, and a fixed set of observations $\boldsymbol{y}^{(1:T)}$.

For the finite-state setting considered in this work the backward filtering forward guiding method allows us to construct (in theory) a *conditioned process* $\mathcal{X}^\star$ with distribution

$$P^\star_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right). \tag{21}$$

This process is by definition conditioned on the observations $\boldsymbol{y}^{(1:T)}$ and we suppress the remaining dependence of the initial state and/or transition model on $\boldsymbol{\theta}$ with subscripts for clarity.

The conditioned process $\mathcal{X}^\star$ is not constructed as one large monolithic object but instead by operating on the constituents of the original process $\mathcal{X}$. We will show how a re-weighting of the initial state and later transitions with appropriate *guiding terms* allows us to construct such the conditioned process in a compositional manner, and subsequently show how the guiding terms can be computed. In this subsection we will first ignore the conditionally independent state variables for reasons that will be elucidated at a later point.

3.2.1 An intuitive introduction to the change of measure operation

By definition the initial state of the conditioned process must satisfy the following relation:

$$P^\star_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)} \mid \boldsymbol{X}^{(1)}\right)}{\sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)} \mid \boldsymbol{X}^{(1)}\right)}. \tag{22}$$

The numerator of the term on the right hand side of the preceding display should be considered a re-weighting of the initial state distribution according to the likelihood of the observations, and the term in the denominator can be considered a normalising constant. The re-weighting operation ensures that the first time step of the conditioned process is correctly conditioned on the observations, and we refer to such an operation as a *change of measure*.

A similar change of measure for transitions of the process at later time steps results in a purely compositional construction of the conditioned process. Later transitions must satisfy the following relation:

$$P^\star_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(t:T)} = \boldsymbol{y}^{(t:T)} \mid \boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(t:T)} = \boldsymbol{y}^{(t:T)} \mid \boldsymbol{X}^{(t)}\right)}. \tag{23}$$

The preceding relation is less obvious and a formal proof is provided at a later point. It follows from the two preceding relations that if we can compute the likelihood terms used in the change of measure operations for all time steps that we can incrementally guide a sample through the structure of the forward model, resulting in correctly distributed samples.

3.2.2 The backward information filter

We now concern ourselves with the computation of the closed-form conditional likelihood terms encountered previously, referred to as *backward filtering*. Throughout the backward filtering procedure the model parameter $\boldsymbol{\theta}$ is fixed and therefore we now suppress its notation completely.

Definition 3.1. Define the ***h*-transform** of a time step:

$$h_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t:T)} = \boldsymbol{y}^{(t:T)} \mid \boldsymbol{X}^{(t)}\right). \tag{24}$$

In the setting considered in this work an h-transform represents the conditional likelihood for a time step given all observations for later time steps. The *backward information filter* allows us to compute[2] all h-transforms recursively, starting at the terminal time step T and moving to earlier time steps incrementally.

Theorem 3.2. Successive h-transforms are related through the following relation:

$$h_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) h_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{25}$$

Proof: By the conditional independence of observations it immediately follows that

$$P\left(\boldsymbol{Y}^{(t:T)} = \boldsymbol{y}^{(t:T)} \mid \boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times P\left(\boldsymbol{Y}^{(t+1:T)} = \boldsymbol{y}^{(t+1:T)} \mid \boldsymbol{X}^{(t)}\right),$$
$$P\left(\boldsymbol{Y}^{(t+1:T)} = \boldsymbol{y}^{(t+1:T)} \mid \boldsymbol{X}^{(t)}\right) = \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) P\left(\boldsymbol{Y}^{(t+1:T)} = \boldsymbol{y}^{(t+1:T)} \mid \boldsymbol{X}^{(t+1)}\right).$$

A graphical representation of this recursion is shown in Figure 6. Van der Meulen & Schauer (2022) refer to the summation operation in equation (25) as *pullback*, and denote the operation and resulting term with

$$h_{\to t+1}\left(\boldsymbol{X}^{(t)}\right) = \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) h_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{26}$$

In the original work the backward filtering forward guiding method is considered for models featuring more general graph structures, including directed trees and directed acyclic graphs. In such settings it proves fruitful to be very explicit in denoting the terms which appear during the backward filtering procedure. The other term appearing in equation (25) is therefore similarly referred to as *pullback from an observation*:

$$h_{\to \mathcal{V}_t}\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right), \tag{27}$$

and the multiplication of the two resulting terms is referred to as *fusion*:

$$h_t\left(\boldsymbol{X}^{(t)}\right) = h_{\to \mathcal{V}_t}\left(\boldsymbol{X}^{(t)}\right) \times h_{\to t+1}\left(\boldsymbol{X}^{(t)}\right). \tag{28}$$

With these three operations the authors introduce a framework which allows for backward filtering procedures which operate directly on the graph structure. The setting considered in this work is comparatively simple, and because the methods I propose in this work do not necessarily fit into this pullback-fusion framework but rather compute the guiding terms for time steps directly we will refrain from using this terminology.

[2]Note that by *computing* an h-transform h_t we implicitly mean computing the values of $\boldsymbol{x}^{(t)} \mapsto h_t\left(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}\right)$.

3.2.3 Exact forward guiding

We now consider the second phase of the method, referred to as *forward guiding*. Recall that an instance of the forward model and the backward filtering procedure both assume the same fixed model parameter values, the notation of which has been suppressed for clarity.

Theorem 3.3. Let $\mathcal{X}$ be discrete-time finite-state time-invariant Markov process described by the familiar triplet form and let the maps $\boldsymbol{x}^{(t)} \mapsto h_t\left(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}\right)$ be specified for all time steps t.

The conditioned process $\mathcal{X}^\star$ constructed by a change of measure with the h-transforms computed for all time steps t features the exact conditional distribution of the forward model given the observations.

The initial state of $\mathcal{X}^\star$ is given by the relation:

$$P^\star\left(\boldsymbol{X}^{(1)}\right) = \frac{P\left(\boldsymbol{X}^{(1)}\right) h_1\left(\boldsymbol{X}^{(1)}\right)}{\sum_{\boldsymbol{X}^{(1)}} P\left(\boldsymbol{X}^{(1)}\right) h_1\left(\boldsymbol{X}^{(1)}\right)}, \tag{29}$$

and transitions at later time steps are given by the relation:

$$P^\star\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) = \frac{P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) h_t\left(\boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t)}} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) h_t\left(\boldsymbol{X}^{(t)}\right)}. \tag{30}$$

Proof. It follows from equation (29) and equation (30) that the joint distribution of $\mathcal{X}^\star$ is given by

$$P^\star\left(\boldsymbol{X}^{(1:T)}\right) = \frac{P\left(\boldsymbol{X}^{(1:T)}\right)}{\sum_{\boldsymbol{X}^{(1)}} P\left(\boldsymbol{X}^{(1)}\right) h_1\left(\boldsymbol{X}^{(1)}\right)} \times \frac{\prod_{t=1}^{T} h_t\left(\boldsymbol{X}^{(t)}\right)}{\prod_{t=2}^{T} \sum_{\boldsymbol{X}^{(t)}} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) h_t\left(\boldsymbol{X}^{(t)}\right)}.$$

The denominator in the first term on the right hand side of the preceding display equates to the likelihood of the observations $\boldsymbol{y}^{(1:T)}$, and by cancellation of terms the second term further simplifies to the product of only observation model terms:

$$P^\star\left(\boldsymbol{X}^{(1:T)}\right) = \frac{P\left(\boldsymbol{X}^{(1:T)}\right)}{P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)} \mid \boldsymbol{X}^{(1:T)}\right).$$

It immediately follows that the resulting relation in the preceding display is true by definition.

3.2.4 Discussion

Recall that in the finite-state setting considered in this work the h-transforms represent closed-from conditional likelihood terms given all 'subsequent' observations $\boldsymbol{y}^{(t:T)}$ and recall that the state space of $\mathcal{X}$ is decomposed into disjoint subspaces corresponding to interacting sub-processes, such that the transition model factorises as

$$P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} P\left(X_i^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t)}\right). \tag{31}$$

Because of the conditional independence between state variables in the transition model one may on first consideration hope that the conditional likelihood terms represented by h-transforms also admit some compact factorisation. However, upon further study it becomes clear that over multiple time steps even the states of 'distant' sub-processes become correlated due to the interaction of sub-processes 'between' them.[3] A graphical representation of this phenomenon is shown in Figure 7.

Due to this effect the h-transforms do not admit any factorisation in general and must instead be represented as singular objects (i.e. vectors) in computer memory. Because the joint state space of a model grows exponentially with respect to the number of state variables it follows that representing h-transforms for state space models featuring a large number of sub-processes is intractable for computational considerations, and therefore so is directly sampling from the conditional distribution as we have previously described it.

Furthermore, in the description presented in this subsection we have assumed that we can sample from the posterior distribution of the model parameters given the observations. However, if computing the h-transforms is intractable then so is directly computing the likelihood given the model parameters, as this implicitly requires summation over all possible states of the forward model:

$$P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = \sum_{\boldsymbol{X}^{(1:T)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = \sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) h_1\left(\boldsymbol{X}^{(1)}\right). \tag{32}$$

It follows that if the h-transforms are intractable then so is the likelihood, and thus it is also intractable to directly sample from the posterior of model parameters given the observations.

The key insight provided by Van der Meulen & Schauer (2022) is that computing an approximation of the likelihood and constructing a guided process with *approximate* guiding terms allows us to sample from a process featuring a distribution that approximates the exact distribution of interest, with known likelihood ratio between the two. Weighted samples from this approximate guided process not only correct for the likelihood ratio with respect to the exact distribution, but may also account for the approximated likelihood of the observations given $\boldsymbol{\theta}$ itself. In this way the method enables *exact* inference while only requiring an approximate (and most importantly) tractable backward information filtering procedure.

In the following subsection we derive the approach for sampling from the distribution of the exact conditioned process $\mathcal{X}^\star$ through weighted sampling of a tractable secondary process. Here we assume that we can compute the necessary approximate guiding terms and we describe our approach to actually computing these terms in section 4. We additionally derive the fully Bayesian approach for sampling from the posterior of latent states and model parameters jointly in Appendix E.

[3] To be explicit: If the sub-processes interact then the conditional likelihood does not factorise in general.

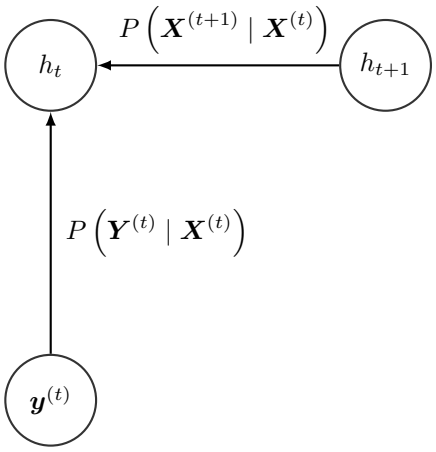

Figure 6: A representation of the recursion for the backward filtering procedure.

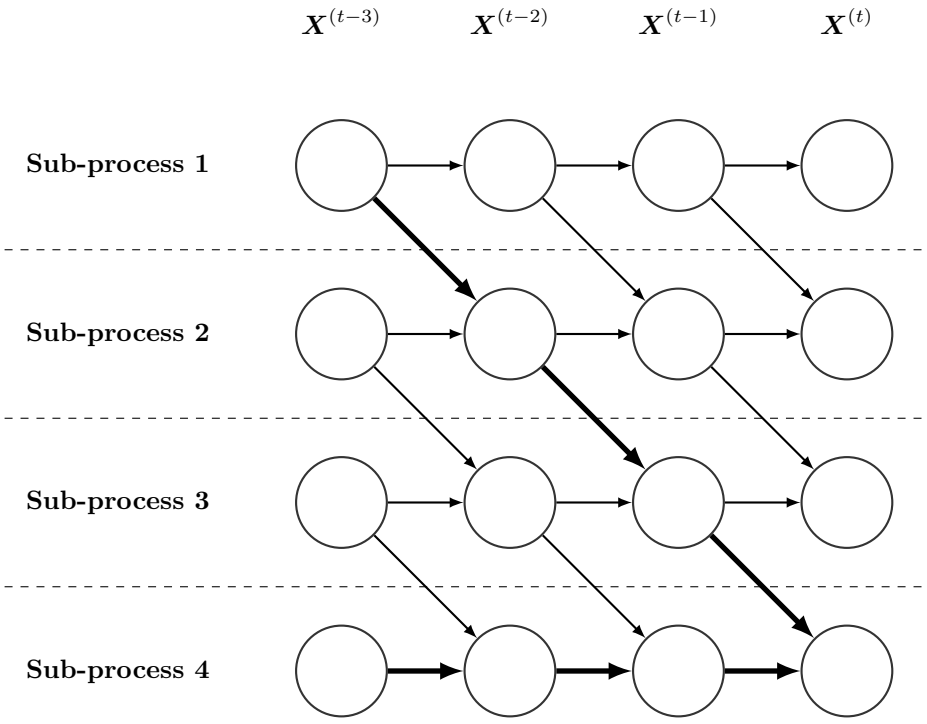

Figure 7: A representation of the latent states of four sub-processes, each of which interacts the sub-process immediately 'above' it, i.e. the evolution of sub-process $\mathcal{X}_i$ depends on the state of sub-process $\mathcal{X}_{i-1}$. Circles represent state variables and directed edges indicate conditional dependence between them. Time flows from left to right. Note that over multiple time steps even 'distant' sub-processes exert influence on one another due to the interaction of sub-processes 'between' them. This effect is highlighted with darker arrows.

3.3 Forward guiding with approximate guiding terms

Suppose that we can compute some tractable guiding terms g which approximate the exact h-transforms. We can then construct a *guided process* $\mathcal{X}^\circ$ by guiding with these functions in precisely the same way as before. If the functions g approximate the exact conditional likelihood terms well then we may reasonably expect the distribution of the guided process $\mathcal{X}^\circ$ to also be a good approximation of the exact conditioned process $\mathcal{X}^\star$, and most importantly we can additionally account for the likelihood ratio between the distributions of the two.

Definition 3.4. Let $\mathcal{X}$ be a discrete-time finite-state time-invariant Markov process described by the familiar triplet form and let the maps $\boldsymbol{x}^{(t)} \mapsto g_t\left(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}\right)$ be specified for all time steps t.

We define the guided process $\mathcal{X}^\circ$ as the process constructed by a change of measure with the guiding terms g. Note that the guiding terms do not directly depend on the model parameters $\boldsymbol{\theta}$.

The initial state of $\mathcal{X}^\circ$ is given by the relation:

$$P^\circ_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) g_1\left(\boldsymbol{X}^{(1)}\right)}{\sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) g_1\left(\boldsymbol{X}^{(1)}\right)}, \tag{33}$$

and the transitions at later time steps are given by the relation:

$$P^\circ_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}. \tag{34}$$

For a given instance of the forward model given the model parameters $\boldsymbol{\theta}$ we seek to sample from the distribution of $\mathcal{X}^\star$. We are therefore interested in the likelihood ratio between samples from the conditioned process and the guided process $\mathcal{X}^\circ$. Denote the law of $\mathcal{X}^\star$ with $\mathbb{P}^\star_{\boldsymbol{\theta}}$ and note that

$$\frac{\mathrm{d}\mathbb{P}^\star_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right) = \frac{1}{P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times \prod_{t=1}^{T} P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right), \tag{35}$$

where $\mathbb{P}_{\boldsymbol{\theta}}$ denotes the law of $\mathcal{X}$. The proof is a direct consequence of theorem 3.3.

Theorem 3.5. Let $\mathbb{P}^\circ_{\boldsymbol{\theta}}$ denote the law of $\mathcal{X}^\circ$. If $\mathbb{P}^\star_{\boldsymbol{\theta}} \ll \mathbb{P}^\circ_{\boldsymbol{\theta}}$, then

$$\frac{\mathrm{d}\mathbb{P}^\star_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}^\circ_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right) = \frac{\widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times \Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right), \tag{36}$$

where

$$\widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = \sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) g_1\left(\boldsymbol{X}^{(1)}\right), \tag{37}$$

and

$$\Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) = \prod_{t=1}^{T-1} \left(\frac{P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right)}{g_t\left(\boldsymbol{X}^{(t)}\right)} \right) \tag{38}$$

$$\times \left(\frac{P\left(\boldsymbol{Y}^{(T)} = \boldsymbol{y}^{(T)} \mid \boldsymbol{X}^{(T)}\right)}{g_T\left(\boldsymbol{X}^{(T)}\right)} \right). \tag{39}$$

Proof. It follows from equation (33) and equation (34) that the joint distribution of $\mathcal{X}^\circ$ is given by

$$P^\circ_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right)}{\sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) g_1\left(\boldsymbol{X}^{(1)}\right)} \times \frac{\prod_{t=1}^{T} g_t\left(\boldsymbol{X}^{(t)}\right)}{\prod_{t=2}^{T} \sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)},$$

and as a consequence

$$\begin{aligned}\frac{\mathrm{d}\mathbb{P}^\circ_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right) = &\frac{1}{\sum_{\boldsymbol{X}^{(1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1)}\right) g_1\left(\boldsymbol{X}^{(1)}\right)} \\ &\times \prod_{t=1}^{T-1}\left(\frac{g_t\left(\boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t+1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right)}\right) \\ &\times g_T\left(\boldsymbol{X}^{(T)}\right).\end{aligned}$$

The result now follows from

$$\frac{\mathrm{d}\mathbb{P}^\star_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}^\circ_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right) = \frac{\mathrm{d}\mathbb{P}^\star_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right) \times \left(\frac{\mathrm{d}\mathbb{P}^\circ_{\boldsymbol{\theta}}}{\mathrm{d}\mathbb{P}_{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)}\right)\right)^{-1},$$

substituting in the appropriate relations.

Upon inspection of equation (36) we see that we can target the exact conditional distribution by sampling from a tractable process $\mathcal{X}^\circ$ and *up-* or *down-weighting* the resulting samples depending on the specific realisation, i.e.

$$P^\star_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) = \frac{\widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times \Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times P^\circ_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right). \tag{40}$$

We can substitute the preceding relation into the joint distribution for the model parameters and latent states shown in equation (20). The resulting relation still includes certain intractable terms which must be dealt with appropriately, and we derive a full Gibbs sampling treatment of the problem at hand at a later point.

Remark 3.6. Note that we do not need to compute the entire guided transition probability distribution, but rather just

$$P^\circ_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}. \tag{41}$$

We may say that we compute the required weights and guided probability terms ‘at sample-time’.

3.4 Discussion

In this section we introduced the backward filtering forward guiding method for finite-state space models specifically. We have demonstrated that even if the exact conditional likelihood terms are intractable that we can still target the exact distribution of interest through weighted sampling of a guided process constructed with approximate guiding terms.

Recall that we introduce approximate guiding terms g not because the exact h-transforms do not admit closed-form solutions, but instead because such terms must be represented as intractably large singular vectors as they do not admit any factorisation in general. A guiding term g_t must necessarily include an entry for each possible assignment $\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}$, and to limit the memory requirements we propose using guiding terms which exploit the natural factorisation of the state space via the state variables.

The most straightforward way to achieve such a factorisation is by specifying guiding terms g as products of purely marginal terms, i.e.

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} \phi_i\left(X_i^{(t)}\right). \tag{42}$$

With such *fully factorised* guiding terms it follows that the guided transition probability also fully factorises, i.e.

$$P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} P_{\boldsymbol{\theta}}\left(X_i^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}\right) \phi_i\left(X_i^{(t)}\right), \tag{43}$$

and as a result we can then sample the state variables independently of one another in the forward guiding phase. Van der Meulen & Schauer (2022) note that this implies that sampling from such a guided process $\mathcal{X}^\circ$ has similar complexity to sampling from the original process $\mathcal{X}$, "predisposing guided processes for sampling based inference."

While being able to sample all state variables independently is a desirable property, using fully factorised guiding terms throughout the backward filtering procedure may be too restrictive. We have a strong intuition that interacting sub-processes may be highly correlated, and such correlations can clearly not be represented by fully factorised guiding terms. Guiding terms must additionally be computed recursively, and thus 'earlier' ($t' < t$) guiding terms $g_{t'}$ may inherit errors made in the approximation of 'later' guiding terms g_t.

In the following section we therefore propose an *expectation propagation* approach for the backward filtering procedure which allows us to approximately maintain certain desired correlations. The resulting *rich approximations* of the conditional likelihood terms may not allow the state variables to be sampled independently of one another. However, after computing rich guiding terms g for all time steps we are free to project the terms onto another factorised family of functions, preserving (part of) the conditional independence between state variables in the forward guiding phase. We may then expect the resulting fully factorised guiding terms to still be 'better' than if we worked with fully factorised approximations throughout the backward filtering procedure, as we do not incur the approximation errors made at later time steps. Alternatively we can also guide with 'rich' guiding terms g directly, a result I derive at a later point. We refer to this as *dynamic guiding*.

4 Backward filtering with expectation propagation

In this section we introduce an expectation propagation approach to the backward filtering phase for finite-state space models. We consider expectation propagation as it relates to backward filtering forward guiding specifically and discuss related theory provided by Minka (2005) at a later point. We first provide a high-level description of the approach in general. We then describe additional assumed structure before considering remaining aspects in subsequent subsections.

4.1 Approach

Recall that the exact h-transforms are computationally intractable because the conditional likelihood terms they represent do not factorise. We instead construct a guided process with *approximate* guiding terms g and in this work we specifically propose the use of guiding terms which exploit the natural factorisation of the state space via the state variables.

We are therefore interested in 'rich' yet tractable approximations of the conditional likelihood for all time steps. This is achieved by defining a tractable *guiding family* $\mathcal{Q}$ and computing the 'optimal' function q in $\mathcal{Q}$ for each time step. Specifically, $\mathcal{Q}$ is defined as a family of factorised functions, and by allowing this family to contain factors of *sets of state variables* we can maintain correlations between the corresponding sub-processes.

Recall that the exact backward filtering recursion is given by

$$h_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) h_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{44}$$

For reasons that will be elucidated at a later point the approximate backward information filter must be a similar recursive procedure, and thus without specifying the form of the guiding terms g we consider the relation

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{45}$$

For any non-trivial transition model the *exact recursion* $\widehat{g}_t$ is generally not a member of $\mathcal{Q}$ due to the interaction between sub-processes inducing further correlations which can not be represented by any tractable guiding family. Expectation propagation therefore consists of finally computing

$$g_t = \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(\frac{\widehat{g}_t}{\|\widehat{g}_t\|_1} \,||\, q\right), \tag{46}$$

which we refer to as the *M-projection* of $\widehat{g}_t$.[4] The expectation propagation approach to backward filtering can therefore be seen as a series of *recurse*- and *project*-steps, as shown in Figure 8.

In a later subsection we prove that an M-projection of $\widehat{g}_t$ can be represented as a product of a number of its 'moments'. For a multivariate categorical distribution such moments are given by expectations for the sets of variables between which we wish to approximate the correlation, and thus we seek to compute a number of (suitably normalised) marginal distributions of $\widehat{g}_t$.

By exploiting the fact that the transition model, the observation model and the guiding term g_{t+1} all factorise in some way we can then compute the relevant marginals of $\widehat{g}_t$ with the *junction tree algorithm*, providing us with an efficient and general approach to computing M-projections.

[4] Note that with $\mathrm{KL}\left(p \,||\, q\right)$ we denote the well-known *relative entropy* from q to p.

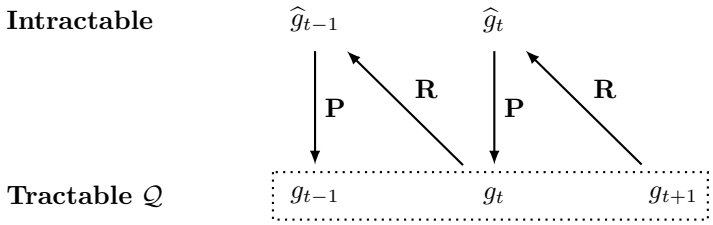

Figure 8: A representation of the expectation propagation approach to backward filtering. An **R** represents a *recurse*-step and a **P** represents a *project*-step. Note that the guiding terms g_{t-1}, g_t, and g_{t+1} are by definition members of $\mathcal{Q}$ and that the exact recursion terms $\widehat{g}_t$ and $\widehat{g}_{t-1}$ are not.

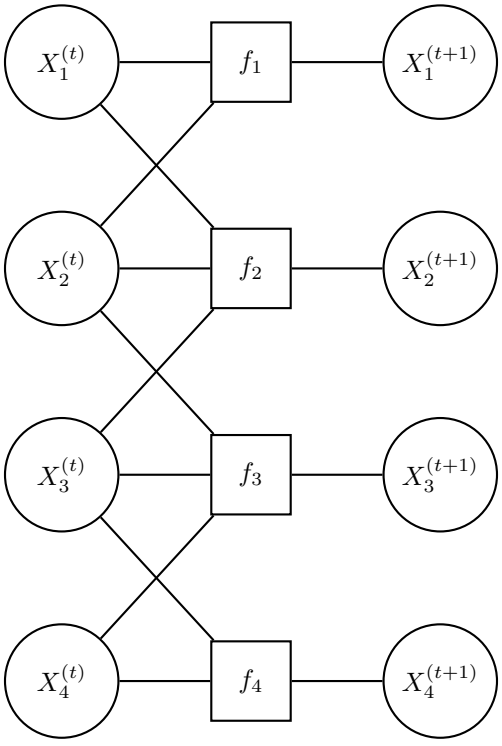

Figure 9: A factor graph representation of the transition model featuring in example 4.2. Circles represent variable nodes and squares represent factor nodes. Undirected edges between variable nodes and factor nodes represent the coupling of variables through the factors. Note that we have divided the set of variable nodes by their time step, and also note that the state variables in time step $t+1$ are each directly coupled through only one factor: This is expected, as the state variables are conditionally independent of another given the preceding state.

4.2 Representation and additional assumed structure

As practitioners we are free to choose the form of the guiding family $\mathcal{Q}$ and in general we will choose to represent the correlations between sub-processes which directly interact with one another, as we expect those to be strongest. The choice of $\mathcal{Q}$ will therefore generally depend on the transition model of the specific process under consideration.

The junction tree algorithm allows us to compute the required marginals for $\widehat{g}_t$ for many forms of transition models (and guiding families) in an efficient manner, but if we do not simplify the interaction between sub-processes somewhat then the notation required in the following subsections will be extremely involved.

We will therefore impose an additional assumption which contributes greatly to ease of exposition. Understanding of this assumption requires understanding of the *junction tree* representation of a (conditional) probability distribution, which we therefore first introduce here before describing the related junction tree algorithm at a later point. We additionally introduce *factor graphs*, as junction trees are perhaps best understood as alternative representations of factor graphs.

4.2.1 Factor graphs

A *factor graph* provides a graphical representation of the factorisation of a function featuring a number of variables coupled by some *factors*. Factor graphs can be used to represent any factorisable function and will in this work be employed to represent transition models and guiding terms.

Definition 4.1. A **factor graph** is a bipartite graph which represents the factorisation of some function Q. Given a suitable factorisation

$$Q\left(\boldsymbol{X}\right) = \prod_{i=1}^{N} f_i\left(\boldsymbol{S}_i\right) \tag{47}$$

where $\boldsymbol{S}_i \subseteq \boldsymbol{X}$, the corresponding factor graph $\mathcal{G} = (\boldsymbol{X}, \mathcal{F}, \mathcal{E})$ consists of the **variable nodes** $\boldsymbol{X}$, the **factor nodes** $\mathcal{F}$ and a set of undirected edges $\mathcal{E}$ between variable nodes and factor nodes.

The graph features a variable node for each variable $X_j \subset \boldsymbol{X}$ featuring in the function and a factor node for each factor f_i, and we include an edge between a variable node and a factor node if the variable features in the *scope* of that factor, i.e. if $X_j \subseteq \boldsymbol{S}_i$.

Example 4.2. Consider a process $\mathcal{X}$ which consists of four sub-processes. Each sub-process $\mathcal{X}_i$ interacts with its immediate neighbours $\mathcal{X}_{i-1}$ and $\mathcal{X}_{i+1}$, with suitable restrictions for the 'edge' processes $\mathcal{X}_1$ and $\mathcal{X}_4$.

The factorisation of the transition model of $\mathcal{X}$ can then be represented as

$$Q\left(\boldsymbol{X}^{(t:t+1)}\right) = f_1\left(\boldsymbol{X}_{1:2}^{(t)}, X_1^{(t+1)}\right) f_2\left(\boldsymbol{X}_{1:3}^{(t)}, X_2^{(t+1)}\right) f_3\left(\boldsymbol{X}_{2:4}^{(t)}, X_3^{(t+1)}\right) f_4\left(\boldsymbol{X}_{3:4}^{(t)}, X_4^{(t+1)}\right), \tag{48}$$

and a graphical representation of the factorisation as a factor graph is shown in Figure 9.

4.2.2 Junction trees

Whereas a factor graph provides a graphical representation of the factorisation of a function, *junction trees* serve as computational tools for efficiently answering queries about functions, such as computing (conditional) marginal distributions for sets of variables. We describe the junction tree algorithm for this task at a later point and consider and motivate only its representation here.

Definition 4.3. A **junction tree** is an undirected graph which represents the factorisation of some function Q. Given a suitable factorisation

$$Q(\boldsymbol{X}) = \prod_{i=1}^{N} f_i(\boldsymbol{S}_i) \tag{49}$$

where $\boldsymbol{S}_i \subseteq \boldsymbol{X}$, a junction tree $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ consists of the **regions** $\mathcal{R}$ and a set of undirected edges between certain regions $\mathcal{E}$.

A region may consist of both factors and variables, and we may include an edge between any two regions which have at least one variable in common. Additionally, the following restrictions apply:

- Each factor f_i must be assigned to exactly one region.
- If a region is assigned the factor f_i, then it must also contain the variables $\boldsymbol{S}_i$.
- The graph must be an undirected tree, i.e. a connected graph with no cycles.
- The graph must respect the *running intersection property*.

Definition 4.4. Let $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ be a valid junction tree for a set of variables $\boldsymbol{X}$. The **running intersection property** mandates that for each variable $X_i \subset \boldsymbol{X}$ the regions of $\mathcal{J}$ featuring that variable must form a **connected sub-tree**.

As seen in Figure 9, the factor graph for any non-trivial transition model will generally feature many *cycles*. The presence of cycles poses numerous computational challenges and the junction tree representation alleviates this by clustering sets of variables into regions, thereby transforming the factor graph into a tree and thus enabling the use of efficient *message passing algorithms*.

Example 4.5. Consider the situation described in example 4.2. A graphical representation of a valid junction tree $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ with $\mathcal{R} = \{r_1, r_2, r_3, r_4\}$ and $\mathcal{E} = \{(r_1, r_2), (r_2, r_3), (r_3, r_4)\}$, where

$$r_i = \left\{ \boldsymbol{X}_{\mathrm{ne}(i)}^{(t)}, f_i, X_i^{(t+1)} \right\} \tag{50}$$

is shown in Figure 10. Note that the underlying graph for this junction tree is a *line graph*, i.e. it does not feature any branches. Also note that a function does not necessarily admit only one unique junction tree representation: In the preceding example we may merge any two 'adjacent' regions into larger ones, and the resulting graph will still be a valid junction tree. For a given function we are generally interested in the junction tree featuring the 'smallest' regions possible (in terms of variables), and one may verify that the junction tree shown satisfies this property.

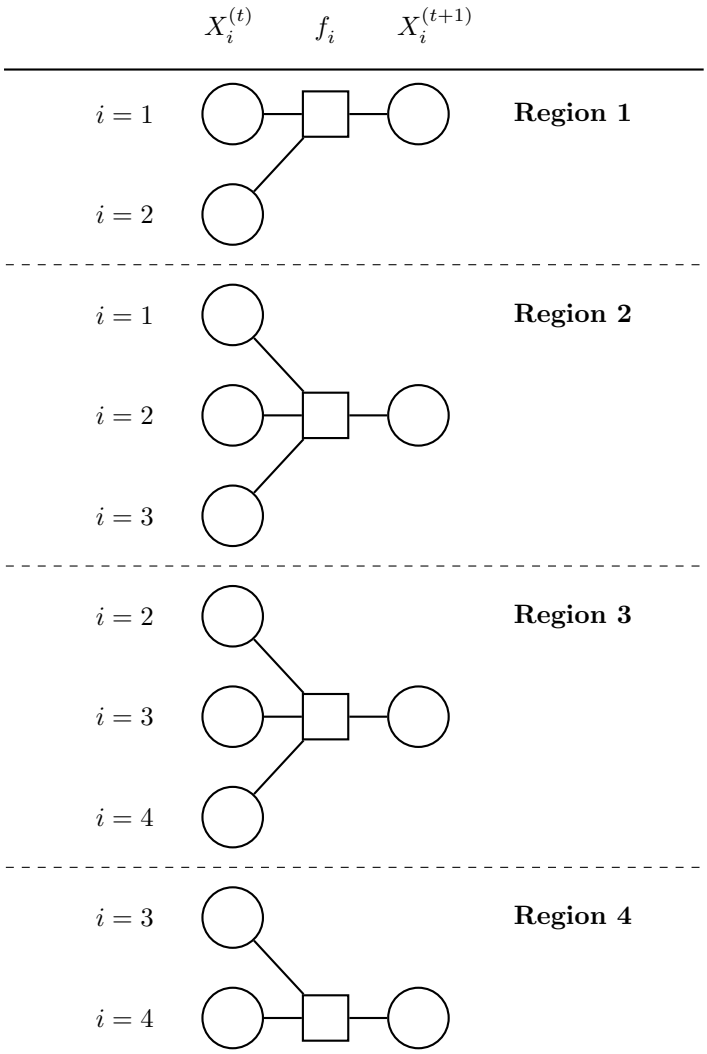

Figure 10: A representation of a valid junction tree for transition model featuring in example 4.2. The junction tree consists of four regions. The factor nodes and variable nodes assigned to each region are displayed as factor graphs. Circles represent variable nodes and squares represent factor nodes. Note that state variables in time step t feature in multiple regions. Note that the 'nodes' of the junction tree are the regions themselves and that we have displayed the variables and factors assigned to each region as factor graphs merely for clarity. Note that if we 'overlay' or merge all four regions that the resulting object would be equivalent to the factor graph shown in Figure 9: The point of a junction tree representation of a factorisation is to impose acyclic structure between regions, enabling the use of efficient message passing algorithms.

4.2.3 Additional assumed structure

Recall that the transition model generally factorises as

$$P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} P\left(X_i^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t)}\right). \tag{51}$$

The goal of this work is to introduce an expectation propagation approach to backward filtering and not to derive arbitrarily complex junction trees. To facilitate the derivations and discussions in later subsections we will therefore require that the transition model of a process under consideration admits a particularly simple junction tree representation.

Assumption 4.6. Throughout this work we assume that the neighbourhood sets $\{\mathrm{ne}(i),\ i = 1 \dots N\}$ allow us to represent the transition model of a process under consideration as a valid junction tree $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ with $\mathcal{R} = \{r_i,\ i = 1 \dots N\}$ and $\mathcal{E} = \{(r_i, r_{i+1}),\ i = 1 \dots N-1\}$, where

$$r_i = \left\{\boldsymbol{X}_{\mathrm{ne}(i)}^{(t)}, f_i, X_i^{(t+1)}\right\}. \tag{52}$$

This assumption requires that the sub-processes may be considered to be *globally ordered*, such that $\mathcal{X}_{i-1}$ may be considered the sub-process to the 'left' of $\mathcal{X}_i$, and $\mathcal{X}_{i+1}$ may be considered the sub-process to the 'right' of $\mathcal{X}_i$.

The running intersection property then requires that if sub-process $\mathcal{X}_i$ interacts with $\mathcal{X}_K$, that it must also interact with all sub-processes $\mathcal{X}_j$, where $(k < j < i)$ if $k < i$ or $(i < j < k)$ if $i < k$. We may therefore write

$$\mathrm{ne}(i) = \left\{l_i^-, \dots, l_i^+\right\}, \tag{53}$$

where with l_i^- and l_i^+ we denote the indices of the 'farthest' sub-processes that a sub-process $\mathcal{X}_i$ interacts with to its respective 'left' and 'right'. The running intersection property additionally requires that the sequences of *lower* and *upper interaction bounds*

$$\mathcal{I}^- = \left\{l_i^-,\ i = 1 \dots N\right\}, \tag{54}$$

$$\mathcal{I}^+ = \left\{l_i^+,\ i = 1 \dots N\right\} \tag{55}$$

are both non-decreasing.

An intuitive way we can think of this is that each sub-process index references some location in one-dimensional physical space. We then require sub-processes to interact only with other sub-processes within some range in each direction, with the restriction that a sub-process can not interact with sub-processes 'farther' to either side than the sub-process immediately adjacent to it on that side interacts with.

From this restriction for the transition model the guiding family $\mathcal{Q}$ inherits a similar line-graph structure, simplifying the involved notation and graphical representation of the involved objects. Note that the example problem described in section 2 satisfies this assumption, as does any process where each sub-process interacts with the k sub-processes to its 'left' and 'right' (such as the transition model considered in example 4.2 and example 4.5).

To be explicit: assumption 4.6 is imposed only so that the following discussions are easier to follow for the reader, as it makes the involved notation of all involved objects much simpler. We describe the implications of relaxation of this assumption at a later point.

4.3 The M-projection

We now describe the forms of guiding functions featuring in expectation propagation backward filtering procedures. Consider the relation for the exact recursion once more:

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{56}$$

The term $\widehat{g}_t$ is an approximation of the conditional likelihood of all subsequent observations and is therefore not a (normalised) probability distribution. We will treat it as such, however, and therefore formally define its *M-projection* as

$$g_t = \arg\min_{q \in \mathcal{Q}} \mathrm{KL}\left(\frac{\widehat{g}_t}{\|\widehat{g}_t\|_1} \;||\; q\right). \tag{57}$$

Recall that we seek to approximate $\widehat{g}_t$ in a tractable factored form. The most straightforward approach is to define the guiding family $\mathcal{Q}$ as a product of strictly marginal terms, such that

$$\mathcal{Q}_{\mathrm{FF}} = \left\{ \left(\prod_{i=1}^{N} \phi_i\left(x_i^{(t)}\right) \right) \text{ such that } \left(\phi_i : x_i^{(t)} \mapsto [0,1], \sum_{X_i^{(t)}} \phi_i\left(x_i^{(t)}\right) = 1 \right) \; \forall i = 1 \dots N \right\}. \tag{58}$$

In some situations such a *fully factorised* approximation may prove adequate, though for processes featuring a high degree of interaction between sub-processes we may wish to admit a richer approximation of the conditional likelihood to preserve part of the correlation between their states. A very natural idea is therefore to allow $\mathcal{Q}$ to contain factors of *sets* of state variables.

Recall that by assumption 4.6 the transition model of a process $\mathcal{X}$ must be representable as a junction tree given by a line graph and that the sub-processes are therefore 'global ordered'. We have a strong intuition that the strongest correlations will be between states of sub-processes which interact with each other 'most', i.e. those that are 'adjacent' to one another in one-dimensional space, and we will therefore seek to maintain correlations for precisely such sets. We refer to a set of states of 'adjacent' sub-processes as a set of *adjacent state variables* and refer to the chosen sets of adjacent state variables jointly as a *factorisation of the state space*.

Definition 4.7. Let $\mathcal{F}_{\mathcal{R}} = \left\{ \boldsymbol{X}_{r_1}^{(t)}, \dots, \boldsymbol{X}_{r_K}^{(t)} \right\}$ be a set of K sets of **adjacent state variables**, i.e.

$$\boldsymbol{X}_{r_i}^{(t)} = \left\{ X_j^{(t)}, \dots, X_{j+|r_i|-1}^{(t)} \right\}. \tag{59}$$

We refer to $\mathcal{F}_{\mathcal{R}}$ as a valid **factorisation of the state space** of a process $\mathcal{X}$ if and only if:

- (i) the chosen sets of state variables 'cover' the entire state space, i.e.

$$\bigcup_{i=1}^{K} \boldsymbol{X}_{r_i}^{(t)} \equiv \boldsymbol{X}^{(t)}, \tag{60}$$

- (ii) the order of **region indices** $r_i \in \mathcal{R} = \{r_1, \dots, r_K\}$ 'respects' the global ordering of the sub-processes of $\mathcal{X}$, i.e. we have that $X_1^{(t)} \subseteq \boldsymbol{X}_{r_1}^{(t)}$ and $X_N^{(t)} \subseteq \boldsymbol{X}_{r_K}^{(t)}$, and for all pairs of inner region indices (r_i, r_{i+1}):

$$\min(r_i) < \min(r_{i+1}), \tag{61}$$

$$\max(r_i) < \max(r_{i+1}). \tag{62}$$

Motivation for the first requirement is obvious and the second is added for ease of exposition. A graphical representation of an example of a valid factorisation is shown in Figure 11.

4.3.1 Moment matching

Let $\mathcal{F}_\mathcal{R}$ be a factorisation of the state space of some process $\mathcal{X}$. Due to assumption 4.6 the sets of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_\mathcal{R}$ may either (i) all be disjoint from one another, (ii) form one long 'chain' of intersecting sets, or (iii) form multiple shorter 'chains' of intersecting sets.

Let $f\left[\boldsymbol{X}_r\right]$ denote the marginal of a function $f : \boldsymbol{x} \mapsto [0, 1]$ for the subset of variables $\boldsymbol{X}_r \subset \boldsymbol{X}$, i.e.

$$f\left[\boldsymbol{X}_r\right] = \sum_{\boldsymbol{X} \setminus \boldsymbol{X}_r} f\left(\boldsymbol{X}\right). \tag{63}$$

To maintain the correlation for a set of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)}$ we may hope that the tractable approximation of $\widehat{g}_t$ features the same marginal distribution for that set as the (suitably normalised[5]) exact recursion itself, i.e.

$$g_t\left[\boldsymbol{X}_{r_i}^{(t)}\right] = c \times \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right], \tag{64}$$

where $c = \|\widehat{g}_t\|_1^{-1}$. Koller & Friedman (2009) prove for the three cases described previously that there exists a tractable function g_t which 'matches' the marginal distribution for each set $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_\mathcal{R}$, and show that we can construct such a function directly in terms of marginal distributions of $\widehat{g}_t$.

The authors additionally prove that the constructed function g_t minimises the KL-divergence to the exact recursion from the entire family of distributions $\mathcal{Q}$ encompassed by that factorisation. We refer to the construction of such a function as *moment matching*.

To be explicit: The chosen factorisation $\mathcal{F}_\mathcal{R}$ fixes the guiding family $\mathcal{Q}$, and the function $\widehat{g}_t$ which minimises the KL-divergence from $\mathcal{Q}$ to g_t is then computed by construction through moment matching. Further motivation for this approach is provided in Appendix B.

The third of the three cases resulting from assumption 4.6 can be seen as a combination of the first two, and for ease of exposition we therefore consider only those two cases in this work. In the remainder of this subsection we describe how to construct a function g_t for the two remaining cases and in the following subsection we describe how to actually compute the required marginal terms. The KL-divergence minimisation proof is outside of the scope of this work and may be found in Koller & Friedman (2009) (p. 277).

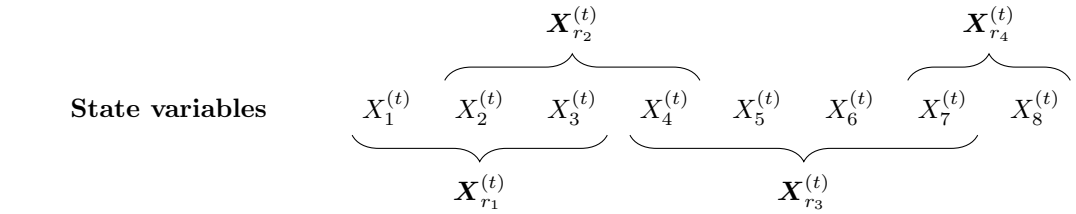

Figure 11: A representation of a valid factorisation of a state space which consists of eight state variables. Adjacent sets of state variables are grouped by brackets. Note that the order of region indices r_i 'respects' the global ordering of sub-process indices j.

[5]When referring to (marginal distributions of) $\widehat{g}_t$ we implicitly mean its normalised form as we are only interested in the relative difference of likelihoods of states, i.e. up to proportionality.

4.3.2 Simply factorised guiding families

If the sets of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_{\mathcal{R}}$ are all disjoint from one another then it is particularly simple to construct a tractable approximation g_t which can represent the desired correlations. We refer to such a factorisation as a *simple factorisation of the state space*.

Definition 4.8. We refer to a factorisation $\mathcal{F}_{\mathcal{R}}$ as a **simple factorisation of the state space** if all sets of adjacent state variables are disjoint from one another, i.e. if for all pairs (r_i, r_j) we have

$$\boldsymbol{X}_{r_i}^{(t)} \cap \boldsymbol{X}_{r_j}^{(t)} = \emptyset. \tag{65}$$

Theorem 4.9. Let $\mathcal{F}_{\mathcal{R}}$ be a simple factorisation of the state space of a process $\mathcal{X}$. To maintain the correlation of a set of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_{\mathcal{R}}$ we require that the approximation g_t must feature the same marginal distribution for that set of variables as $\widehat{g}_t$ itself, and because the sets of adjacent state variables are all disjoint the M-projection g_t is directly given by:

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{K} \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right]. \tag{66}$$

Proof. From the preceding equation it immediately follows that:

$$g_t\left[\boldsymbol{X}_{r_i}^{(t)}\right] = \sum_{\boldsymbol{X}^{(t)} \setminus \boldsymbol{X}_{r_i}^{(t)}} \prod_{j=1}^{K} \widehat{g}_t\left[\boldsymbol{X}_{r_j}^{(t)}\right] = \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right], \tag{67}$$

Note that for each desired correlation we have essentially included a (normalised) factor[6]

$$\phi_{r_i} : \boldsymbol{x}_{r_i}^{(t)} \mapsto [0, 1] \tag{68}$$

in the guiding family $\mathcal{Q}$. The guiding family is therefore directly defined by the chosen factorisation.

Definition 4.10. Let $\mathcal{F}_{\mathcal{R}}$ be a simple factorisation of the state space of a process $\mathcal{X}$. It follows that the **guiding family** $\mathcal{Q}_{\mathcal{R}}$ induced by $\mathcal{F}_{\mathcal{R}}$ is defined as:

$$\mathcal{Q}_{\mathcal{R}} = \left\{ \left(\prod_{r_i \in \mathcal{R}} \phi_{r_i}\left(\boldsymbol{x}_{r_i}^{(t)}\right) \right) \text{ such that } \left(\phi_{r_i} : \boldsymbol{x}_{r_i}^{(t)} \mapsto [0,1], \sum_{\boldsymbol{X}_{r_i}^{(t)}} \phi_{r_i}\left(\boldsymbol{x}_{r_i}^{(t)}\right) = 1 \right) \forall r_i \in \mathcal{R} \right\}. \tag{69}$$

Note that the fully factorised guiding family shown in equation (58) may be considered the 'simplest' factorised guiding family. Another particularly simple factorisation might be to maintain the correlations between disjoint pairs of immediately adjacent state variables.

Example 4.11. Consider the situation described in example 4.2. A valid simple factorisation of the state space of $\mathcal{X}$ may be given by the disjoint sets of adjacent state variables

$$\boldsymbol{X}_{r_1}^{(t)} = \left\{ X_1^{(t)}, X_2^{(t)} \right\}, \tag{70}$$

$$\boldsymbol{X}_{r_2}^{(t)} = \left\{ X_3^{(t)}, X_4^{(t)} \right\}. \tag{71}$$

A graphical representation of the induced guiding family is shown in Figure 12.

[6] Note once more that the function g_t minimises the KL-divergence from $\mathcal{Q}_{\mathcal{R}}$ to $\widehat{g}_t$ by construction.

4.3.3 Fully connected guiding families

A simple factorisation of the state space allows us to maintain correlations between a number of disjoint sets, such that each state variable features in the scope of exactly one factor of the induced guiding family. In some situations such a factorisation may prove adequate, though we may additionally be interested in maintaining the correlation for intersecting sets of adjacent state variables.

The intersecting sets of desired correlations may form one long 'chain' over the entire state space, from which it follows that even the correlation between two distant state variables which we do not represent explicitly can be partially maintained through the state variables 'between' them. We refer to such as factorisation as a *fully connected factorisation of the state space.*

Definition 4.12. We refer to a factorisation $\mathcal{F}_{\mathcal{R}}$ as a **fully connected factorisation of the state space** if we can represent the factorisation as a valid junction tree $\mathcal{J} = (\mathcal{F}_{\mathcal{R}}, \mathcal{E})$, where

$$\mathcal{E} = \left\{ \left(\boldsymbol{X}_{r_i}^{(t)}, \boldsymbol{X}_{r_{i+1}}^{(t)} \right),\ i = 1 \dots K-1 \right\}. \tag{72}$$

Theorem 4.13. Let $\mathcal{F}_{\mathcal{R}}$ be a fully connected factorisation of the state space of a process $\mathcal{X}$. To maintain the correlation of a set of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_{\mathcal{R}}$ we require that the approximation g_t must feature the same marginal distribution for that set of variables as $\widehat{g}_t$ itself.

However, because the sets of adjacent state variables intersect we must account for the 'over-counting' of the intersections $\boldsymbol{X}_{s_i}^{(t)} = \boldsymbol{X}_{r_i}^{(t)} \cap \boldsymbol{X}_{r_{i+1}}^{(t)}$, and thus the M-projection g_t is given by:

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \frac{\prod_{i=1}^{K} \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right]}{\prod_{j=1}^{K-1} \widehat{g}_t\left[\boldsymbol{X}_{s_j}^{(t)}\right]}. \tag{73}$$

Proof. First, note that for each desired correlation we have that

$$\sum_{\boldsymbol{X}_{r_i}^{(t)} \setminus \boldsymbol{X}_{s_i}^{(t)}} \frac{\widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right]}{\widehat{g}_t\left[\boldsymbol{X}_{s_i}^{(t)}\right]} = \frac{\widehat{g}_t\left[\boldsymbol{X}_{s_i}^{(t)}\right]}{\widehat{g}_t\left[\boldsymbol{X}_{s_i}^{(t)}\right]} = 1, \tag{74}$$

and note that a similar argument holds for summation from the 'other side.' It follows that when we compute the marginal of g_t for any set of adjacent state variables $\boldsymbol{X}_{r_i}^{(t)} \in \mathcal{F}_{\mathcal{R}}$ that terms in the numerator and the denominator cancel along the length of the 'chain', i.e.

$$g_t\left[\boldsymbol{X}_{r_i}^{(t)}\right] = \sum_{\boldsymbol{X}^{(t)} \setminus \boldsymbol{X}_{r_i}^{(t)}} g_t\left(\boldsymbol{X}^{(t)}\right) \tag{75}$$

$$= \left(\sum_{\boldsymbol{X}_{r_1}^{(t)} \setminus \boldsymbol{X}_{s_1}^{(t)}} \cdots \sum_{\boldsymbol{X}_{r_{i-1}}^{(t)} \setminus \boldsymbol{X}_{s_{i-1}}^{(t)}} \prod_{j=1}^{i-1} \frac{\widehat{g}_t\left[\boldsymbol{X}_{r_j}^{(t)}\right]}{\widehat{g}_t\left[\boldsymbol{X}_{s_j}^{(t)}\right]} \right) \times \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right] \tag{76}$$

$$\times \left(\sum_{\boldsymbol{X}_{r_{i+1}}^{(t)} \setminus \boldsymbol{X}_{s_i}^{(t)}} \cdots \sum_{\boldsymbol{X}_{r_K}^{(t)} \setminus \boldsymbol{X}_{s_{K-1}}^{(t)}} \prod_{j=i+1}^{K} \frac{\widehat{g}_t\left[\boldsymbol{X}_{r_j}^{(t)}\right]}{\widehat{g}_t\left[\boldsymbol{X}_{s_{j-1}}^{(t)}\right]} \right) = \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right], \tag{77}$$

and thus g_t again features the correct marginal distributions by construction and therefore minimises the KL-divergence from $\mathcal{Q}_{\mathcal{R}}$ to $\widehat{g}_t$.

The guiding family now consists of not only factors in the numerator, but also includes factors

$$\phi_{s_i} : \boldsymbol{x}_{s_i}^{(t)} \mapsto [0,1] \tag{78}$$

which account for the 'over-counting' of intersections of desired correlations in the denominator. The guiding family is therefore defined by both the chosen factorisation and the *separator sets* on the edges between regions of the junction tree defined by $\mathcal{F}_{\mathcal{R}}$.

Definition 4.14. Let $\mathcal{F}_{\mathcal{R}}$ be a fully connected factorisation of the state space of a process $\mathcal{X}$. It follows that the **guiding family** $\mathcal{Q}_{\mathcal{R}}$ induced by $\mathcal{F}_{\mathcal{R}}$ is defined as:

$$\mathcal{Q}_{\mathcal{R}} = \left\{ \left(\prod_{r_i \in \mathcal{R}} \phi_{r_i}\left(\boldsymbol{x}_{r_i}^{(t)}\right) \right) \text{ such that } \left(\phi_{r_i} : \boldsymbol{x}_{r_i}^{(t)} \mapsto [0,1]\,,\ \sum_{\boldsymbol{X}_{r_i}^{(t)}} \phi_{r_i}\left(\boldsymbol{x}_{r_i}^{(t)}\right) = 1 \right) \ \forall r_i \in \mathcal{R} \right\} \tag{79}$$

$$\times \left\{ \left(\prod_{s_i \in \mathcal{S}} \phi_{s_i}^{-1}\left(\boldsymbol{x}_{s_i}^{(t)}\right) \right) \text{ such that } \left(\phi_{s_i} : \boldsymbol{x}_{s_i}^{(t)} \mapsto [0,1]\,,\ \sum_{\boldsymbol{X}_{s_i}^{(t)}} \phi_{s_i}\left(\boldsymbol{x}_{s_i}^{(t)}\right) = 1 \right) \ \forall s_i \in \mathcal{S} \right\}, \tag{80}$$

where $\mathcal{S} = \{s_i = (r_i \cap r_{i+1})\,,\ i = 1 \ldots K-1\}$ is the set of region indices of **separator sets** on the edges between regions of the junction tree defined by $\mathcal{F}_{\mathcal{R}}$. Note that with $\phi_{s_i}^{-1}$ we denote the reciprocal of that factor, and not its inverse.

Example 4.15. Consider the situation described in example 4.2. A valid fully connected factorisation of the state space of $\mathcal{X}$ may be given by the intersecting sets of adjacent state variables

$$\boldsymbol{X}_{r_1}^{(t)} = \left\{ X_1^{(t)}, X_2^{(t)} \right\}, \tag{81}$$

$$\boldsymbol{X}_{r_2}^{(t)} = \left\{ X_2^{(t)}, X_3^{(t)} \right\}, \tag{82}$$

$$\boldsymbol{X}_{r_3}^{(t)} = \left\{ X_3^{(t)}, X_4^{(t)} \right\}, \tag{83}$$

with appropriate separator sets to account for the over-counting of state variables $X_2^{(t)}$ and $X_3^{(t)}$. A graphical representation of the induced guiding family is shown in Figure 13.

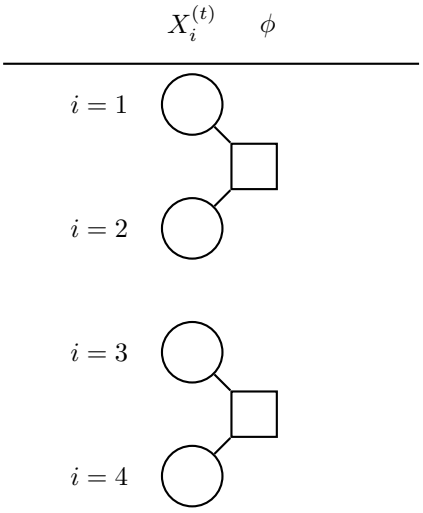

Figure 12: A representation of the induced guiding family featuring in example 4.11. Circles represent variable nodes and squares represent factor nodes. Note that the correlations between disjoint pairs of immediately adjacent state variables can be maintained with such a factorisation.

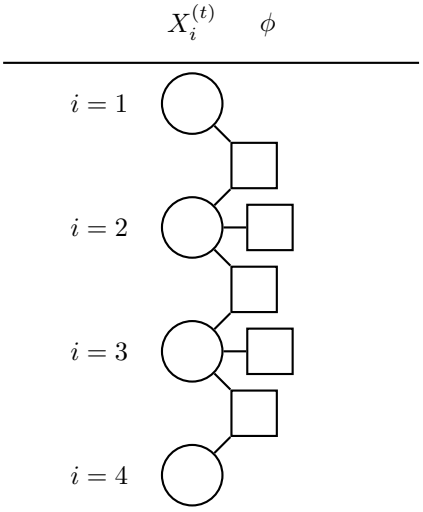

Figure 13: A representation of the induced guiding family featuring in example 4.15. Circles represent variable nodes and squares represent factor nodes. Note that the correlations between all intersecting pairs of immediately adjacent state variables can be maintained with such a factorisation, and note that the additional factors for the two 'middle' nodes account for their over-counting.

4.4 The junction tree algorithm

We have shown that for both simply factorised and fully connected factorisations the M-projection of $\widehat{g}_t$ is given by a product of a number of its marginal distributions, and therefore we now describe how to efficiently compute such terms. Consider the relation for the exact recursion once more:

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right). \tag{84}$$

Note that the involved terms (i.e. the observation model, the transition model, and the approximate guiding term g_{t+1}) can all be represented in some factored form, and thus so can their product. We will exploit this insight to compute all required marginal distributions of $\widehat{g}_t$ with the *junction tree algorithm*, thereby providing an efficient and general approach to computing M-projections.

4.4.1 Introduction

We first consider the junction tree algorithm in general before describing its implementation for both the simply factorised and fully connected cases. Let $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ with $\mathcal{R} = \{r_i,\ i = 1 \dots K\}$ and $\mathcal{E} = \{(r_i, r_{i+1}),\ i = 1 \dots K-1\}$ be a valid junction tree for a suitable factorisation

$$Q(\boldsymbol{X}) = \prod_{i=1}^{N} f_i(\boldsymbol{S}_i). \tag{85}$$

Denote the product of all factors f_j assigned to region r_i with ψ_i and let $\widetilde{\boldsymbol{S}}_{\widetilde{r}_i}$ denote its scope, such that we may alternatively write

$$Q(\boldsymbol{X}) = \prod_{i=1}^{K} \psi_i\left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_i}\right). \tag{86}$$

Let $\boldsymbol{S}_{\widetilde{r}_i}$ denote the set of all variables assigned to region r_i and denote the intersections of variables assigned to pairs of adjacent regions with $\boldsymbol{S}_{\widetilde{s}_i} = \left(\boldsymbol{S}_{\widetilde{r}_i} \cap \boldsymbol{S}_{\widetilde{r}_{i+1}}\right)$. To construct a junction tree which satisfies the running intersection property it may be necessary to assign a variable X_j to a region r_i even if it does not feature in the scope of any factor assigned to that region, and thus $\widetilde{\boldsymbol{S}}_{\widetilde{r}_i} \subseteq \boldsymbol{S}_{\widetilde{r}_i}$.

For each set $\boldsymbol{S}_{\widetilde{r}_i}$ it follows by definition that its marginal distribution is given by

$$Q[\boldsymbol{S}_{\widetilde{r}_i}] = \sum_{\boldsymbol{X} \setminus \boldsymbol{S}_{\widetilde{r}_i}} \prod_{j=1}^{K} \psi_j\left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_j}\right), \tag{87}$$

and by splitting the summation operation into smaller steps and multiplying in each factor ψ_j only when 'required' we can now efficiently compute all marginal distributions of this form by defining a *message-passing algorithm* along the edges of $\mathcal{J}$, referred to as the *junction tree algorithm*.

It is difficult to describe the junction tree algorithm in general due to the heavy notation involved. We will therefore not display the algorithm in 'algorithmic' form but instead discuss the resulting computational procedure for the line graph structures considered in this work through text. A description of the computational data structure used to represent the involved objects is provided in Appendix A.

Algorithm 4.16. Let $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ with $\mathcal{R} = \{r_i,\ i = 1 \dots K\}$ and $\mathcal{E} = \{(r_i, r_{i+1}),\ i = 1 \dots K-1\}$ be a valid junction tree for a general suitable factorisation Q, such that we may write

$$Q(\boldsymbol{X}) = \prod_{i=1}^{K} \psi_i \left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_i} \right). \tag{88}$$

Setting: We seek to efficiently compute the marginal distribution for each set $\boldsymbol{S}_{\widetilde{r}_i}$.

The **ascending sweep** of the junction tree algorithm is initialised from region r_1. The first message to region r_2 is defined as

$$\mu_{r_1 \to r_2} (\boldsymbol{S}_{\widetilde{s}_1}) = \sum_{\boldsymbol{S}_{\widetilde{r}_1} \setminus \boldsymbol{S}_{\widetilde{s}_1}} \psi_1 (\boldsymbol{S}_{\widetilde{r}_1}), \tag{89}$$

and all subsequent ascending sweep messages are defined as

$$\mu_{r_i \to r_{i+1}} (\boldsymbol{S}_{\widetilde{s}_i}) = \sum_{\boldsymbol{S}_{\widetilde{r}_i} \setminus \boldsymbol{S}_{\widetilde{s}_i}} \left(\mu_{r_{i-1} \to r_i} (\boldsymbol{S}_{\widetilde{s}_{i-1}}) \times \psi_i \left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_i} \right) \right). \tag{90}$$

The **descending sweep** of the junction tree algorithm is initialised from region r_K. The first message to region r_{K-1} is defined as

$$\mu_{r_{K-1} \leftarrow r_K} \left(\boldsymbol{S}_{\widetilde{s}_{K-1}} \right) = \sum_{\boldsymbol{S}_{\widetilde{r}_K} \setminus \boldsymbol{S}_{\widetilde{s}_{K-1}}} \psi_K (\boldsymbol{S}_{\widetilde{r}_K}), \tag{91}$$

and all subsequent descending sweep messages are defined as

$$\mu_{r_{i-1} \leftarrow r_i} \left(\boldsymbol{S}_{\widetilde{s}_{i-1}} \right) = \sum_{\boldsymbol{S}_{\widetilde{r}_i} \setminus \boldsymbol{S}_{\widetilde{s}_{i-1}}} \left(\mu_{r_i \leftarrow r_{i+1}} (\boldsymbol{S}_{\widetilde{s}_i}) \times \psi_i \left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_i} \right) \right). \tag{92}$$

Each marginal distribution of the form shown in equation (87) is then given by the product of all 'incoming' messages for that respective region and the 'local' factor ψ_i, i.e.

$$Q[\boldsymbol{S}_{\widetilde{r}_i}] = \mu_{r_{i-1} \to r_i} \left(\boldsymbol{S}_{\widetilde{s}_{i-1}} \right) \times \psi_i \left(\widetilde{\boldsymbol{S}}_{\widetilde{r}_i} \right) \times \mu_{r_i \leftarrow r_{i+1}} (\boldsymbol{S}_{\widetilde{s}_i}), \tag{93}$$

and marginal distributions for any set of variables which is a subset of any set $\boldsymbol{S}_{\widetilde{r}_i}$ can in turn be derived through further marginalisation of the corresponding marginal distribution.

Remark 4.17. The order in which messages are computed can alternatively be interpreted as 'pushing' each factor ψ_j as far into the nested summation operation as possible, aiming to keep the scopes of the terms we operate on (with the sum and product operations) as 'small' as possible and thereby minimising the computational complexity. It follows that

$$\mu_{r_{i-1} \to r_i} \left(\boldsymbol{S}_{\widetilde{s}_{i-1}} \right) = \sum_{\boldsymbol{S}_{\widetilde{r}_1} \setminus \boldsymbol{S}_{\widetilde{s}_1}} \cdots \sum_{\boldsymbol{S}_{\widetilde{r}_{i-1}} \setminus \boldsymbol{S}_{\widetilde{s}_{i-1}}} \prod_{j=1}^{i-1} \psi_j \left(\boldsymbol{S}_{\widetilde{r}_j} \right), \tag{94}$$

$$\mu_{r_i \leftarrow r_{i+1}} \left(\boldsymbol{S}_{\widetilde{s}_{i-1}} \right) = \sum_{\boldsymbol{S}_{\widetilde{r}_{i+1}} \setminus \boldsymbol{S}_{\widetilde{s}_i}} \cdots \sum_{\boldsymbol{S}_{\widetilde{r}_K} \setminus \boldsymbol{S}_{\widetilde{s}_{K-1}}} \prod_{j=i+1}^{K} \psi_j \left(\boldsymbol{S}_{\widetilde{r}_j} \right), \tag{95}$$

from which the result shown in equation (93) follows immediately. We now consider the implementation of the junction tree algorithm for both the simply factorised and fully connected cases.

4.4.2 The simply factorised case

If the guiding function g_{t+1} represents a simple factorisation of the state space then the implementation of the junction tree algorithm is somewhat more straightforward than for the fully connected case: Let $\mathcal{F}_{\mathcal{R}}$ be a simple factorisation of the state space of a process $\mathcal{X}$, such that

$$g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right) = \prod_{i=1}^{K} \widehat{g}_{t+1}\left[\boldsymbol{X}_{r_i}^{(t+1)}\right]. \tag{96}$$

Because the sets of adjacent state variables are all disjoint from one another the summation operation shown in equation (84) factorises as

$$\sum_{\boldsymbol{X}^{(t+1)}} P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right) = \sum_{\boldsymbol{X}_{r_1}^{(t+1)}} \cdots \sum_{\boldsymbol{X}_{r_K}^{(t+1)}} \prod_{i=1}^{K} P\left(\boldsymbol{X}_{r_i}^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right) \widehat{g}_{t+1}\left[\boldsymbol{X}_{r_i}^{(t+1)}\right] \tag{97}$$

$$= \prod_{i=1}^{K} \sum_{\boldsymbol{X}_{r_i}^{(t+1)}} P\left(\boldsymbol{X}_{r_i}^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right) \widehat{g}_{t+1}\left[\boldsymbol{X}_{r_i}^{(t+1)}\right], \tag{98}$$

where $\boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)} = \bigcup_{j \in r_i} \boldsymbol{X}_{\mathrm{ne}(j)}^{(t)}$. It follows that we can first compute the *summation factors*

$$\widetilde{\psi}_i\left(\boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right) = \sum_{\boldsymbol{X}_{r_i}^{(t+1)}} P\left(\boldsymbol{X}_{r_i}^{(t+1)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right) \widehat{g}_{t+1}\left[\boldsymbol{X}_{r_i}^{(t+1)}\right], \tag{99}$$

and subsequently concern ourselves only with with the simplified relation

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \prod_{i=1}^{K} \widetilde{\psi}_i\left(\boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right). \tag{100}$$

Finally, recall that by assumption 2.8 for any set of observed state variables there exists at least one sub-process featuring a neighbourhood which encompasses that set. We can therefore multiply each conditionally independent observation model factor into exactly one compatible $\widetilde{\psi}_i$ factor, such that we may succinctly write

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{K} \psi_i\left(\boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}\right). \tag{101}$$

Due to assumption 4.6 the factors ψ_i and their scopes now define a junction tree given by a line graph which we may exploit to efficiently compute all required marginal terms with the junction tree algorithm described previously.

Definition 4.18. Let $\mathcal{F}_{\mathcal{R}}$ be a simple factorisation of the state space of a process $\mathcal{X}$. It follows from the preceding discussion that the factorisation of $\widehat{g}_t$ can be represented as a junction tree $\mathcal{J} = (\widetilde{\mathcal{R}}, \mathcal{E})$ with $\widetilde{\mathcal{R}} = \{\widetilde{r}_i,\ i = 1 \dots K\}$ and $\mathcal{E} = \{(\widetilde{r}_i, \widetilde{r}_{i+1}),\ i = 1 \dots K-1\}$, where

$$\widetilde{r}_i = \left\{\boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t)}, \psi_i\right\}. \tag{102}$$

We refer to $\mathcal{J}$ as the **simplified junction tree** induced by $\mathcal{F}_{\mathcal{R}}$ for $\widehat{g}_t$.

4.4.3 The fully connected case

We now consider the implementation of the junction tree algorithm if the guiding function g_{t+1} represents a fully connected factorisation of the state space. Recall that the observation model, the transition model, and the guiding term g_{t+1} itself all factorise, and recall that for a fully connected factorisation each state variable may feature in the scope of multiple factors of g_{t+1}.

It follows that now the summation operation shown in equation (84) does not factorise at all. The 'exact recursion' $\widehat{g}_t$ is therefore intractable, and thus we seek to compute its M-projection *without actually computing* the term itself.

The key insight is to note that without the summation operation the product of all involved terms does factorise. Instead of computing $\widehat{g}_t$ and then computing the required marginals from that term we therefore define the *two-time step approximation*

$$\psi\left(\boldsymbol{X}^{(t:t+1)}\right) = P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times P\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) \times g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right), \tag{103}$$

and equivalently compute the required marginals for g_t from ψ, as

$$\widehat{g}_t\left(\boldsymbol{X}^{(t)}\right) = \sum_{\boldsymbol{X}^{(t+1)}} \psi\left(\boldsymbol{X}^{(t:t+1)}\right). \tag{104}$$

Definition 4.19. Let $\mathcal{F}_{\mathcal{R}}$ be a fully connected factorisation of the state space of a process $\mathcal{X}$. The exact recursion $\widehat{g}_t$ is intractable and instead we compute its marginal distributions by constructing a junction tree for the two-time step approximation ψ. We refer to such a junction tree as the **two-time step junction tree** induced by $\mathcal{F}_{\mathcal{R}}$ for $\widehat{g}_t$.

The construction of a valid junction tree for the two-time step approximation depends on (i) the transition model of the process under consideration and (ii) the specific factorisation of g_{t+1}. It is therefore not possible to derive any general construction and instead we consider the implementation of the junction tree algorithm for the fully connected case with an example.[7]

Example 4.20. Consider the transition model of the process described in example 4.2, the guiding term described in example 4.15, and suppose that

$$P\left(\boldsymbol{Y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) = P\left(Y_{v_1}^{(t)} \mid \boldsymbol{X}_{r_1}^{(t)}\right) P\left(Y_{v_2}^{(t)} \mid \boldsymbol{X}_{r_2}^{(t)}\right) P\left(Y_{v_3}^{(t)} \mid \boldsymbol{X}_{r_3}^{(t)}\right). \tag{105}$$

To construct a valid junction tree for the two-time step approximation resulting from the product of these three terms we must 'overlay' all three objects, and to efficiently compute all required marginals we seek to construct regions featuring as few variable nodes as possible. A graphical representation of a valid two-time step junction tree is shown in Figure 14.

[7] To construct a two-time step junction tree we must assign each conditionally independent factor of both the observation model and the transition model and each factor of the guiding term g_{t+1} to exactly one region. We can then pass messages between regions which contain variables from both time steps to compute the marginals for regions, from which the marginals for g_t can in turn be computed. There is no longer any notion of a 'pullback'.

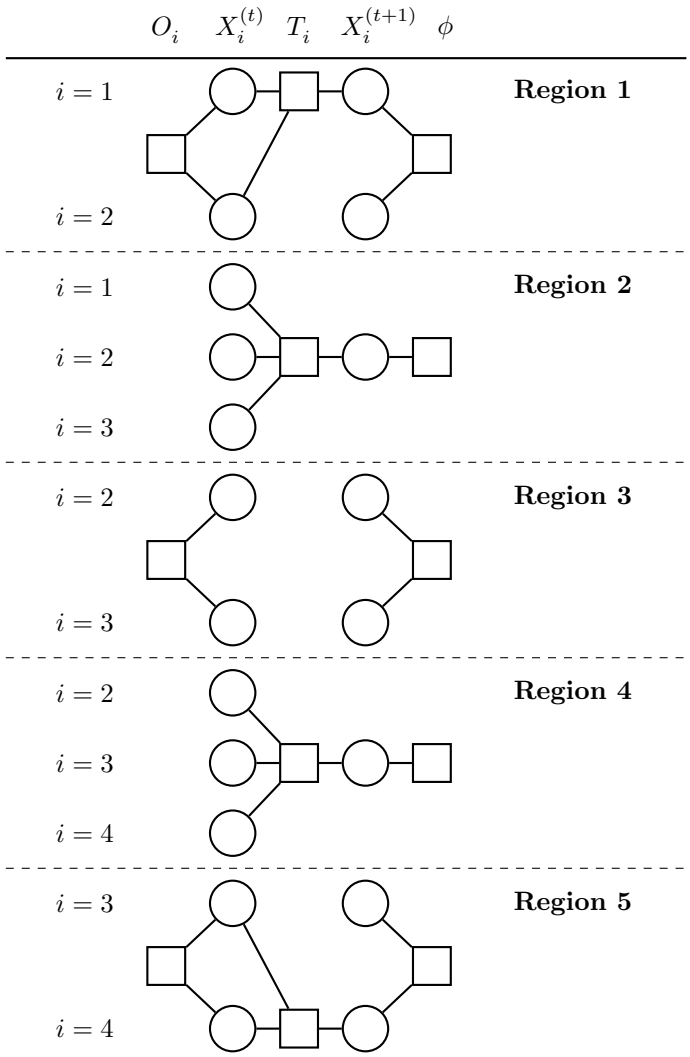

Figure 14: A representation of a valid junction tree for the two-time step approximation featuring in example 4.20. The junction tree consists of five regions. The factor nodes and variable nodes assigned to each region are displayed as (potentially disconnected) factor graphs. Circles represent variables nodes and squares represent factor nodes. Factors of the observation model are denoted with O_i and factors of the transition model are denoted with T_i. Note that although the two-time step approximation consists of eight variable nodes in total, each region consists of only four variable nodes. The junction tree algorithm allows us to efficiently compute marginal distributions for the sets of state variables assigned to each region by passing messages between them, from which the required marginals for g_t can in turn be computed.

4.5 Forward guiding with factorised guiding terms

Backward filtering with expectation propagation allows us to efficiently compute rich approximations of the conditional likelihood terms for all time steps. However, if the chosen factorisation of the state space is not fully factorised then the resulting guiding terms do not allow all state variables to be sampled independently of one another. In this subsection we motivate a number of approaches to dealing with this issue.

4.5.1 Simply factorised guiding terms

Note that if simply factorised guiding terms are used directly in the forward guiding phase that the resulting guided probability distribution for a time step does partially factorise, i.e.

$$P_{\boldsymbol{\theta}}^{\circ}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}{\sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)} \tag{106}$$

$$= \frac{\prod_{i=1}^{K} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}_{r_i}^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t-1)} = \boldsymbol{x}_{\mathrm{ne}(r_i)}^{(t-1)}\right) \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right]}{\prod_{i=1}^{K} \sum_{\boldsymbol{X}_{r_i}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}_{r_i}^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t-1)} = \boldsymbol{x}_{\mathrm{ne}(r_i)}^{(t-1)}\right) \widehat{g}_t\left[\boldsymbol{X}_{r_i}^{(t)}\right]} \tag{107}$$

$$= \prod_{i=1}^{K} P_{\boldsymbol{\theta}}^{\circ}\left(\boldsymbol{X}_{r_i}^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(r_i)}^{(t-1)} = \boldsymbol{x}_{\mathrm{ne}(r_i)}^{(t-1)}\right). \tag{108}$$

The sets of adjacent state variables for which we maintain the correlation must be sampled jointly, but disjoint sets can be sampled independently of one another. To further decrease the computational complexity of the forward guiding phase such guiding terms can also be *projected* onto an even 'simpler' factorised guiding family, the simplest of all again being the fully factorised case.

Recall from Figure 8 that backward filtering with expectation propagation can be seen as a series of *recurse*- and *project*-steps. If the resulting guiding terms g are additionally projected onto a simpler guiding family then the computation of the resulting guiding terms $\widetilde{g}$ can be represented similarly, with the inclusion of a second projection step from g_t to $\widetilde{g}_t$ for all time steps. Note that 'projecting' a guiding term onto a simpler factorised guiding family amounts to computing the required marginals of the projected guiding terms from corresponding factors of the original.

4.5.2 Fully connected guiding terms

In the fully connected case the intersections of scopes of factors of g_t form one long 'chain' over the entire state space, and therefore the resulting guided probability distribution for a time step does not factorise at all. To deal with this issue fully connected guiding terms may again be projected onto a simply factorised guiding family, or alternatively such guiding terms can be used directly through *dynamic guiding*, which we describe here.

Let $\mathcal{F}_{\mathcal{R}}$ be a fully connected factorisation of the state space. We wish to sample from the guided probability distribution

$$P_{\boldsymbol{\theta}}^{\circ}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) = \frac{P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right)}{Z_{\boldsymbol{\theta}}^{(t)}\left(\boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right)}, \tag{109}$$

where $Z_{\boldsymbol{\theta}}^{(t)}$ may be considered a normalising constant, i.e.

$$Z_{\boldsymbol{\theta}}^{(t)}\left(\boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) = \sum_{\boldsymbol{X}^{(t)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right). \tag{110}$$

It is not possible to sample the entire latent state $\boldsymbol{X}^{(t)}$ at once as the state space is intractably large, and instead we therefore sample (sets of) state variables sequentially by exploiting conditional independence between the variables of the guided probability distribution. For ease of exposition we consider the dynamic guiding procedure through an example.

Example 4.21. Consider a process $\mathcal{X}$ which consists of N sub-process. Each sub-process $\mathcal{X}_i$ interacts with its immediate neighbours $\mathcal{X}_{i-1}$ and $\mathcal{X}_{i+1}$, with suitable restrictions for the 'edge' sub-processes $\mathcal{X}_1$ and $\mathcal{X}_N$. We consider a fully connected guiding function g_t which consists of factors for pairs of immediately adjacent state variables, such that we may write

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \frac{\prod_{i=1}^{N-1} \phi\left(\boldsymbol{X}_{i:i+1}^{(t)}\right)}{\prod_{j=2}^{N-1} \phi\left(X_j^{(t)}\right)}. \tag{111}$$

Recall that the factors ϕ in the preceding display correspond to marginal distributions of $\widehat{g}_t$ for their respective scopes. We may therefore alternatively write

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \phi\left(\boldsymbol{X}_{1:2}^{(t)}\right) \prod_{i=3}^{N} \phi\left(X_i^{(t)} \mid X_{i-1}^{(t)}\right), \tag{112}$$

and we refer to the preceding as the *conditional chain* form of the guiding term g_t. As shorthand we additionally introduce the notation

$$f_i\left(X_i^{(t)}\right) = P_{\boldsymbol{\theta}}\left(X_i^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)} = \boldsymbol{x}_{\mathrm{ne}(i)}^{(t-1)}\right), \tag{113}$$

such that we may succinctly write

$$P_{\boldsymbol{\theta}}^{\circ}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) \propto P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right) \tag{114}$$

$$= \left(\phi\left(\boldsymbol{X}_{1:2}^{(t)}\right) \prod_{i=3}^{N} \phi\left(X_i^{(t)} \mid X_{i-1}^{(t)}\right)\right) \times \left(\prod_{i=1}^{N} f_i\left(X_i^{(t)}\right)\right). \tag{115}$$

The task of sampling from the preceding distribution can be achieved through backward filtering forward guiding *within the time step*: The conditional chain form of g_t acts as a forward model and the conditionally independent transition model factors f_i appear as observation model terms.

With the familiar backward filtering process described in theorem 3.2 we can therefore first compute the required h-transforms and subsequently sample from the 'initial state' and 'guided distributions'

$$\phi^{\star}\left(\boldsymbol{X}_{1:2}^{(t)}\right) \propto \phi\left(\boldsymbol{X}_{1:2}^{(t)}\right) f_1\left(X_1^{(t)}\right) h_2\left(X_2^{(t)}\right), \tag{116}$$

$$\phi^{\star}\left(X_i^{(t)} \mid X_{i-1}^{(t)}\right) \propto \phi\left(X_i^{(t)} \mid X_{i-1}^{(t)}\right) h_i\left(X_i^{(t)}\right), \tag{117}$$

resulting in correctly distributed samples. Finally, recall that the normalising term $Z_{\boldsymbol{\theta}}^{(t)}$ is required for the computation of the likelihood ratio described in theorem 3.5, and note that

$$Z_{\boldsymbol{\theta}}^{(t)}\left(\boldsymbol{X}^{(t-1)} = \boldsymbol{x}^{(t-1)}\right) = \sum_{\boldsymbol{X}_{1:2}^{(t)}} \phi\left(\boldsymbol{X}_{1:2}^{(t)}\right) f_1\left(X_1^{(t)}\right) h_2\left(X_2^{(t)}\right). \tag{118}$$

Remark 4.22. Dynamic guiding with more complex guiding terms is generally similar, though depending on the specific factorisation used for g_t the resulting conditional chain form may be higher-order Markov, may necessitate sampling certain sets of state variables jointly, or both.

4.6 Discussion

We discuss a number of aspects related to backward filtering procedures with expectation propagation in a broader sense. First, note that many of the assumptions we have introduced in section 2 and section 4 are not essential for the feasibility of the methods presented in this work. Throughout this section we have assumed that both the transition model and the observation model are time-invariant and that the neighbourhood structure of sub-processes is such that the transition model can be represented as a junction tree given by a line graph. In actuality neither the transition model nor the observation model need necessarily be time-invariant, and additionally the neighbourhood structure can be much more general: If sub-processes interact with varying sets of other sub-processes throughout time then we may simply introduce a different factorisation of the state space for each time step, such that we can adequately represent the strongest correlation between sets of interacting sub-processes for all time steps. If the resulting neighbourhood structures do not allow for the transition model to be represented as a junction tree given by a line graph then we may employ a more general variant of the junction tree algorithm which operates on tree-structures, rather than the line graph variant described in this section.

What we are actually interested in is efficiently computing 'good' approximations of the conditional likelihood of all later observations for all time steps, and to achieve this we summarise such functions in terms of the most relevant correlations between the state variables. The factorised representation of guiding terms essentially forms an 'information bottleneck' by which the procedure remains tractable, and with expectation propagation we then compute the optimal guiding terms for all time steps through moment matching. The backward filtering procedure to compute the optimal guiding terms for all time steps must then necessarily be a recursive procedure, as by definition it is not possible to construct a better approximation of the conditional likelihood for any time step by incorporating more guiding terms than just the 'next' in its computation without exceeding the computational resources required for the computation of that guiding term.[8]

As per Boyen & Koller (1998), the size of factors used in the approximation of g is necessarily a trade-off between accuracy and computational efficiency: Depending on the problem under consideration we may wish to construct larger sets for certain groups of sub-processes and allow a more granular representation for others. As part of a inference problem we can therefore vary many different elements of the backward filtering forward guiding method: Given a fixed amount of computational resources we can choose to either re-compute a less accurate set of guiding terms (given a new estimate of the model parameters) more often or compute a more accurate set of guiding terms less frequently. Such options provide us with more tools to solving complex problems.

Upon first impression it may seem that dynamic guiding is excessively computationally demanding. However, note that throughout the backward filtering and forward guiding procedures 'within a time step' that the largest factors encountered only take two adjacent state variables as their scopes. It follows that the computational complexity of dynamic guiding with factors of *intersecting* pairs of adjacent state variables scales identically to forward guiding with factors of *disjoint* pairs, though with a larger 'constant' C. In many settings this may be favourable, especially if the states of latent sub-processes are highly correlated.

Finally, note that use of the canonical KL-divergence is defining for the expectation propagation procedure specifically, but that we may define other backward filtering procedures by computing guiding functions through optimisation with respect to alternative divergences. This is explored further in Appendix C and may be an avenue for future research.

[8] To be explicit: If g_{t+1} is the optimal function q in $\mathcal{Q}$ for time step $t+1$, then incorporating g_{t+2} in the computation of g_t is redundant.

5 Numerical assessment

In this section we compare a number of backward filtering procedures through a numerical investigation. As an example we consider the problem described in section 2. We first briefly describe the backward filtering procedures considered throughout this section and then present the results.

5.1 Fully factorised guiding terms

We first compare expectation propagation with fully factorised guiding terms with the two approaches presented by Van der Meulen & Schauer (2022). The expectation propagation approach has been described in section 4 and we summarise the two other approaches here.

5.1.1 Backward diagonalisation

In the original consideration of the SIR-process Van der Meulen & Schauer (2020) provide a very rough approximate backward filtering approach, referred to as *backward diagonalisation*. Recall from section 3 that the backward filtering forward guiding method allows samples from a guided process $\mathcal{X}^\circ$ to be up- or down-weighted, resulting in correctly distributed samples from the exact conditioned process $\mathcal{X}^\star$.

The backward diagonalisation approach is based on the insight that if the interaction between individuals is ignored throughout the backward filtering procedure that the resulting guiding terms may still induce a sufficiently accurate guided process. Backward diagonalisation therefore consists of replacing the actual transition model with a secondary transition model

$$\widetilde{P}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) = \prod_{i=1}^{N} \widetilde{P}\left(X_i^{(t)} \mid X_i^{(t-1}\right), \tag{119}$$

and computing the set of guiding terms with it, thereby completely ignoring the interaction between individuals. Recall the transition matrices of the N interacting discrete-time Markov chains:

$$K_i\left(\boldsymbol{x}^{(t)}\right) = \begin{bmatrix} \psi\left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & 1-\psi\left(\lambda_0 + \lambda N_i\left(\boldsymbol{x}^{(t)}\right)\right) & 0 \\ 0 & \psi(\mu) & 1-\psi(\mu) \\ 1-\psi(\nu) & 0 & \psi(\nu) \end{bmatrix}, \tag{120}$$

where the functions N_i return the number of infected neighbours of individual i. Note that an individual only interacts with its neighbours when in the **S**usceptible state. It is therefore emphatically a 'good' idea to identify factors of the secondary transition model with transition matrices

$$K_i^t = \begin{bmatrix} \psi\left(\lambda_0 + \lambda \widetilde{N}_i^t\right) & 1-\psi\left(\lambda_0 + \lambda \widetilde{N}_i^t\right) & 0 \\ 0 & \psi(\mu) & 1-\psi(\mu) \\ 1-\psi(\nu) & 0 & \psi(\nu) \end{bmatrix}, \tag{121}$$

where the parameters $\widetilde{N}_i^t$ provide an estimate of the number of infected neighbours of individual i at time step t. An advantage of this approach is that the resulting backward filtering procedure is very computationally efficient, as it is not necessary to project the resulting 'exact recursion' onto a factorised guiding family as the recursion does not induce any new correlations, i.e.

$$\sum_{\boldsymbol{X}^{(t)}} \widetilde{P}\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} \sum_{X_i^{(t)}} \widetilde{P}\left(X_i^{(t)} \mid X_i^{(t-1)}\right) \phi_i\left(X_i^{(t)}\right) \tag{122}$$

$$= \prod_{i=1}^{N} \psi_i\left(X_i^{t-1}\right). \tag{123}$$

Note that this is functionally equivalent to backward filtering for N independent Markov chains.

5.1.2 Backward marginalisation

Van der Meulen & Schauer (2022) provide another more general approach to backward filtering, referred to as *backward marginalisation*. Like expectation propagation, the backward marginalisation approach does not replace the transition model with a secondary (simpler) transition model, but rather operates with the actual transition model of the process $\mathcal{X}$.

As originally presented backward marginalisation operates at the level of individual random variables featuring in a graphical model. We will therefore 'emulate' backward marginalisation from the state space model perspective considered in this work, so that similarities and differences between the approaches are more readily apparent.

Similar to expectation propagation, with backward marginalisation the exact recursion is first computed as:

$$\sum_{\boldsymbol{X}^{(t)}} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{N} P\left(X_i^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}\right) \phi_i\left(X_i^{(t)}\right) \tag{124}$$

$$= \prod_{i=1}^{N} \psi_i\left(\boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}\right). \tag{125}$$

Instead of computing an M-projection directly from the product of all preceding factors (as with expectation propagation), backward marginalisation then consists of computing for each factor ψ_i a marginal for each respective state variable $X_u^{(t-1)} \subset \boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}$, i.e.

$$\mu_u^i\left(X_u^{(t-1)}\right) = \sum_{\boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)} \setminus X_u^{(t-1)}} \psi_i\left(\boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}\right). \tag{126}$$

We refer to such a function as a *message* from sub-process $\mathcal{X}_i$ to sub-process $\mathcal{X}_u$. Backward marginalisation finally consists of computing for each sub-process in the preceding time step the product of all 'incoming' messages, i.e.

$$\phi_i\left(X_i^{(t-1)}\right) = \prod_{j \in \inf(i)} \mu_i^j\left(X_i^{(t-1)}\right), \tag{127}$$

where $\inf(i) = \{j : i \in \mathrm{ne}(j)\}$, i.e. the set of indices of sub-processes of which $\mathcal{X}_i$ is part of the neighbourhood.

To be explicit: expectation propagation consists of computing marginal distributions from a product of all factors, whereas backward marginalisation consists of computing a product of all 'relevant' marginal distributions of individual factors. A detailed comparison of the exact computations involved for the backward marginalisation and fully factorised expectation propagation approaches is provided in Appendix D.

It is interesting to note that the backward marginalisation procedure is functionally equivalent to a single iteration of loopy belief propagation. Koller & Friedman (2009) (p. 391) provide numerous interesting generalisations of loopy belief propagation, and comparing the backward marginalisation procedure with these generalisations is a potential avenue for future research.

5.1.3 Problem description

Throughout this section we consider as an example the setting originally introduced by Van der Meulen & Schauer (2020), which we have previously described in section 2. In their consideration of the problem the state of the entire population of 100 individuals is observed without error every 50 time steps, thereby splitting a total of 500 time steps into 10 segments of 50 time steps each. Throughout our investigation we have noted that given the time discretisation of $\tau = 0.1$ used in the original work that the guiding terms generally 'die out' within approximately 50 time steps.

Consider as an example the graphical representation of the values of the factor ϕ_i particular to each time step t, i.e. the values with which $\mathcal{X}_i$ is guided. We note that individual i has been observed to be in the **R**ecovered state at time step 50. The guiding terms are then recursively computed for all preceding time steps (for which there are no observations), and we note that the values of the factor ϕ_i seemingly converge.

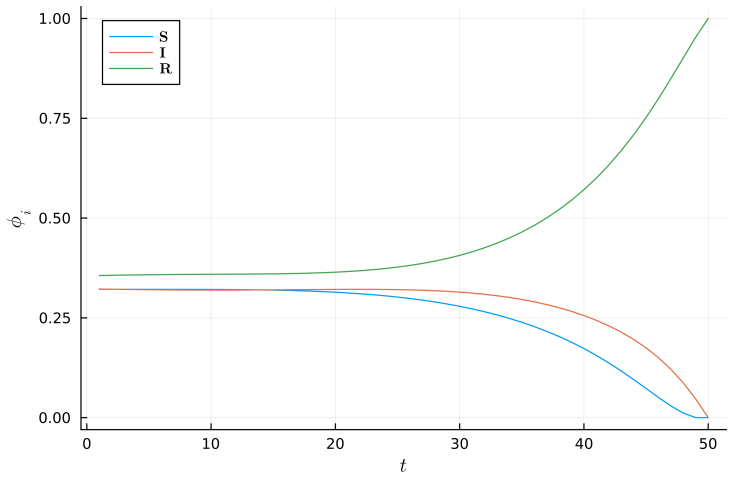

Figure 15: An example of the values of the factor ϕ_i particular to each time step t.

Recall that in each time step we compute a *normalised* guiding term g_t. As we continue recursively backward filtering the exact likelihood rapidly decays, and the factors of guiding terms therefore converge to normalised uniform functions, i.e. with values $1/3$. In this section we therefore also consider only segments of 50 time steps, as past this point the guiding terms have little to no effect.

We compare the performance of the three approaches to backward filtering for three different observation settings of the same realisation. We seek to infer the model parameters $\boldsymbol{\theta} = (\lambda, \mu, \nu)$ and assume that the base infection intensity $\lambda_0 = 0.001$ is known.

As per Van der Meulen & Schauer (2020), we construct a Markov chain to iteratively update the model parameters $\boldsymbol{\theta}$ and latent states of the process.[9] A derivation of the steps involved is given in Appendix E. In contrast to the original work we update all model parameters $\boldsymbol{\theta}$ jointly with a single random walk Metropolis-Hastings step. We update latent states with a preconditioned Crank–Nicolson algorithm "by turning [sample]-simulation into a normalising flow."[10]

[9] The code is provided online.

[10] The construction of such a normalising flow is outside of the scope of this work and for additional discussion we refer the reader to the original work provided by Papaspiliopoulos et al. (2003).

5.1.4 Results

Consider the realisation of an SIR process with time discretisation $\tau = 0.1$ shown in Figure 16. Note the spontaneous emergence of infection which appears at around time step 20 and subsequently spreads throughout a large part of the population. We seek to infer the model parameters $\boldsymbol{\theta}$ given only a limited set of observations of states of individuals at time step 50.

Our intuition is that the comparative advantage that the backward marginalisation and expectation propagation approaches have over backward diagonalisation is that throughout the backward filtering procedure the sub-processes may 'share' information amongst one another. By gradually decreasing the number of individuals observed at time step 50 we therefore expect these two approaches to eventually compare favourably to the backward diagonalisation approach.

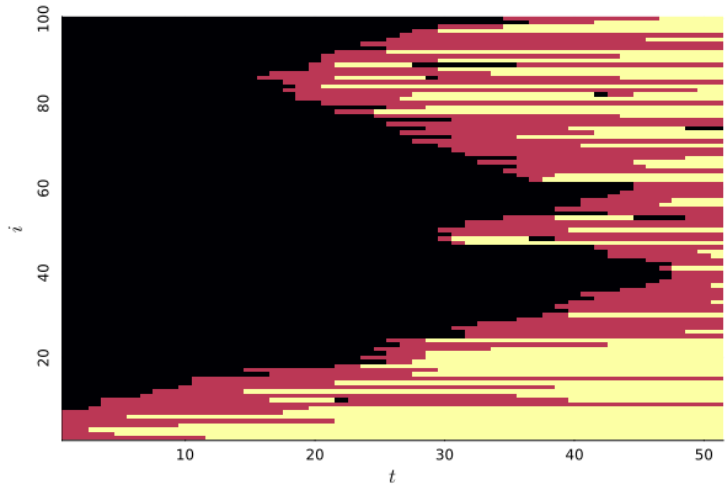

Figure 16: A realisation of a discrete-time SIR process of 100 individuals and 50 time steps. **S**usceptible, **I**nfected and **R**ecovered individuals are respectively denoted as black, red and yellow. Note the spontaneous emergence of infection at certain points because of the 'base' intensity λ_0.

To assess this hypothesis we introduce the following three observation settings:

- **Setting 1:** We directly observe the state of every other individual at time step 50.
- **Setting 2:** We directly observe the state of every third individual at time step 50.
- **Setting 3:** We directly observe the state of every fifth individual at time step 50.

Trace plots of the three model parameters for 30000 updates of $\boldsymbol{\theta}$ are shown for the three approaches and three observation settings in Figure 17, Figure 18, Figure 19, Figure 20, Figure 21, Figure 22, Figure 23, Figure 24 and Figure 25. We observe that in each setting there is little discernible difference in the quality of inference between the three approaches, and may conclude that for this specific problem setting the three approaches are equally effective.

Investigations for similar problem settings have led us to conclude that the quality of inference in most settings consisting of 'simple' observations of individuals is generally similar between the three approaches. Such settings include (among others) observation with error and partial observation of individuals, i.e. measuring their states to be either **S** or {**I** or **R**}. This result speaks to the versatility of the method itself but also to the comparatively 'simple' nature of observing only singular individuals. In the following subsection we therefore consider a more difficult setting which we can not expect the fully factorised guiding terms considered here to handle well.

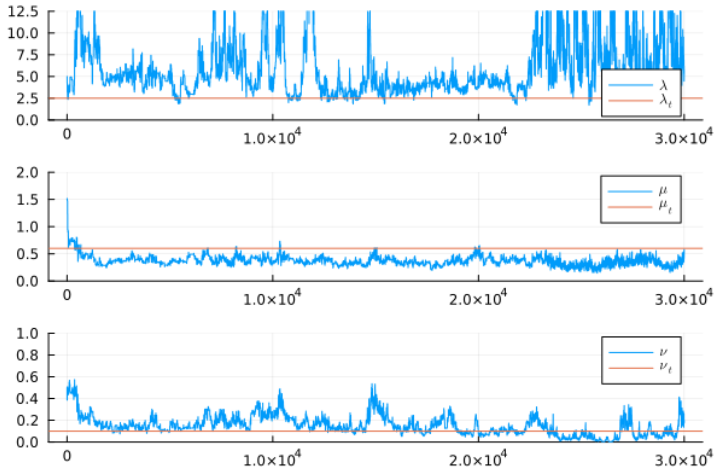

Figure 17: Trace plots for the backward diagonalisation approach for observation setting 1.

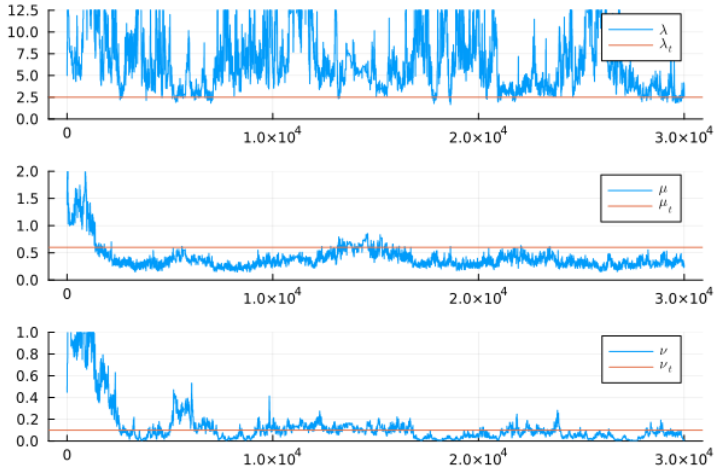

Figure 18: Trace plots for the backward marginalisation approach for observation setting 1.

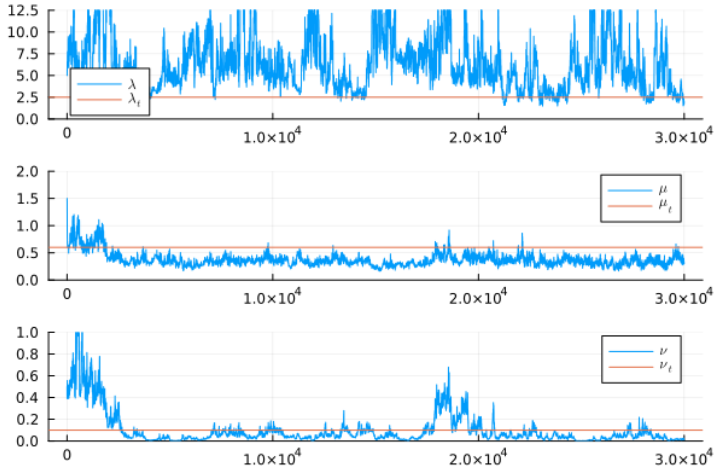

Figure 19: Trace plots for the expectation propagation approach for observation setting 1.

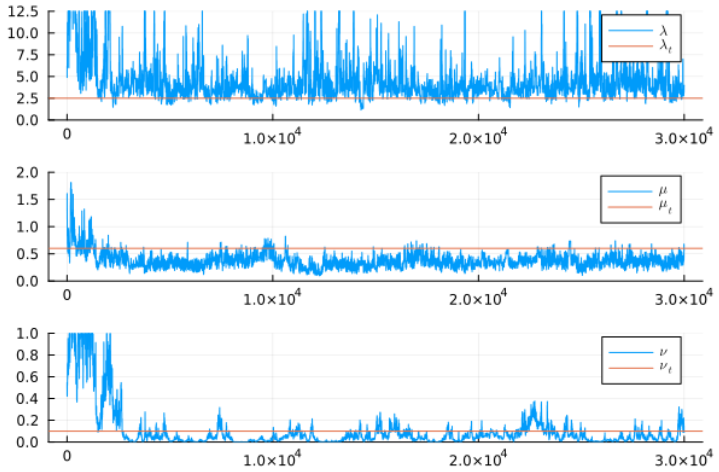

Figure 20: Trace plots for the backward diagonalisation approach for observation setting 2.

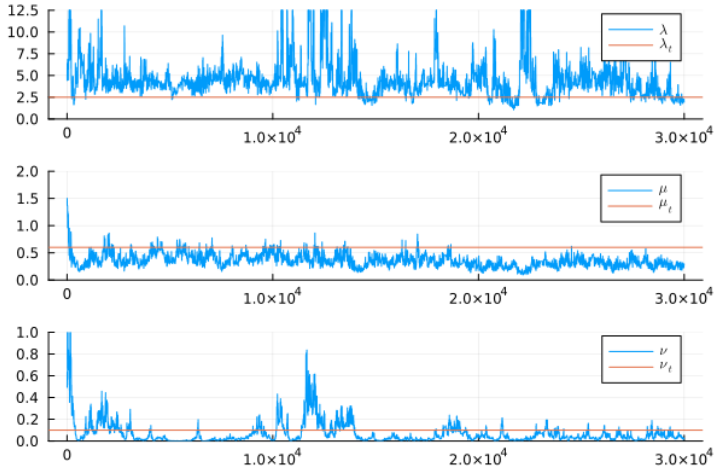

Figure 21: Trace plots for the backward marginalisation approach for observation setting 2.

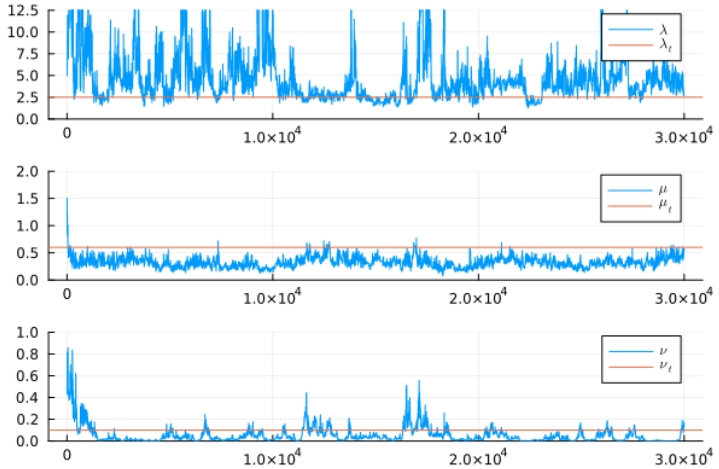

Figure 22: Trace plots for the expectation propagation approach for observation setting 2.

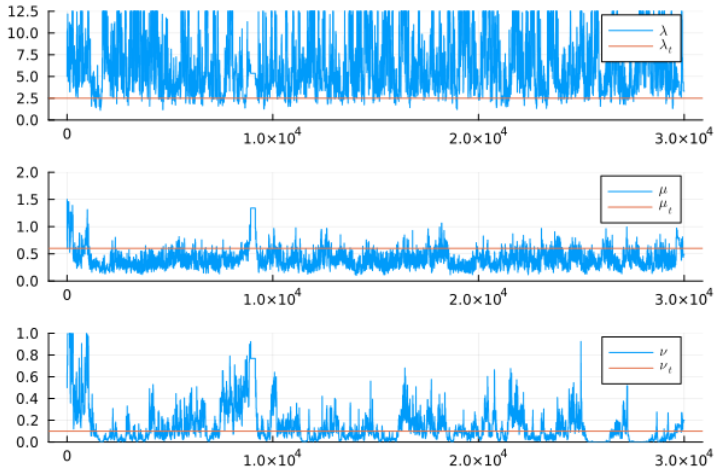

Figure 23: Trace plots for the backward diagonalisation approach for observation setting 3.

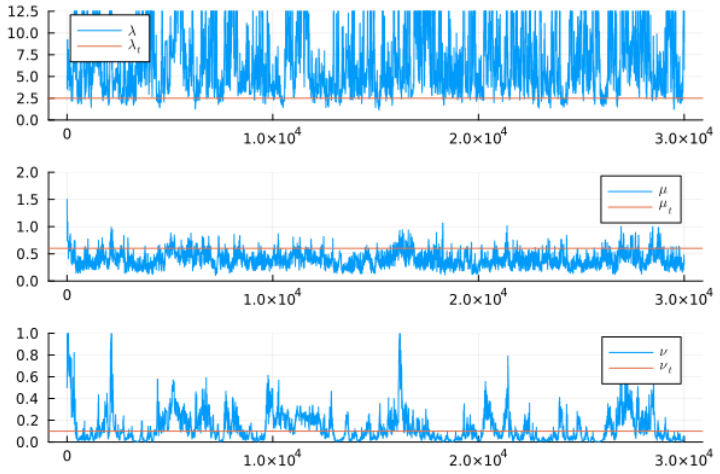

Figure 24: Trace plots for the backward marginalisation approach for observation setting 3.

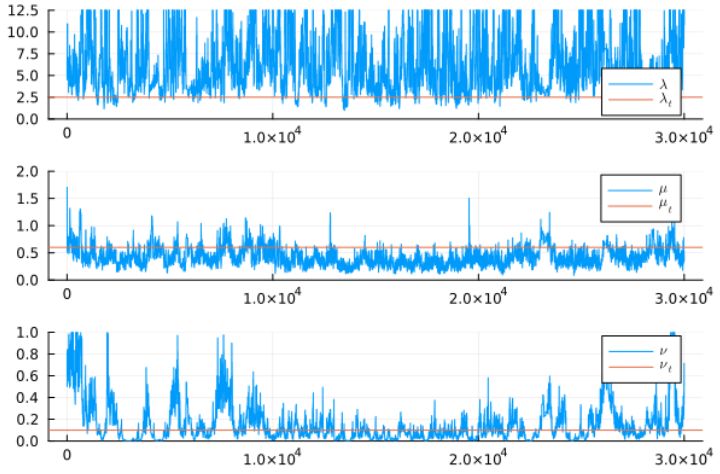

Figure 25: Trace plots for the expectation propagation approach for observation setting 3.

5.2 Fully connected guiding terms

In the previous subsection we have shown that the three approaches to backward filtering with fully factorised guiding terms display comparable performance. In this subsection we increase the complexity of the SIR problem setting and show that we can still derive adequate (or even better) results by computing fully connected guiding terms and employing dynamic guiding.

5.2.1 Problem description

Consider the realisation of an SIR process with time discretisation $\tau = 0.1$ shown in Figure 26. Similar to before we seek to infer the model parameters $\boldsymbol{\theta} = (\lambda, \mu, \nu)$ and reconstruct the realisation given only a limited set of observations, again only at time step 50. We assume that the 'base' infection intensity $\lambda_0 = 0.001$ is known. We do not measure the state of singular individuals, but rather measure states of pairs of adjacent individuals *jointly*.

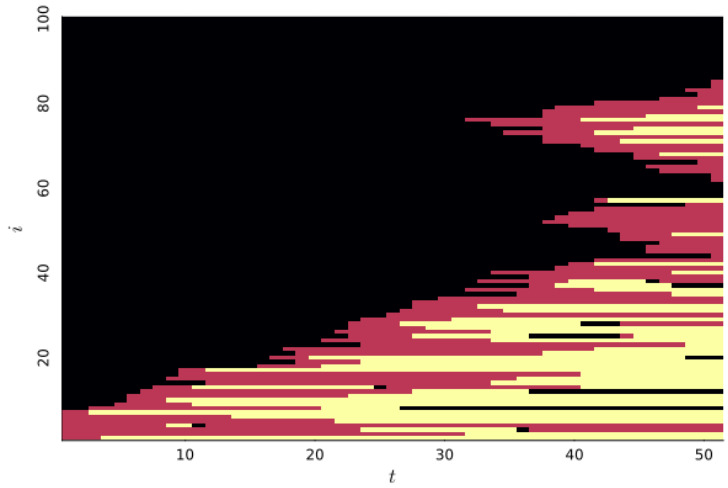

Figure 26: A realisation of a discrete-time SIR process of 100 individuals and 50 time steps. **S**usceptible, **I**nfected and **R**ecovered individuals are respectively denoted as black, red and yellow. Note the spontaneous emergence of infection at certain points because of the 'base' intensity λ_0.

Suppose that at time step 50 we 'test' (i.e. observe) all pairs of adjacent individuals for their states, but that we 'mix up' the results: For each pair of individuals $(i, i+1)$ we measure their states to be $\{\mathbf{SS}\}$, $\{\mathbf{II}\}$, $\{\mathbf{RR}\}$, $\{\mathbf{SI}, \mathbf{IS}\}$, $\{\mathbf{SR}, \mathbf{RS}\}$, or $\{\mathbf{IR}, \mathbf{RI}\}$. If both individuals are in the same state then nothing changes (compared to before), but if individuals are in different states then we can not ascribe a single result to each individual.

With fully factorised guiding terms it is intractable to perform inference in this setting, as each joint observation would have to be 'split' such that all pairs of state variables can be sampled conditionally independent of one another. In the forward guiding phase we then sample pairs of states which do not agree with the joint observations, resulting in very many zero probability samples.

In contrast, fully connected guiding terms consisting of factors of pairs of adjacent state variables (as described in example 4.15) allow us to fully represent such pairwise observations at time step 50 and additionally allow us to approximate the correlations between adjacent individuals in earlier time steps through expectation propagation. We can then construct a guided process with the resulting guiding terms and sample from it with dynamic guiding, as described in example 4.21.

5.2.2 Results

Trace plots of the three model parameters for 30000 updates of $\boldsymbol{\theta}$ are shown in Figure 27. The Markov chain has been fine-tuned to target an acceptance rate of 0.234. We observe that the Markov chain quickly diverges from its starting estimate $\boldsymbol{\theta}_1 = (\lambda = 0.5, \mu = 1.5, \nu = 0.5)$ and explores the posterior very well. For computational efficiency we recompute the set of guiding terms every 100 updates, and by inspection we note that this is justified.

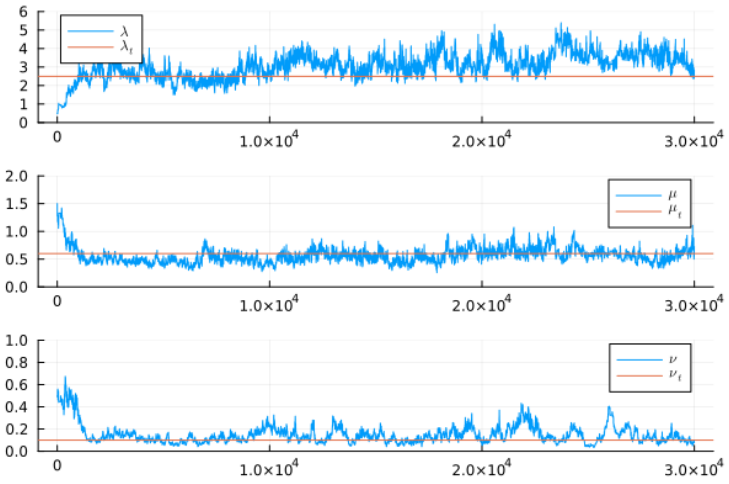

Figure 27: Trace plots of the three model parameters for 30000 updates of $\boldsymbol{\theta}$. The true values of the model parameters $\widehat{\boldsymbol{\theta}} = (\lambda = 2.5, \mu = 0.6, \nu = 0.1)$ of the realisation shown in Figure 26 are denoted by orange horizontal lines.

The initial iterate of the latent states, the final iterate of the latent states and a superimposition of all iterates of latent states attained by the Markov chain are shown in Figure 28, Figure 29, and Figure 30, respectively. It is interesting to note that in the initial iterate, the spontaneous emergence of infection appearing at around time step 30 in the true realisation is already present.

Many of the individuals in the range $\{60, \ldots, 90\}$ are observed to be either **I**nfected or **R**ecovered at time step 50, and because we can represent pairwise correlations throughout the backward filtering procedure even 'bad' (i.e. early) samples of the guided process manage to capture this effect. For later iterations the spontaneous emergence and subsequent spread of infection is modelled even better, as the Markov chain settles in a region of high likelihood samples.

We additionally highlight the increased performance in estimating the model parameter λ compared to the fully factorised approach, which is precisely the parameter particular to the transition to the infected state. Van der Meulen & Schauer (2020) note that in their implementation "only parameter λ is not reconstructed to the value used in the forward simulated model. This is not surprising, as the rate of transitioning from susceptible to infected depends on the product of λ and the number of infected neighbours."

With fully connected guiding terms we are able to partially maintain the correlation between individuals, and therefore it is easier for the method to approximate the number of infected neighbours for each individual at each time step. We may expect that if the interaction between sub-processes were even stronger that the comparative advantage of fully connected guiding terms may be even greater, as fully factorised guiding terms struggle to correctly reconstruct more model parameters.

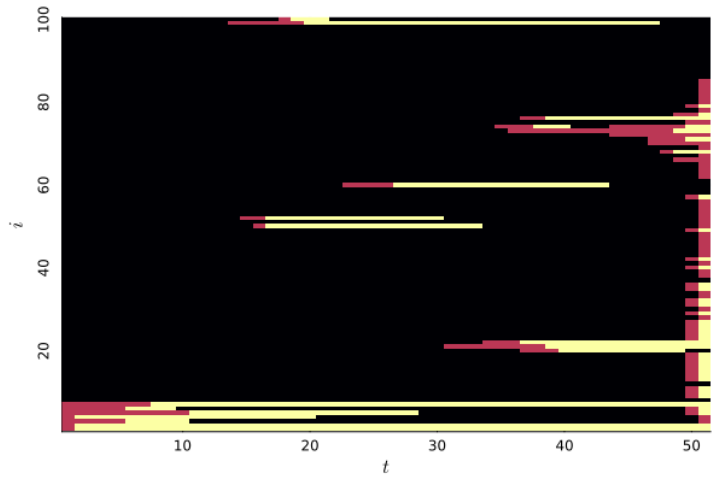

Figure 28: The initial iterate of the latent states.

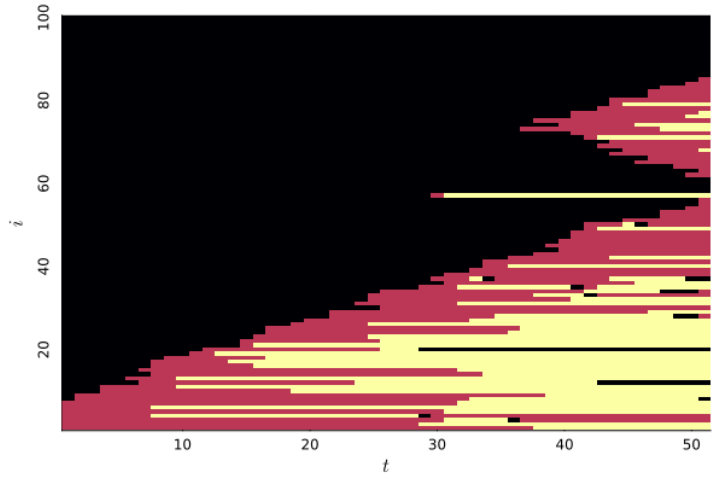

Figure 29: The final iterate of the latent states.

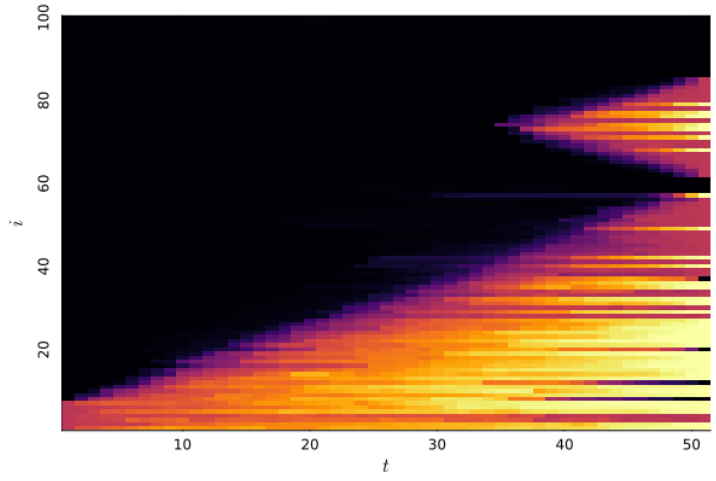

Figure 30: A superimposition of all iterates of latent states attained by the Markov chain. Colours on the gradients 'between' the familiar black, red and yellow colours signify state variables which have attained different states at different iterations of the Markov chain.

6 Conclusion

In this thesis I have described an expectation propagation approach to backward filtering for finite-state space models consisting of a large number of conditionally independent state variables. The approach provides an elegant extension of the backward filtering forward guiding method to such models by considering both the backward filtering phase and the forward guiding phase from the perspective of entire time steps.

Specifically, by exploiting the natural factorisation of the state space via the set of state variables we can construct very expressive guiding terms. By additionally allowing such guiding terms to alter the underlying graphical structure of the model in the forward guiding phase we can sample highly correlated state variables with only a small increase in computational complexity. We have noted that allowing the likelihood to alter the underlying graphical structure of the model in an informed manner proves to be useful, as it allows us to construct a guided process featuring a distribution which better approximates that of the exact conditional process. Though not fully in the spirit of 'automatic' backward filtering forward guiding, taking this idea into consideration for more general (non-sequential) models may be an avenue for future research.

The results show an expectation propagation approach with fully factorised guiding terms to perform as well as existing methods in the SIR-process example setting if we allow only 'simple' observations of individual state variables, which speaks to the versatility of the backward filtering forward guiding method itself. However, we may expect that for processes featuring much 'stronger' interaction between sub-processes that a more holistic approach to approximating conditional likelihood terms may prove prudent, i.e. with methods such as expectation propagation.

By introducing more complex 'joint' observations we have essentially emulated this setting, in which case the fully connected guiding terms have proven to perform very well. We conclude that an advanced treatment of the backward filtering phase may be warranted (or even necessary) in more complex settings, and the fully connected guiding terms introduced in this work therefore add to the capabilities of the backward filtering forward guiding method as a whole.

More generally it may prove fruitful to seek out more connections between the backward filtering forward guiding method and existing approximate likelihood filtering procedures. The functional equivalence between backward marginalisation and (a single iteration of) loopy belief propagation is of particular interest, as much research has been done to predict when the method performs well and when it does not. Any failure mode of loopy belief propagation is shared with backward filtering with backward marginalisation, and thus care should be taken to account for these cases.

Finally, because of the large amount of literature available regarding expectation propagation methods for other types of models we may reasonably expect the methods presented in this work to generalise more readily compared to the familiar backward marginalisation approach. Perhaps the main value of this thesis is therefore as a first effort towards such a generalisation, and I hope to see the expectation propagation approach introduced in this work find application elsewhere.

References

[1] Frank van der Meulen and Moritz Schauer. *Automatic Backward Filtering Forward Guiding for Markov processes and graphical models*. 2020. arXiv: 2010.03509v1 [stat.CO].

[2] Frank van der Meulen and Moritz Schauer. *Automatic Backward Filtering Forward Guiding for Markov processes and graphical models*. 2022. arXiv: 2010.03509v3 [stat.CO].

[3] Frank van der Meulen. *Introduction to Automatic Backward Filtering Forward Guiding*. 2022. arXiv: 2203.04155 [stat.CO].

[4] Thomas P. Minka. *Expectation Propagation for approximate Bayesian inference*. 2013. arXiv: 1301.2294 [cs.AI].

[5] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive computation and machine learning. MIT Press, 2009. ISBN: 9780262013192. URL: https://books.google.co.in/books?id=7dzpHCHzNQ4C.

[6] Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002.

[7] Xavier Boyen and Daphne Koller. *Tractable Inference for Complex Stochastic Processes*. 2013. arXiv: 1301.7362 [cs.AI].

[8] Omiros Papaspiliopoulos and Gareth Roberts. "Non-Centered Parameterisations for Hierarchical Models and Data Augmentation". In: *Bayesian Statistics* 7 (Jan. 2003), pp. 307–326.

[9] Tom Heskes and Onno Zoeter. *Expectation Propogation for approximate inference in dynamic Bayesian networks*. 2012. arXiv: 1301.0572 [cs.AI].

[10] Thomas Minka. *Divergence measures and message passing*. Tech. rep. 2005. URL: https://www.seas.harvard.edu/courses/cs281/papers/minka-divergence.pdf.

[11] José Miguel Hernández-Lobato et al. *Black-box α-divergence Minimization*. 2016. arXiv: 1511.03243 [stat.ML].

[12] Xavier Boyen and Daphne Koller. "Exploiting the Architecture of Dynamic Systems". In: *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*. AAAI '99/IAAI '99. Orlando, Florida, USA: American Association for Artificial Intelligence, 1999, pp. 313–320. ISBN: 0262511061.

A Manipulating factor objects

To manipulate the conditionally independent factors of the transition model and the observation model and factors of the guiding terms we employ a factor `struct`, for which we define `marginalise`, `multiply`, and `divide` operations. Higher-level procedures such as the junction tree algorithm are then defined using these operations. The data structure and operations are all implemented in Julia.

A.1 Struct

Let f_j be a factor featuring a scope which consists of N variables $\boldsymbol{S}_j = \{X_1, \ldots, X_N\}$. Denote the possible realisations of variable $X_i \subset \boldsymbol{S}_j$ with $E_i = \{e_1, \ldots, e_{l_i}\}$, where l_i denotes the number of possible realisations X_i can take. It follows that we may represent the values of the factor

$$f_j : \{x_1, \ldots, x_N\} \mapsto \mathbb{R} \tag{128}$$

with an N-dimensional array, where the i-th dimension has length l_i. We additionally supply a list of unique variable ids, where the i-th element corresponds to the i-th argument of f_j. The order of ids is not arbitrary and adheres to a global ordering system which is shared among all factors. This global ordering provides a consistent framework to relate variables across different factors.

```
struct Factor{N}
    variable_ids::Vector{Int64}
    values::Array{Float64, N}
end
```

A.2 Marginalise

Let f_j be a factor featuring a scope which consists of N variables $\boldsymbol{S}_j = \{X_1, \ldots, X_N\}$. Let f_i denote the factor f_j marginalised over some variables $\boldsymbol{S}_m \subset \boldsymbol{S}_j$, such that $\boldsymbol{S}_i = (\boldsymbol{S}_j \setminus \boldsymbol{S}_m)$. We may compute the factor f_i by matching the unique variable ids to the correct dimensions and then 'collapsing' those dimensions with the built-in `sum` function.

```
function marginalise(phi::Factor, sum_ids::Vector{Int64})
    variable_ids = setdiff(phi.variable_ids, sum_ids)

    sum_dims = [findfirst(id -> id == sum_id, phi.variable_ids)
                for sum_id in sum_ids]

    keep_dims = setdiff(1:lastindex(phi.variable_ids), sum_dims)

    temp = sum(phi.values, dims=sum_dims)
    values = reshape(temp, size(phi.values)[keep_dims])

    Factor(variable_ids, values)
end
```

A.3 Multiply

Let f_i and f_j be two factors featuring scopes $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$. The two scopes may be identical, one scope may be a subset of the other, the two scopes may partially overlap or the two scopes may be fully disjoint from one another. We wish to compute the product of the two factors, i.e.

$$(f_i \cdot f_j)(\boldsymbol{S}_i \cup \boldsymbol{S}_j) = f_i(\boldsymbol{S}_i) \times f_j(\boldsymbol{S}_j). \tag{129}$$

Let $\boldsymbol{S}_k = (\boldsymbol{S}_i \cup \boldsymbol{S}_j)$ denote the union of scopes of f_i and f_j, such that each variable $X \subset \boldsymbol{S}_k$ is a member of $\boldsymbol{S}_i$, a member of $\boldsymbol{S}_j$, or a member of both. If the scope of a factor does not feature some variable $X \subset \boldsymbol{S}_k$, then the N-dimensional array which represents the values of that factor is essentially 'missing' that dimension. For both factors we may add all 'missing' dimensions with the built-in `repeat` function and finally compute the product through element-wise multiplication.

```
function multiply(phi1::Factor, phi2::Factor)
    variable_ids = sort(union(phi1.variable_ids, phi2.variable_ids))

    variable_domain_length_1 = collect(size(phi1.values))
    variable_domain_length_2 = collect(size(phi2.values))
    variable_domain_length = Vector{Int64}(undef, lastindex(variable_ids))

    for (i, id) in enumerate(phi1.variable_ids)
        variable_domain_length[findfirst(jd -> jd == id, variable_ids)] =
        variable_domain_length_1[i]
    end

    for (i, id) in enumerate(phi2.variable_ids)
        variable_domain_length[findfirst(jd -> jd == id, variable_ids)] =
        variable_domain_length_2[i]
    end

    reshape_dims1 = map(i -> (variable_ids[i] in phi1.variable_ids ?
    variable_domain_length[i] : 1), 1:lastindex(variable_ids))

    repeat_dims1  = map(i -> (variable_ids[i] in phi1.variable_ids ?
    1 : variable_domain_length[i]), 1:lastindex(variable_ids))

    temp1 = reshape(phi1.values, tuple(reshape_dims1...))
    vals1 = repeat(temp1, outer=repeat_dims1)

    reshape_dims2 = map(i -> (variable_ids[i] in phi2.variable_ids ?
    variable_domain_length[i] : 1), 1:lastindex(variable_ids))

    repeat_dims2  = map(i -> (variable_ids[i] in phi2.variable_ids ?
    1 : variable_domain_length[i]), 1:lastindex(variable_ids))

    temp2 = reshape(phi2.values, tuple(reshape_dims2...))
    vals2 = repeat(temp2, outer=repeat_dims2)

    values = vals1 .* vals2
    Factor(variable_ids, values)
end
```

A.4 Divide

Let f_i and f_j be two factors featuring scopes $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$, such that $\boldsymbol{S}_j \subset \boldsymbol{S}_i$. The division operation is performed element-wise on the values of f_i and f_j, aligning variables in f_j with the corresponding variables in f_i according to the globally defined variable order.

The division operation is generally only used when computing a conditional distribution from a pair of joint and marginal distributions. If a division by zero occurs, resulting in NaN, such values are replaced with 0 in the resulting factor, such that re-computing the joint distribution results in an object identical to the original.

```
function divide(phi1::Factor, phi2::Factor)
    variable_domain_length = collect(size(phi1.values))

    reshape_dims2 = map(i -> (phi1.variable_ids[i] in phi2.variable_ids ?
    variable_domain_length[i] : 1), 1:lastindex(phi1.variable_ids))

    repeat_dims2  = map(i -> (phi1.variable_ids[i] in phi2.variable_ids ?
    1 : variable_domain_length[i]), 1:lastindex(phi1.variable_ids))

    temp2 = reshape(phi2.values, tuple(reshape_dims2...))
    vals2 = repeat(temp2, outer=repeat_dims2)

    values = phi1.values ./ vals2
    values[isnan.(values)] .= 0.

    Factor(phi1.variable_ids, values)
end
```

B Balanced junction tree representation

The exact recursion $\widehat{g}_t$ will generally not be a member of any tractable family $\mathcal{Q}$, and thus g_t will merely be an approximation. As further motivation for the representation of a distribution in terms of a number of its marginal distributions we discuss an example for which this representation allow us to represent some distribution exactly, in which case Koller & Friedman (2009) refer to the resulting representation as a *balanced junction tree*.

Example B.1. Let Q be a distribution for the random variables A, B, C and D defined as

$$Q(A,B,C,D) = \frac{1}{Z} f_1(A,B) f_2(B,C) f_3(C,D), \tag{130}$$

where

$$Z = \sum_{A,B,C,D} f_1(A,B) f_2(B,C) f_3(C,D) \tag{131}$$

may be considered a normalising constant. We may define a valid junction tree $\mathcal{J} = (\mathcal{R}, \mathcal{E})$ with $\mathcal{R} = \{r_1, r_2, r_3\}$ and $\mathcal{E} = \{(r_1, r_2), (r_2, r_3)\}$, where

$$r_1 = \{A, B, f_1\}, \tag{132}$$
$$r_2 = \{B, C, f_2\}, \tag{133}$$
$$r_3 = \{C, D, f_3\}, \tag{134}$$

such that we have the separators $s_1 = (r_1 \cap r_2) = \{B\}$ and $s_2 = (r_2 \cap r_3) = \{C\}$.

Note that the marginals of the relevant regions are given by

$$\phi_1(A,B) = Q[A,B] = \frac{1}{Z} f_1(A,B) \left(\sum_{C,D} f_2(B,C) f_3(C,D) \right), \tag{135}$$

$$\phi_2(B,C) = Q[B,C] = \frac{1}{Z} \left(\sum_{A} f_1(A,B) \right) f_2(B,C) \left(\sum_{D} f_3(C,D) \right), \tag{136}$$

$$\phi_3(C,D) = Q[C,D] = \frac{1}{Z} \left(\sum_{A,B} f_1(A,B) f_2(B,C) \right) f_3(C,D), \tag{137}$$

and that the marginals of separators are given by

$$\phi_{(1,2)}(B) = Q[B] = \frac{1}{Z} \left(\sum_{A} f_1(A,B) \right) \left(\sum_{C,D} f_2(B,C) f_3(C,D) \right), \tag{138}$$

$$\phi_{(2,3)}(C) = Q[C] = \frac{1}{Z} \left(\sum_{A,B} f_1(A,B) f_2(B,C) \right) \left(\sum_{D} f_3(C,D) \right). \tag{139}$$

By cancellation of terms it follows that

$$Q(A,B,C,D) = \frac{\phi_1(A,B)\,\phi_2(B,C)\,\phi_3(C,D)}{\phi_{(1,2)}(B)\,\phi_{(2,3)}(C)}, \tag{140}$$

i.e. the distribution Q can be *exactly* represented in terms of marginals for the regions which define a valid junction tree. Expectation propagation essentially consists of projecting distributions which do **not** admit such a representation onto a family of distributions (defined by $\mathcal{F}_{\mathcal{R}}$) which does, computing the specific function which minimises the KL-divergence.

C The family of α-divergences

In statistics and information theory literature there exist a great number of divergences which are employed for a variety of applications. In this section we consider the family of *α-divergences* which encompasses many of these divergences as special cases.

Definition C.1. We define the family of **α-divergences** parameterised by $\alpha \in (-\infty, \infty)$:

$$\mathrm{D}_\alpha\left(p \,||\, q\right) = \frac{1}{\alpha\left(1-\alpha\right)}\left(1 - \sum_{\boldsymbol{X}} p\left(\boldsymbol{X}\right)^\alpha q\left(\boldsymbol{X}\right)^{1-\alpha}\right). \tag{141}$$

Note that in particular we have as special cases the well known *Kullback-Leibler* divergences:

$$\mathrm{D}_1\left(p \,||\, q\right) = \lim_{\alpha\to 1} \mathrm{D}_\alpha\left(p \,||\, q\right) = \mathrm{KL}\left(p \,||\, q\right), \tag{142}$$

$$\mathrm{D}_0\left(p \,||\, q\right) = \lim_{\alpha\to 0} \mathrm{D}_\alpha\left(p \,||\, q\right) = \mathrm{KL}\left(q \,||\, p\right). \tag{143}$$

Minka (2005) derives a wide variety of common message passing algorithms as special cases of α-divergence minimisation, and Hernández-Lobato et al. (2016) provide a 'black-box' approach to computing the divergence minimising q for any value of α. Hernández-Lobato et al. note that "the optimal setting of α might reasonably be expected to depend on the learning task that is being considered." Both authors provide further analysis on the effect of α on the resulting target functions q. Much of these discussions are beyond the scope of this work, however, and we therefore highlight only one result provided by Minka which informs our choice of $\alpha = 1$.

Recall that we are not only interested in the term g_t itself but that we must also compute the *next* recursion, i.e. we are also interested in the term $\widehat{g}_{t-1}$ resulting from the approximation $g_t = q_t$. We can think of $\mathcal{Q}$ as an information bottleneck which the backward filtering process must necessarily pass through to remain tractable, such that the divergence D_α defines the manner in which the exact recursion is 'summarised' as a member of $\mathcal{Q}$ for each time step.[11]

Denote the 'un-projected' recursion term which would result for time step $t-1$ if we *do not* first project the exact recursion $\widehat{g}_t$ onto $\mathcal{Q}$ with g'_{t-1}, and denote the recursion if we approximate $\widehat{g}_t$ with q_t with the familiar $\widehat{g}_{t-1}\left(q_t\right)$. It follows that we are not only interested in the projection shown in equation (46) but additionally in a divergence of the form

$$\widetilde{\mathrm{D}}\left(g'_{t-1} \,||\, \widehat{g}_{t-1}\left(q_t\right)\right), \tag{144}$$

for some additional divergence $\widetilde{\mathrm{D}}$. Computing the optimal q_t with respect to the divergence shown in equation (144) is difficult, as it is not a point-to-point comparison but rather involves summation of all possible states $\boldsymbol{x}^{(t)}$. Minka considers essentially this setting from the perspective of prediction, seeking to find the optimal value of α to minimise the divergence between the 'un-projected' recursion and the 'predicted' recursion, using the canonical *relative entropy* (i.e. KL-divergence) shown in equation (142) as $\widetilde{\mathrm{D}}$. The results presented by Minka (2005) show that a value of $\alpha = 1$ is optimal in a variety of settings and in this work we will also work with this value, though we note that analysing the effect of varying α on the performance of the backward filtering forward guiding method may be an interesting avenue for future research. Boyen & Koller (1999) additionally prove that under reasonable assumptions the recursion for the approximate terms g with a value of $\alpha = 1$ employed for the projection step forms a contraction with respect to the relative entropy, lending further credence to this choice.

[11] In summary, the parameter α determines the ratio at which an approximation q is 'punished' for assigning either too much or too little probability mass to a state $\boldsymbol{x}$ compared to p, thus determining if q should either cover the entire distribution or approximate only one *mode*, or something in between.

D Comparison of backward filtering computations

In this section we compare in detail the computations involved throughout the backward marginalisation approach introduced by Van der Meulen & Schauer (2022) and the fully factorised expectation propagation approach introduced in this work.

Consider a process $\mathcal{X}$ which consists of four sub-processes. Each sub-process $\mathcal{X}_i$ interacts with its immediate neighbours $\mathcal{X}_{i-1}$ and $\mathcal{X}_{i+1}$, with suitable restrictions for the 'edge' sub-processes $\mathcal{X}_1$ and $\mathcal{X}_4$. It follows that the transition model of $\mathcal{X}$ is given by

$$P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) = P\left(X_1^{(t)} \mid \boldsymbol{X}_{1:2}^{(t-1)}\right) P\left(X_2^{(t)} \mid \boldsymbol{X}_{1:3}^{(t-1)}\right) P\left(X_3^{(t)} \mid \boldsymbol{X}_{2:4}^{(t-1)}\right) P\left(X_4^{(t)} \mid \boldsymbol{X}_{3:4}^{(t-1)}\right). \tag{145}$$

Let g_t be a guiding term given by a fully factorised factorisation of the state space:

$$g_t\left(\boldsymbol{X}^{(t)}\right) = \prod_{i=1}^{4} \phi_i\left(X_i^{(t)}\right). \tag{146}$$

Note that both the transition model and the guiding term factorise with respect to the set of state variables, and thus:

$$\widehat{g}_{t-1}\left(\boldsymbol{X}^{(t-1)}\right) = \sum_{\boldsymbol{X}^{(t)}} P\left(\boldsymbol{X}^{(t)} \mid \boldsymbol{X}^{(t-1)}\right) g_t\left(\boldsymbol{X}^{(t)}\right) \tag{147}$$

$$= \prod_{i=1}^{4} \left(\sum_{X_i^{(t)}} P\left(X_i^{(t)} \mid \boldsymbol{X}_{\mathrm{ne}(i)}^{(t-1)}\right) \phi_i\left(X_i^{(t)}\right) \right) \tag{148}$$

$$= \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)}\right) \psi_2\left(\boldsymbol{X}_{1:3}^{(t-1)}\right) \psi_3\left(\boldsymbol{X}_{2:4}^{(t-1)}\right) \psi_4\left(\boldsymbol{X}_{3:4}^{(t-1)}\right). \tag{149}$$

Assume that each state variable takes one of 3 possible realisations in $E = \{e_1, e_2, e_3\}$. We identify the values of the factor ψ_1 with a 2-dimensional array:

$$F^1 = \begin{bmatrix} \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_1, e_1)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_1, e_2)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_1, e_3)\right) \\ \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_2, e_1)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_2, e_2)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_2, e_3)\right) \\ \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_3, e_1)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_3, e_2)\right) & \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)} = (e_3, e_3)\right) \end{bmatrix}. \tag{150}$$

We identify the values of the factor ψ_4 with a similar (3×3)-array F^4 with elements:

$$f_{i,j}^4 = \psi_4\left(\boldsymbol{X}_{3:4}^{(t-1)} = (e_i, e_j)\right). \tag{151}$$

The factors ψ_2 and ψ_3 each concern three state variables. We identify their values with 3-dimensional arrays F^2 and F^3 with elements:

$$f_{i,j,k}^2 = \psi_2\left(\boldsymbol{X}_{1:3}^{(t-1)} = (e_i, e_j, e_k)\right), \tag{152}$$

$$f_{i,j,k}^3 = \psi_3\left(\boldsymbol{X}_{2:4}^{(t-1)} = (e_i, e_j, e_k)\right). \tag{153}$$

To be explicit: We identify the values of ψ_2 with a $(3 \times 3 \times 3)$-array:

$$F^2_{:,:,1} = \begin{bmatrix} \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_1, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_2, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_3, e_1)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_1, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_2, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_3, e_1)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_1, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_2, e_1)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_3, e_1)\right) \end{bmatrix}, \tag{154}$$

$$F^2_{:,:,2} = \begin{bmatrix} \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_1, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_2, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_3, e_2)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_1, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_2, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_3, e_2)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_1, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_2, e_2)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_3, e_2)\right) \end{bmatrix}, \tag{155}$$

$$F^2_{:,:,3} = \begin{bmatrix} \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_1, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_2, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_1, e_3, e_3)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_1, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_2, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_2, e_3, e_3)\right) \\ \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_1, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_2, e_3)\right) & \psi_2\left(\boldsymbol{X}^{(t-1)}_{1:3} = (e_3, e_3, e_3)\right) \end{bmatrix}. \tag{156}$$

We now describe how to compute g_{t-1} from the factors ψ_i, assuming that there is no observation at time step $t-1$.

D.1 Backward marginalisation

With backward marginalisation we 'split' each factor into a set of marginal functions for each state variable, and then for each state variable take the product of all relevant marginal functions. For example:

$$\phi_1\left(X_1^{(t-1)}\right) = \left(\sum_{X_2^{(t-1)}} \psi_1\left(X_1^{(t-1)}, X_2^{(t-1)}\right)\right) \times \left(\sum_{X_2^{(t-1)}, X_3^{(t-1)}} \psi_2\left(X_1^{(t-1)}, X_2^{(t-1)}, X_3^{(t-1)}\right)\right). \tag{157}$$

We identify the factor ϕ_1 with a 1-dimension array, i.e. a vector, of length 3:

$$\Phi_1 = \begin{bmatrix} \phi_1\left(X_1^{(t-1)} = e_1\right) \\ \phi_1\left(X_1^{(t-1)} = e_2\right) \\ \phi_1\left(X_1^{(t-1)} = e_3\right) \end{bmatrix} = \begin{bmatrix} \left(\sum_{j=1}^3 f^1_{1,j}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^2_{1,j,k}\right) \\ \left(\sum_{j=1}^3 f^1_{2,j}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^2_{2,j,k}\right) \\ \left(\sum_{j=1}^3 f^1_{3,j}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^2_{3,j,k}\right) \end{bmatrix} \tag{158}$$

$$= \begin{bmatrix} \left(f^1_{1,1} + f^1_{1,2} + f^1_{1,3}\right) \times \left(\left(f^2_{1,1,1} + f^2_{1,2,1} + f^2_{1,3,1}\right) + \left(f^2_{1,1,2} + f^2_{1,2,2} + f^2_{1,3,2}\right) + \left(f^2_{1,1,3} + f^2_{1,2,3} + f^2_{1,3,3}\right)\right) \\ \left(f^1_{2,1} + f^1_{2,2} + f^1_{2,3}\right) \times \left(\left(f^2_{2,1,1} + f^2_{2,2,1} + f^2_{2,3,1}\right) + \left(f^2_{2,1,2} + f^2_{2,2,2} + f^2_{2,3,2}\right) + \left(f^2_{2,1,3} + f^2_{2,2,3} + f^2_{2,3,3}\right)\right) \\ \left(f^1_{3,1} + f^1_{3,2} + f^1_{3,3}\right) \times \left(\left(f^2_{3,1,1} + f^2_{3,2,1} + f^2_{3,3,1}\right) + \left(f^2_{3,1,2} + f^2_{3,2,2} + f^2_{3,3,2}\right) + \left(f^2_{3,1,3} + f^2_{3,2,3} + f^2_{3,3,3}\right)\right) \end{bmatrix}. \tag{159}$$

And similarly for other factors, i.e.

$$\Phi_2 = \begin{bmatrix} \phi_2\left(X_2^{(t-1)} = e_1\right) \\ \phi_2\left(X_2^{(t-1)} = e_2\right) \\ \phi_2\left(X_2^{(t-1)} = e_3\right) \end{bmatrix} = \begin{bmatrix} \left(\sum_{i=1}^3 f^1_{i,1}\right) \times \left(\sum_{i=1}^3 \sum_{k=1}^3 f^2_{i,1,k}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^3_{1,j,k}\right) \\ \left(\sum_{i=1}^3 f^1_{i,2}\right) \times \left(\sum_{i=1}^3 \sum_{k=1}^3 f^2_{i,2,k}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^3_{2,j,k}\right) \\ \left(\sum_{i=1}^3 f^1_{i,3}\right) \times \left(\sum_{i=1}^3 \sum_{k=1}^3 f^2_{i,3,k}\right) \times \left(\sum_{j=1}^3 \sum_{k=1}^3 f^3_{3,j,k}\right) \end{bmatrix}. \tag{160}$$

D.2 Expectation propagation

With expectation propagation we take the product of all factors and then compute a marginal function for each state variable. For example:

$$\phi_1\left(X_1^{(t-1)}\right) = \sum_{\boldsymbol{X}_{2:4}^{(t-1)}} \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)}\right)\psi_2\left(\boldsymbol{X}_{1:3}^{(t-1)}\right)\psi_3\left(\boldsymbol{X}_{2:4}^{(t-1)}\right)\psi_4\left(\boldsymbol{X}_{3:4}^{(t-1)}\right). \tag{161}$$

Of course we do not actually compute the product of all factors directly but instead use the junction tree algorithm to efficiently compute all marginals. For example:

$$\phi_1\left(X_1^{(t-1)}\right) = \sum_{X_2^{(t-1)}} \psi_1\left(\boldsymbol{X}_{1:2}^{(t-1)}\right) \sum_{X_3^{(t-1)}} \psi_2\left(\boldsymbol{X}_{1:3}^{(t-1)}\right) \sum_{X_4^{(t-1)}} \psi_3\left(\boldsymbol{X}_{2:4}^{(t-1)}\right)\psi_4\left(\boldsymbol{X}_{3:4}^{(t-1)}\right). \tag{162}$$

We now identify the factor ϕ_1 with a vector

$$\Phi_1 = \begin{bmatrix} \phi_1\left(X_1^{(t-1)} = e_1\right) \\ \phi_1\left(X_1^{(t-1)} = e_2\right) \\ \phi_1\left(X_1^{(t-1)} = e_3\right) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^3 \left(f_{1,j}^1 \times \left(\sum_{k=1}^3 \left(f_{1,j,k}^2 \times \left(\sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right)\right)\right)\right)\right) \\ \sum_{j=1}^3 \left(f_{2,j}^1 \times \left(\sum_{k=1}^3 \left(f_{2,j,k}^2 \times \left(\sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right)\right)\right)\right)\right) \\ \sum_{j=1}^3 \left(f_{3,j}^1 \times \left(\sum_{k=1}^3 \left(f_{3,j,k}^2 \times \left(\sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right)\right)\right)\right)\right) \end{bmatrix}. \tag{163}$$

And similarly for other factors, i.e.

$$\Phi_2 = \begin{bmatrix} \phi_2\left(X_2^{(t-1)} = e_1\right) \\ \phi_2\left(X_2^{(t-1)} = e_2\right) \\ \phi_2\left(X_2^{(t-1)} = e_3\right) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^3 \sum_{k=1}^3 \left(f_{i,1}^1 \times f_{i,1,k}^2 \times \left(\sum_{l=1}^3 \left(f_{1,k,l}^3 \times f_{k,l}^4\right)\right)\right) \\ \sum_{i=1}^3 \sum_{k=1}^3 \left(f_{i,2}^1 \times f_{i,2,k}^2 \times \left(\sum_{l=1}^3 \left(f_{2,k,l}^3 \times f_{k,l}^4\right)\right)\right) \\ \sum_{i=1}^3 \sum_{k=1}^3 \left(f_{i,3}^1 \times f_{i,3,k}^2 \times \left(\sum_{l=1}^3 \left(f_{3,k,l}^3 \times f_{k,l}^4\right)\right)\right) \end{bmatrix}. \tag{164}$$

We may identify the values of the 'inner-most' computation for both factors with a (3×3)-array with elements

$$\mu = \begin{bmatrix} \sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right) \\ \sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right) \\ \sum_{l=1}^3 \left(f_{j,k,l}^3 \times f_{k,l}^4\right) \end{bmatrix} = \begin{bmatrix} \sum_{l=1}^3 \left(f_{1,1,l}^3 \times f_{1,l}^4\right) & \sum_{l=1}^3 \left(f_{1,2,l}^3 \times f_{2,l}^4\right) & \sum_{l=1}^3 \left(f_{1,3,l}^3 \times f_{3,l}^4\right) \\ \sum_{l=1}^3 \left(f_{2,1,l}^3 \times f_{1,l}^4\right) & \sum_{l=1}^3 \left(f_{2,2,l}^3 \times f_{2,l}^4\right) & \sum_{l=1}^3 \left(f_{2,3,l}^3 \times f_{3,l}^4\right) \\ \sum_{l=1}^3 \left(f_{3,1,l}^3 \times f_{1,l}^4\right) & \sum_{l=1}^3 \left(f_{3,2,l}^3 \times f_{2,l}^4\right) & \sum_{l=1}^3 \left(f_{3,3,l}^3 \times f_{3,l}^4\right) \end{bmatrix}. \tag{165}$$

To be explicit: the computation of these elements is **shared** between the computation of Φ_1 and Φ_2, and exploiting this insight forms the basis of the efficiency of the junction tree algorithm.

Note that great care must be taken to sum over the correct dimension of involved factors if they do not feature the same variables in their scope. In Appendix A we refer to this as adding the 'missing' dimensions, i.e. copying each value along a 'missing' axes, as this is how it is handled in the current implementation.

E Markov-chain Monte Carlo

We want to target the joint distribution

$$\Pi\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{\Theta}\right) = P\left(\boldsymbol{X}^{(1:T)}, \boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right).$$

E.1 X-step

By definition

$$P\left(\boldsymbol{X}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = P^{\star}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) = P^{\circ}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) \times \frac{\widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times \Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right),$$

where

$$\Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) = \prod_{t=1}^{T-1} \left(\frac{P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right)}{g_t\left(\boldsymbol{X}^{(t)}\right)} \right) \times \left(\frac{P\left(\boldsymbol{Y}^{(T)} = \boldsymbol{y}^{(T)} \mid \boldsymbol{X}^{(T)}\right)}{g_T\left(\boldsymbol{X}^{(T)}\right)} \right).$$

The conditional likelihood term

$$P_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)$$

is intractable but cancels between samples from the posterior. Assuming that the terminal observation term is exactly representable in factored from by g_T (i.e. g_T is not an approximation) it follows that

$$\frac{\Pi\left(\boldsymbol{X}^{(1:T)} = \widehat{\boldsymbol{x}}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}\right)}{\Pi\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}\right)} = \frac{P^{\circ}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \widehat{\boldsymbol{x}}^{(1:T)}\right) \times \widetilde{\Psi}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T-1)} = \widehat{\boldsymbol{x}}^{(1:T-1)}\right)}{P^{\circ}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times \widetilde{\Psi}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T-1)} = \boldsymbol{x}^{(1:T-1)}\right)},$$

where

$$\widetilde{\Psi}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T-1)}\right) = \prod_{t=1}^{T-1} \left(\frac{P\left(\boldsymbol{Y}^{(t)} = \boldsymbol{y}^{(t)} \mid \boldsymbol{X}^{(t)}\right) \times \sum_{\boldsymbol{X}^{(t+1)}} P_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(t+1)} \mid \boldsymbol{X}^{(t)}\right) g_{t+1}\left(\boldsymbol{X}^{(t+1)}\right)}{g_t\left(\boldsymbol{X}^{(t)}\right)} \right).$$

E.2 θ-step

By definition

$$P\left(\boldsymbol{\Theta} \mid \boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = \frac{P\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}, \boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)},$$

where

$$P\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}, \boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = P\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)} \mid \boldsymbol{\Theta}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) P\left(\boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right).$$

The first term on the left hand side is the distribution of the conditional model $\mathcal{X}^\star$, and the posterior distribution of the model parameters is additionally given by

$$P\left(\boldsymbol{\Theta} \mid \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = \frac{P\left(\boldsymbol{\Theta}\right)}{P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \times P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)} \mid \boldsymbol{\Theta}\right).$$

The likelihood term in the denominator

$$P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)$$

is intractable but cancels between samples from the posterior. The conditional likelihood term in the numerator does *not* cancel between samples from the guided distribution, but notably also appears in the relation:

$$P\left(\boldsymbol{X}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}, \boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right) = P^{\circ}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right) \times \frac{\widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}\right)} \times \Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)}\right).$$

The conditional likelihood term for $\boldsymbol{\theta}$ cancels between the probability terms for the guided sample and the posterior of the model parameters, and it follows that

$$\begin{aligned}\frac{\Pi\left(\boldsymbol{\Theta} = \widehat{\boldsymbol{\theta}} \mid \boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right)}{\Pi\left(\boldsymbol{\Theta} = \boldsymbol{\theta} \mid \boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right)} &= \frac{P^{\circ}_{\widehat{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times \widetilde{P}_{\widehat{\boldsymbol{\theta}}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)}{P^{\circ}_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times \widetilde{P}_{\boldsymbol{\theta}}\left(\boldsymbol{Y}^{(1:T)} = \boldsymbol{y}^{(1:T)}\right)} \\ &\times \frac{\Psi_{\widehat{\boldsymbol{\theta}}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times P\left(\boldsymbol{\Theta} = \widehat{\boldsymbol{\theta}}\right)}{\Psi_{\boldsymbol{\theta}}\left(\boldsymbol{X}^{(1:T)} = \boldsymbol{x}^{(1:T)}\right) \times P\left(\boldsymbol{\Theta} = \boldsymbol{\theta}\right)}.\end{aligned}$$