Automatic Backward Filtering Forward Guiding for Markov processes and graphical models

Frank van der Meulen[*]
Delft Institute of Applied Mathematics (DIAM)
Delft University of Technology
Email: f.h.vandermeulen@tudelft.nl

Moritz Schauer[§]
Chalmers University of Technology
and University of Gothenburg
E-mail: smoritz@chalmers.se

Abstract

We incorporate discrete and continuous time Markov processes as building blocks into probabilistic graphical models with latent and observed variables. We introduce the automatic Backward Filtering Forward Guiding (BFFG) paradigm (Mider et al., 2020) for programmable inference on latent states and model parameters. Our starting point is a generative model, a forward description of the probabilistic process dynamics. We backpropagate the information provided by observations through the model to transform the generative (forward) model into a pre-conditional model guided by the data. It approximates the actual conditional model with known likelihood-ratio between the two. The backward filter and the forward change of measure are suitable to be incorporated into a probabilistic programming context because they can be formulated as a set of transformation rules.

The guided generative model can be incorporated in different approaches to efficiently sample latent states and parameters conditional on observations. We show applicability in a variety of settings, including Markov chains with discrete state space, interacting particle systems, state space models, branching diffusions and Gamma processes.

Keywords: Bayesian network, Branching diffusion process, category of optics, guided process, guided proposal, h-transform, $\widetilde{h}$-transform, normalising flow, probabilistic graphical model, string diagram, variational inference.

1 Introduction

1.1 Problem description

A large number of scientific disciplines makes use of stochastic processes for probabilistic modelling of time evolving phenomena and complex systems under uncertainty. In this setting data assimilation and statistical inference is challenging. More so, if this is to be incorporated as components into larger probabilistic models, to be described using modelling languages, and if it is desired to perform inference in an at least semi-automated fashion in probabilistic programming. In this paper we provide a way to incorporate Markovian stochastic processes as building blocks into probabilistic graphical models and efficiently perform inference on them.

[*]Mekelweg 4, 2628 CD Delft, The Netherlands
[§]Chalmers Tvärgata 3, 41296 Göteborg, Sweden

Probabilistic graphical models capture the structure of a probabilistic model in the form of a graph from which conditional independence properties can easily be extracted. We assume that the dependence structure of random quantities is represented by a directed acyclic graph (DAG) with vertices $t \in \mathcal{T}$ and edges directed from a set of source vertices towards the leaves (commonly used terminology for graphical models is reviewed in appendix A.) To each vertex t of the DAG corresponds a random quantity X_t, constituting a Bayesian network. Often the DAG arises as the dependency structure of a probabilistic program. The conditional distribution of X_t given all X_s corresponding to the ancestors s of t in the DAG is prescribed and assumed to depend only on the parents, the random quantity $X_{\mathrm{pa}(t)}$, the set of vertices s for which there is a directed edge from s to t. Thus the DAG can be obtained from a generative model where each new value X_t is sampled conditionally on its parents $X_{\mathrm{pa}(t)}$ (except for the sources). See Figure 1 for a simple example with a root (a single source).

Assume we only observe X_v for each leaf v of the DAG. Leaves can be attached anywhere in the DAG, also at the sources. Therefore edges ending in leaves can be conceptualised as observation schemes representing random errors. The smoothing problem consists of finding the exact joint distribution of all X_s, where $s \in \mathcal{T}$ is not a leaf, conditional on the observations. In this paper we give a novel way to sample from the smoothing distribution thereby extending well known algorithms for graphical models. Contrary to other methods, the proposed framework does not require modification of the DAG or dependency structure of the probabilistic program, for example by addition of edges. We are especially interested in models where some X_t are obtained as the endpoint of the evolution of a Markov process starting from the parent X_s over a certain time span. The transitions may depend on parameters in the Markov kernel which need to be estimated.

The problem setting encompasses training of stochastic residual neural networks (Wang et al., 2018) and their continuous-time counterparts neural stochastic differential equations (Chen et al., 2018), as well as other settings where the output of a complicated latent process is observed and the latent process is obtained from iteration and composition of models which are tractable (only) locally in time, in space or along single edges in the DAG.

1.2 Approach

We start from a Markovian forward description of the possibly nonlinear, non-reversible process dynamics. In essence, by first filtering backwards an approximate process from the leaves towards the root and then performing a change of measure on the transitions of the generative (forward) model using the filtering information, we derive a generative model *guided* by the data. This model provides a structure preserving approximation to the true model conditional on the observations to sample from. Notably, we also transform transitions which correspond to continuous time Markov processes using an exponential change of measure (Palmowski and Rolski, 2002). Also the seminal paper Delyon and Hu (2006) on diffusion bridges uses related ideas. This *pre-conditional* model, which combines information from data and the model, can then be incorporated in different approaches to efficiently sample the actual smoothing distribution, for example Hamiltonian Monte Carlo, particle methods or variational inference. The setup allows to obtain a differentiable likelihood through a reparametrisation in a natural way.

As a familiar special case of the procedure we obtain the backwards filtering forward sampling algorithm for linear state space models (in this case the guided approximation is exact and can be directly sampled as latent path given the observations). Also our previously introduced backward filtering forward guiding (BFFG) algorithm for discretely observed stochastic differential equations (Mider et al., 2020) is a special case. More examples, involving Gamma processes, discrete Markov chains and Poisson processes are also covered in this paper. Key to this is the

insight that the dynamics of a conditioned Markov process or graphical model can be obtained by Doob's h-transform and that guided proposals (as defined in Schauer et al. (2017)) result from an approximation $\widetilde{h}$ to this transform. Some of the results in Schauer et al. (2017) then become simple consequences of results on exponential change of measure for Markov processes from Palmowski and Rolski (2002). The idea of using an approximate h-transform dates back to at least Glynn and Iglehart (1989) in the setting of rare event simulation. We further use the Feynman-Kac approach to give the backward filtering pass a sampling interpretation.

The backward filter and the forward change of measure are suitable to be incorporated into probabilistic programming environments. Similar to Law and Wilkinson (2019) we use arguments from category theory to derive a principle of compositionality for our method, that is a principle which "describes and quantifies how complex things can be assembled out of simpler parts", as the eponymous journal Compositionality defines it. By this it is enough to provide a library of transformation rules and tractable approximations for backward filtering each single transition, the elementary building blocks of a model, and an implementable composition rule to obtain guided processes for larger models. Note that we speak of "guided processes" rather than "guided proposals" (used in earlier works), the motivation being that its applicability is not restricted to a proposal distribution in a Metropolis-Hastings algorithm.

We provide a library of some of such rules and approximations as Julia package `Mitosis.jl`,[1] following the spirit of the `ChainRules.jl` package, Revels et al. (2018-2020), providing common differentiation rules to the different automatic differentiation packages in Julia ecosystem.

1.3 Related work

In case of a line graph with edges to observation leaves at each vertex (a hidden Markov model) there are two well known cases for computing the smoothing distribution: *(i)* if X is discrete the forward-backward algorithm (Murphy (2012), section 17.4.3), *(ii)* for linear Gaussian systems the Kalman Smoother, also known as Rauch-Tung-Striebel smoother (Murphy (2012), section 18.3.2) for the marginal distributions or its sampling version, where samples from the smoothing distribution are obtained by the forward filtering, backward sampling algorithm, Carter and Kohn (1994). Pearl (1988) gave an extension of the forward-backward algorithm from chains to trees by an algorithm known as "belief propagation" or "sum-product message passing", either on trees or poly-trees. This algorithm consists of two message passing phases. In the "collect evidence" phase, messages are sent from leaves to the root; the "distribute evidence" phase ensures updating of marginal probabilities or sampling joint probabilities from the root towards the leaves. The algorithm can be applied to junction trees as well, and furthermore, the approximative loopy belief propagation applies belief propagation to sub-trees of the graph. A review is given in Jordan (2004).

Chou et al. (1994) extended the classical Kalman smoothing algorithm for linear Gaussian systems to dyadic trees by using a fine-to-coarse filtering sweep followed by a course-to fine smoothing sweep. This setting arises as a special case of our framework. Extensions of filtering on Triplet Markov Trees and pairwise Markov trees are dealt with in Bardel and Desbouvries (2012) and Desbouvries et al. (2006) respectively.

Particle filters have been employed in related settings, see Doucet and Lee (2018) for an overview, but the resampling operations of particle filters result in the simulated likelihood function being non-differentiable in the parameters, which unlike our approach is an obstacle to gradient based inference. Briers et al. (2010) consider particle methods for state-space models using (an approximation to) the backward-information filter. The latter is alike the backward filtering step proposed in this paper. This idea is also exploited recently in Ju et al. (2021) (in particular sec-

[1] https://github.com/mschauer/Mitosis.jl.

tion 3.2) to deal with inferential problems for agent-based models of disease transmission. The numerical example we present in section 12 is related to Ju et al. (2021), though the observation scheme considered in that paper is different from ours.

For variational inference, Ambrogioni et al. (2021) propose an approach which, similar to ours, preserves the Markovian structure of the target, this by learning local approximations to the conditional dynamics.

In continuous time models, smoothing of discretely observed diffusion processes received a lot of attention, see e.g. Yonekura and Beskos (2020) and Mider et al. (2020) and references therein. Some of our ideas are related to Milstein et al. (2007) in which transition densities of Markov processes are estimated by combining forward and reverse representations.

1.4 Novelty and contribution

Building up on successes with guided processes (Schauer et al., 2017) in the specific, but challenging problem of likelihood-based inference for diffusion process, this work gives a new, holistic perspective on automatic probabilistic programming in general.

Our approach stands out by being:

1. *General*: It gives a new understanding of guided processes as approximation of Doob's h-transforms, allowing to treat guided processes for processes indexed by a DAG and for processes with continuous index set such as continuous-time processes jointly, in both cases for discrete and continuous state spaces.

2. *Agnostic*: Our approach is agnostic with respect to the choice of inferential method, allowing for implementations based on particle samplers, variational inference or Metropolis-Hastings samplers. This includes the Metropolis-Hastings version of the backward filtering forward guiding algorithm applied in Mider et al. (2020) to discretely observed diffusion processes.

3. *Scalable*: By using as starting point the unconditional dynamics only modified by a change of measure, which is obtained by back-propagating information from the observations through the system, guided processes can be applied in high- and infinite-dimensional inference problems without weight degeneracy.

4. *Automatic*: The proposed algorithm is suitable for automatic transformation of probabilistic programs.

5. *Compositional*: Transformation of a probabilistic program relies on the composition of the transformations of its elementary building blocks in a backward and a forward pass. We provide a description of the compositional structure of the transformation in the language of category theory.

6. *Differentiable*: The variational class of guided processes is equipped with natural normalising flows.

7. *Probabilistic and likelihood-based*: We provide likelihood-based inference for intractable models and give a probabilistic representation of the backward pass using the adjoint process.

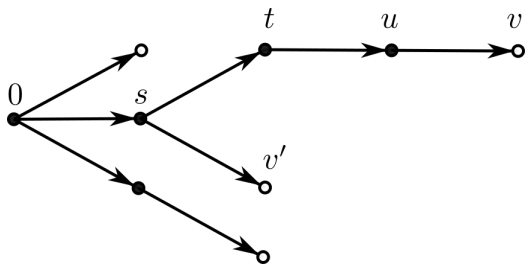


FIGURE 1: A simple tree, with • denoting latent vertices and ∘ leaf-vertices. The root is denoted 0. Along each edge the process evolves according to either one step of a discrete-time Markov chain or a time-span of a continuous-time Markov process.

1.5 Outline

In section 2 we explain the setting and some notation. In section 3 we define Doob's h-transform for Markov processes indexed on a tree. In section 4 we introduce guided processes for directed trees and, more generally, Bayesian networks. Subsequently, the computational structure of Doob's h-transforms and guided processes is detailed in section 5. Here, we explain how an efficient implementation is obtained by piecing together forward and backward maps. In section 6 we treat guided processes from a categorical perspective using optics, later sections do not depend on this section. Examples of guided processes with tractable backward map are given in section 7. Various ways in which the obtained results can be used for inference are discussed in section 8. In section 9 we define guided processes for continuous-time Markov processes via the infinitesimal generator of the Markov process. Examples are given in section 10. In section 11 we give a probabilistic representation of Doob's h-transform using the adjoint process, both in case of discrete time and continuous time transitions over an edge. We end with a numerical example in section 12. The appendix contains some additional material and postponed proofs. Some notation is reviewed in appendix A.

2 Graphical models

Throughout, we use various terms and notations from graph theory (see e.g. chapter 10 in Murphy (2012)) which are defined formally in appendix A. Let $\mathcal{G} = (\mathcal{T}, \mathcal{E})$ be a directed, finite graph without parallel edges and self-loops, with vertices $\mathcal{T}$ and edges $\mathcal{E}$, where (s,t) or $s \to t$ denotes denotes a directed edge from s to t, $s, t \in \mathcal{T}$. Without first argument, $\to t$ denotes the edge or the set of edges to $t \in \mathcal{T}$. We write $s \rightsquigarrow t$ if $s = t$ or there is a directed path from s to t. We denote by $\mathrm{pa}(t)$, $\mathrm{ch}(t)$, $\mathrm{anc}(t)$ the parents, children and ancestors of a vertex t respectively. The set of leaves (sinks) is denoted by $\mathcal{V}$. For each vertex t, we denote by $\mathcal{V}_t = \{v \in \mathcal{V}, t \rightsquigarrow v\}$ the leaves which are descendants of t (including t, if t is a leaf).

To simplify the presentation we will assume a designated root vertex $0 \notin \mathcal{T}$ with deterministic $X_0 = x_0$ and formally define $0 = \mathrm{pa}(s)$ for sources, that is vertices $s \in \mathcal{T}$ without parent(s) in $\mathcal{T}$. In this sense, either x_0 represents a fixed or $X_{\mathrm{ch}(0)}$ a random starting configuration with possibly multiple source vertices (sources in $\mathcal{S}$).

Finally, define $\mathcal{S} = \mathcal{T} \setminus \mathcal{V}$ (the set of non-leaf vertices) and $\mathrm{pa}(\mathcal{V})$ to be the set of vertices in $\mathcal{T}$ that are parent of at least one leaf node and set $\mathcal{S}_0 = \mathcal{S} \cup \{0\}$.

Setting: *We consider a probabilistic graphical model on $\mathcal{G}$, a random process $(X_t, t \in \mathcal{T})$ indexed by $\mathcal{T}$ such that X_t depends on its ancestors* $\mathrm{anc}(t)$ *only through its parent(s)* $\mathrm{pa}(t)$ *and*

where random variables X_s, X_t, $s \not\to t$, $t \not\to s$ are independent conditional on all of their parents.

We will extend this setting in section 9 to the case where edges can represent the evolution of a continuous-time Markov process during a fixed time interval.

We assume that X_t takes values in Borel spaces $(E_t, \mathfrak{B}_t)$ and existence of transition densities $p_{\mathrm{pa}(t)\to t}$ with respect to dominating measures λ, that is densities of the conditional distribution

$$\mathbb{P}(X_t \in \mathrm{d}y \mid X_{\mathrm{pa}(t)} = x) = p_{\mathrm{pa}(t)\to t}(x; y)\lambda(\mathrm{d}y), \qquad t \in \mathcal{T}, \quad x \in E_{\mathrm{pa}(t)}$$

and write $p_{\to t} = p_{\mathrm{pa}(t)\to t}$ for short. λ can be as simple as $\lambda(\mathrm{d}x) = \mathrm{d}x$, but may depend on t and even on $x_{\mathrm{pa}(t)}$. In that case, λ constitutes a reference kernel specific to the transition (s, t) instead of a reference measure, but we omit the dependency in our notation.

Thus, the joint density factorises as

$$p(x_{\mathcal{T}}) = \prod_{t\in\mathcal{T}} p_{\to t}(x_{\mathrm{pa}(t)}; x_t). \tag{2.1}$$

For subsets $B \subset \mathcal{T}$, we write $X_B = (X_b,\, b \in B)$ and $x_B = (x_b,\, b \in B)$.

3 Doob's h-transform for smoothing on a directed tree

We first specialise to the setting where $\mathcal{G}$ is a connected tree directed to the leaves, i.e. each vertex has a unique parent, except from the formal root vertex 0. Thus, edges point towards the leaves and "backwards" is to be understood as pointing from leaves to the root.

We are interested in the distribution of $X_{\mathcal{S}}$ conditional on $X_{\mathcal{V}}$, i.e. the smoothing distribution. The values at the leaf vertices can be interpreted as observations without error. Alternatively, $X_{\mathrm{pa}(\mathcal{V})}$ can be thought of as observed quantities with the edges $\mathrm{pa}(v) \to v$, $v \in \mathcal{V}$ representing observation errors. Note that each vertex, including the root 0, may have one or more leaves as its children. A simple example is depicted in Figure 1.

The density of the smoothing distribution can be derived using Doob's h-transform. Using Bayesian notation for the next display,

$$p(x_t \mid x_{\mathrm{pa}(t)}, x_{\mathcal{V}_t}) \propto p(x_t, x_{\mathcal{V}_t} \mid x_{\mathrm{pa}(t)}) = p(x_t \mid x_{\mathrm{pa}(t)})h_t(x_t),$$

where the final term on the right-hand-side is the h-transform at vertex t, i.e. $h_t(x_t) = p(x_{\mathcal{V}_t} \mid x_t)$. Within the subtree with root vertex t, this can be interpreted as the likelihood of x_t. Note that the observations are dropped from the notation as they are fixed throughout.

We now explain how h_s can be computed recursively for a directed tree, starting from the leaves back to the roots. Firstly, for any vertex connected to a leaf, we define

$$h_{\to v}(x) = p_{\mathrm{pa}(v)\to v}(x; x_v) \quad v \in \mathcal{V} \tag{3.1}$$

(recall we write $h_{\to t} \equiv h_{\mathrm{pa}(t)\to t}$). For edges not connected to a leaf we define

$$h_{\to s}(x) = \int h_s(y)p_{\to s}(x; y)\lambda(\mathrm{d}y) \quad s \in \mathcal{S}. \tag{3.2}$$

By the Markov property and a recursive argument we have

$$h_s(x) = \prod_{t\in\mathrm{ch}(s)} h_{s\to t}(x), \qquad s \in \mathcal{S}_0. \tag{3.3}$$

This can be interpreted as collecting all messages from children at vertex s, which we call *fusion*.

Example 3.1. We illustrate this recursive procedure for the labeled branches of the tree in Figure 1. Here, it is easy to see that $h_{u\to v}(x) = p_{u\to v}(x; x_v)$ and $h_{s\to v'}(x) = p_{s\to v'}(x; x_{v'})$. Next, at vertex u we have $h_u(x) = p_{u\to v}(x; x_v)$. Now moving backwards, we get along the edge connecting vertices u and t

$$h_{u\to t}(x) = \int h_u(y) p_{u\to t}(x; y)\lambda(\mathrm{d}y).$$

As t has only vertex u as child we get $h_t(x) = h_{u\to t}(x)$. Next, we compute $h_{s\to t}(x) = \int h_t(y) p_{s\to t}(x; y)\lambda(\mathrm{d}y)$ and at vertex s set $h_s(x) = h_{s\to t}(x)h_{s\to v'}(x)$.

With this notation, the process X that is conditioned on its values on the leaves has transitions

$$\mathbb{P}(X_s \in \mathrm{d}y \mid X_{\mathrm{pa}(s)} = x, X_{\mathcal{V}_s} = x_{\mathcal{V}_s}) = p^\star_{\to s}(x; y)\lambda(\mathrm{d}y), \qquad s \in \mathcal{S}$$

where

$$p^\star_{\to s}(x; y) = p_{\to s}(x; y)\frac{h_s(y)}{h_{\to s}(x)} \tag{3.4}$$

and $h_{\to s}(x)$ shows up as normalising constant and one could define it that way. The quantity $h_0(x_0)$ sometimes is referred to as the *evidence* and is of interest as well.

Let $X^\star$ be the process on $\mathcal{G}$ initialised starting in x_0 and evolving according to the transition densities in (3.4). Note that $X^\star$ is again a Markov process on $\mathcal{G}$. This only holds if $\mathcal{G}$ is a tree.

The familiar main result of this section concerns the likelihood ratio between $X^\star$ and X globally.

Theorem 3.2. We have

$$\prod_{t\in\mathcal{S}} \frac{p^\star_{\to t}(x_{\mathrm{pa}(t)}; x_t)}{p_{\to t}(x_{\mathrm{pa}(t)}; x_t)} = \frac{1}{h_0(x_0)} \prod_{v\in\mathcal{V}} h_{\to v}(x_{\mathrm{pa}(v)}),$$

where the density of the process conditioned on the values at its leaves is given in (3.4).

Proof. equation (3.4) implies

$$\begin{aligned}\prod_{s\in\mathcal{S}} \frac{p^\star_{\to s}(x_{\mathrm{pa}(s)}; x_s)}{p_{\to s}(x_{\mathrm{pa}(s)}; x_s)} &= \prod_{s\in\mathcal{S}} \frac{h_s(x_s)}{h_{\to t}(x_{\mathrm{pa}(s)})} = \frac{\prod_{s\in\mathcal{S}}\prod_{t\in\mathrm{ch}(s)} h_{s\to t}(x_s)}{\prod_{s\in\mathcal{S}} h_{\to s}(x_{\mathrm{pa}(s)})} \\ &= \frac{\prod_{s\in\mathcal{S}}\prod_{t\in\mathrm{ch}(s)} h_{s\to t}(x_{\mathrm{pa}(t)})}{\prod_{s\in\mathcal{S}} h_{\to s}(x_{\mathrm{pa}(s)})} = \frac{\prod_{v\in\mathcal{V}} h_{\to v}(x_{\mathrm{pa}(v)})}{\prod_{t\in\mathrm{ch}(0)} h_{0\to t}(x_0)},\end{aligned}$$

and the denominator in the final term is easily seen to equal $h_0(x_0)$. The third equality if obtained by replacing x_s by $x_{\mathrm{pa}(t)}$, which is valid since each vertex has a unique parent. The last equality holds since the numerator takes the product over all edges except for those starting from the root, whereas the denominator takes the product over all edges except for those edges that end at a leaf. □

4 Guided processes

4.1 Guided processes on a directed tree

The h-transform can only be computed in closed form in few cases and essentially assumes the graphical model to be a directed tree and even there, in general, it is intractable and so is $p^\star$. Suppose it is possible to find an approximation $\widetilde{h}$ to h, for example by choosing a Gaussian

approximation to X and that computing the h-transform for that approximation is tractable. Then one can use this $\widetilde{h}$ to define an *approximate* Doob's h-transform.

Below we define the *guided process* X° as Doob's h-transform of X, using $\widetilde{h}$ instead of h. To avoid confusion, we wish to make clear that there are three processes on $\mathcal{G}$ involved:

- X, the unconditional process;
- $X^\star$, the process conditioned by the values on its leaves;
- X°, the guided process resembling $X^\star$ but with h replaced by $\widetilde{h}$.

The idea is that X° provides an approximate solution to the smoothing problem such that the law of $X^\star$ is absolutely continuous with respect to the law of X°, and where X° can be sampled forwards recursively from the root. Subsequently, we can correct for the discrepancy between h and $\widetilde{h}$ (and accordingly the discrepancy between $X^\star$ and X°) using weighted samples, thereby sampling from the exact smoothing distribution. A key observation is that the proof of Theorem 3.2 crucially depends on the relation in equation (3.3). That is, once $\widetilde{h}$ is defined on the edges and it is defined at the vertices by this relation, there will be a cancellation of terms in a likelihood ratio of the form as in Theorem 3.2. This applies in particular if $\widetilde{h}$ is obtained as h-transform of a simple, tractable process $\widetilde{X}$ with transition densities $\widetilde{p}$ approximating the transition densities p of X.

We make the following assumption on $\widetilde{h}$.

Assumption 4.1. The maps $x \mapsto \widetilde{h}_{s\to t}(x)$ are specified for each edge (s,t) in $\mathcal{T}$.

Corresponding to (3.3) define for each $s \in \mathcal{S}_0$ fusion of $\widetilde{h}$ by

$$\widetilde{h}_s(x) = \prod_{t\in \mathrm{ch}(s)} \widetilde{h}_{s\to t}(x). \tag{4.1}$$

Definition 4.2. Define the guided process as the Markov process X° starting in $X_0^\circ = 0$ and evolving from the root onwards according to transition densities

$$p^\circ_{\to s}(x;y) = w_{\to s}(x)^{-1} p_{\to s}(x;y) \frac{\widetilde{h}_s(y)}{\widetilde{h}_{\to s}(x)}, \qquad s \in \mathcal{S}, \tag{4.2}$$

with

$$w_{\to s}(x) = \frac{\int p_{\to s}(x;y)\widetilde{h}_s(y)\lambda(\mathrm{d}y)}{\widetilde{h}_{\to s}(x)}$$

featuring as *weights* below.

Replacing h with $\widetilde{h}$, rather than approximating the dynamics of the conditioned process, we aim to approximate the information brought by future observations, and let this approximation guide the process in a natural way. In absence of information from observations, the process evolves just as the unconditional one. This is emphatically the right thing to do to preserve absolute continuity when the graph becomes large. In this regard, our approach is different from for example Ambrogioni et al. (2021). This also explains that guided processes were initially introduced for diffusion processes, that is in the infinite dimensional setting of the later section 9, to yield absolutely continuous approximations to the conditional distribution, even in absence of regularising noise (Schauer et al. (2017), Bierkens et al. (2020)). Suda (2020) gives further evidence to that by investigating the stability of guided processes for diffusions in case the interval between observations becomes large.

Theorem 4.3. For $(x_s,\, s \in \mathcal{T})$ we have

$$\prod_{s\in\mathcal{S}} \frac{p^{\star}_{\to s}(x_{\mathrm{pa}(s)}; x_s)}{p^{\circ}_{\to s}(x_{\mathrm{pa}(s)}; x_s)} = \frac{\widetilde{h}_0(x_0)}{h_0(x_0)} \prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}) \prod_{v\in\mathcal{V}} \frac{h_{\to v}(x_{\mathrm{pa}(v)})}{\widetilde{h}_{\to v}(x_{\mathrm{pa}(v)})}.$$

Proof. The proof is a special case of the proof for the more general Theorem 4.6 ahead. □

If transitions to the leaves are tractable by themselves and $\widetilde{h}_t$ is accordingly chosen to match h_t at these transitions, the likelihood ratio simplifies:

Corollary 4.4. Assume that for all $v \in \mathcal{V}$, $\widetilde{h}_{\to v}(x) = h_{\to v}(x)$. Then

$$\prod_{s\in\mathcal{S}} \frac{p^{\star}_{\to s}(x_{\mathrm{pa}(s)}; x_s)}{p^{\circ}_{\to s}(x_{\mathrm{pa}(s)}; x_s)} = \frac{\widetilde{h}_0(x_0)}{h_0(x_0)} \prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}).$$

While h_0 is generally intractable it only shows up in the denominator of the Radon-Nikodym derivative as a multiplicative constant. Hence, Corollary 4.4 implies that the evidence satisfies

$$h_0(x_0) = \widetilde{h}_0(x_0)\, \mathbb{E} \prod_{s\in\mathcal{S}} w_{\to s}(X^{\circ}_{\mathrm{pa}(s)}). \tag{4.3}$$

4.2 Guided processes on a directed acyclic graph

In this section we drop the assumption that $\mathcal{T}$ is a directed tree and assume it to be a directed acyclic graph (DAG), making X a Bayesian network. Note that a Markov process on a DAG – by grouping vertices to certain overlapping groups – gives rise to a Markov process on a tree of grouped vertices, for which one could define an h-transform and appropriate $\widetilde{h}$-transform. This is not what we consider in this section, here we develop an approach which works directly on the DAG.

We still assume the setting from section 2, in particular that the process X is determined by its formal starting point X_0 and its transition densities $p_{\to t}(x_{\mathrm{pa}(t)}; x_t)$ (now for a set of parents) and that the joint density is

$$p(x_{\mathcal{T}}) = \prod_{t\in\mathcal{T}} p_{\to t}(x_{\mathrm{pa}(t)}; x_t)$$

Note that on a DAG, contrary to a directed tree, conditioning on the values at leaf-vertices changes the dependency structure. In the following definition we define the guided process such that it resembles the recursive structure of the h-transform on a tree. However, unlike the conditioned process the guided process has the same dependency structure as the (unconditional) forward process.

Definition 4.5. We define the guided process X° on a DAG as the process starting in $X^{\circ}_0 = x_0$ and from the roots onwards evolving according to transition densities

$$p^{\circ}_{\to s}(x_{pa(s)}; y) = w_{\to s}(x_{\mathrm{pa}(s)})^{-1} p_{\to s}(x_{pa(s)}; y) \frac{\widetilde{h}_s(y)}{\prod_{u\in\mathrm{pa}(s)} \widetilde{h}_{u\to s}(x_u)}, \tag{4.4}$$

where $s \in \mathcal{S}$ and

$$w_{\to s}(x_{\mathrm{pa}(s)}) = \frac{\int p_{\to s}(x_{pa(s)}; y)\widetilde{h}_s(y)\lambda(\mathrm{d}y)}{\prod_{u\in\mathrm{pa}(s)} \widetilde{h}_{u\to s}(x_u)}. \tag{4.5}$$

Note that this definition contains definition 4.2 as a special case.

The following is the analogue of Theorem 4.3 for a DAG.

Theorem 4.6. For $(x_s,\, s \in \mathcal{T})$ we have

$$\frac{p^\star(x_{\mathcal{S}} \mid x_0)}{\prod_{s\in\mathcal{S}} p^\circ_{\to s}(x_{\mathrm{pa}(s)}; x_s)} = \frac{\widetilde{h}_0(x_0)}{h_0(x_0)} \prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}) \prod_{v\in\mathcal{V}} \frac{h_{\to v}(x_{\mathrm{pa}(v)})}{\widetilde{h}_{\to v}(x_{\mathrm{pa}(v)})}.$$

Proof. First note that

$$\prod_{s\in\mathcal{S}} \frac{p_{\to s}(x_{\mathrm{pa}(s)}; x_s)}{p^\circ_{\to s}(x_{\mathrm{pa}(s)}; x_s)} = \prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}) \times \prod_{s\in\mathcal{S}} \frac{\widetilde{h}_s(x_s)}{\prod_{u\in\mathrm{pa}(s)} \widetilde{h}_{u\to s}(x_u)}.$$

The second term on the right-hand-side can by (4.1) be rewritten as

$$\prod_{s\in\mathcal{S}} \frac{\prod_{t\in\mathrm{ch}(s)} \widetilde{h}_{s\to t}(x_s)}{\prod_{u\in\mathrm{pa}(s)} \widetilde{h}_{u\to s}(x_u)}.$$

Now, similar as in the proof of Theorem 3.2 we notice cancellation of terms so that we can conclude that

$$\prod_{s\in\mathcal{S}} \frac{p_{\to s}(x_{\mathrm{pa}(s)}; x_s)}{p^\circ_{\to s}(x_{\mathrm{pa}(s)}; x_s)} = \prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}) \times \frac{\prod_{v\in\mathcal{V}} \widetilde{h}_{\to v}(x_{\mathrm{pa}(v)})}{\prod_{t\in\mathrm{ch}(0)} \widetilde{h}_{0\to t}(x_0)}.$$

The denominator of the second term on the right-hand-side equals $\widetilde{h}_0(x_0)$.

The result now follows upon noting that

$$p^\star(x_{\mathcal{S}} \mid x_0) = \prod_{s\in\mathcal{S}} p_{\to s}(x_{\mathrm{pa}(s)}; x_s) \times \frac{\prod_{v\in\mathcal{V}} h_{\to v}(x_{\mathrm{pa}(v)})}{h_0(x_0)}$$

and combining the preceding two displayed equations. □

Theorem 4.6 leaves open the choice of $\widetilde{h}_{s\to t}$ which then determines $\widetilde{h}_s$ by (4.1). Two approaches for this are as follows:

- Define a process $\widetilde{X}$ on a directed subtree of the DAG such that it approximates X in law on vertices of this subtree. The $\widetilde{h}$-transform for $\widetilde{X}$ is then defined and gives $\widetilde{h}_{s\to t}$ for all edges of the subtree. For all remaining edges (s,t) of the DAG we set $\widetilde{h}_{s\to t} = 1$. This effectively means that while the forward process is a Bayesian net described by a DAG, it is backward filtered on a directed sub-tree.

- Define

  $$\widetilde{h}_{\mathrm{pa}(t)\to t}(x_{\mathrm{pa}(t)}) := \int \widetilde{p}_{\to t}(x_{\mathrm{pa}(t)}; y)\widetilde{h}_t(y)\lambda(\mathrm{d}y).$$

  Choose, starting from the leaves, $\widetilde{h}_{s\to t}$ such that

  $$\prod_{s\in\mathrm{pa}(t)} \widetilde{h}_{s\to t}(x_s) \approx \widetilde{h}_{\mathrm{pa}(t)\to t}(x_{\mathrm{pa}(t)}).$$

  We will refer to this procedure as *backward marginalisation*. We provide a detailed description in subsection 5.4.

5 The computational structure of Doob's h-transform and guided processes

In section 3 we explained how Doob's h-transform on a directed tree can be obtained backwards recursively starting from the leaves back to the root. Subsequently, the conditional process $X^\star$ can be simulated forwards. In section 4 this was extended in two directions: *(i)* use of an approximate h-transform, in case the actual h-transform is intractable; *(ii)* graphical models described by a DAG, instead of a directed tree.

The resulting computational algorithm consists of a backward filtering step, followed by a forward guiding step. Here, we say *guiding*, as the guided process X° is simulated just like X, though with adjusted dynamics that guide the process to the observations at the leaves.

Depending on the structure of the DAG, a full implementation of this algorithm can easily become complex. We aim to describing how to assemble it from simple parts to facilitate an efficient and possibly automatic implementation. A key ingredient to this is to convert the graphical model to a *string diagram*, well known in category theory. The actual categorical description of the algorithm is postponed to section 6.

5.1 Compositional structure of Doob's h-transform on a line graph

We first give a principle of compositionality for the task of sampling from $X^\star$. For illustrative purposes we consider the simple case of a line graph, i.e. the setting where $X_0, X_1, \dots, X_n, X_{n+1}$ is a Markov chain initialised deterministically with $X_0 = x_0$. This simple setting will allow to formally introduce many concepts required for the more general case. In the left part of Figure 2 the corresponding DAG (a line graph) for $n = 3$ is depicted and each node corresponds to a random variable X_i omitting x_0. Central for the dynamics are the Markov transition kernels

$$\kappa_i(x, \mathrm{d}y) = p_{\to i}(x; y)\lambda(\mathrm{d}y), \qquad i \in \mathcal{T}.$$

We shall assume κ_i is a kernel with "source" the measurable space $(E_{i-1}, \mathfrak{B}_{i-1})$ and "target" the measurable space $(E_i, \mathfrak{B}_i)$.[2] To simplify notation we will write

$$S_i = (E_i, \mathfrak{B}_i).$$

and $\kappa_i \colon S_{i-1} \twoheadrightarrow S_i$ (note the special type of arrow).

The right part of the figure shows the corresponding string diagram (see Chapter 3.1 in Jacobs (2019) or Selinger (2011)), which is to be read from bottom to top. The string diagram emphasises the compositional nature of forward sampling. Here, the boxes represent the Markov kernels κ_i.

Each kernel κ_i induces a push forward of a measure μ on S_{i-1} to S_i via

$$\mu\kappa_i(\cdot) = \int_E \kappa_i(x, \cdot)\mu(\mathrm{d}x). \tag{5.1}$$

For a measurable space S denote by $\mathcal{M}(S)$ the set of bounded measures on S. Then if $\mu_i \in \mathcal{M}(S_i)$, we can iteratively define $\mu_i = \mu_{i-1}\kappa_i$.

The following pseudo-program uses the iterative structure to compute the marginal distribution of $X_1, \dots, X_4$ by pushing forward and returning the marginal distribution of X_4.

```
function kernel1234(mu0)
    mu1 = kernel1(mu0)
```

[2] That is, $\kappa_i \colon E_{i-1} \times \mathfrak{B}_i \to [0, 1]$, where for fixed $B \in \mathfrak{B}_i$ the map $x \mapsto \kappa_i(x, B)$ is $\mathfrak{B}_{i-1}$-measurable and for fixed $x \in E_{i-1}$, the map $B \mapsto \kappa_i(x, B)$ is a probability measure on $(E_i, \mathfrak{B}_i)$.

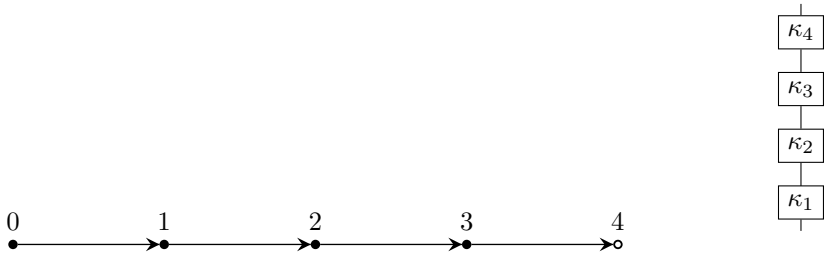


FIGURE 2: Left: a very simple line graph with one leaf and $n = 3$. Each vertex represents a variable. Right: the corresponding string diagram where each box represents a kernel.

```
    mu2 = kernel2(mu1)
    mu3 = kernel3(mu2)
    mu4 = kernel4(mu3)
    return mu4
end
```

Markov kernels $\kappa_1 \colon S_0 \to S_1$ and $\kappa_2 \colon S_1 \to S_2$ can be composed by the Chapman-Kolmogorov equations, here written as juxtaposition

$$(\kappa_1\kappa_2)(x_0, \cdot) = \int_{E_1} \kappa_2(x_1, \cdot)\kappa_1(x_0, \mathrm{d}x_1), \qquad x_0 \in E_0. \tag{5.2}$$

The unit id for this composition is the identity function considered as a Markov kernel, $\mathrm{id}(x, \mathrm{d}y) = \delta_x(\mathrm{d}y)$. Consequently the distribution of X_i is given by a composition of kernels $\delta_{x_0}\kappa_1\kappa_2 \cdots \kappa_i$, obtained by repeatedly pushing forward the initial law (a Dirac measure in x_0 as we assumed deterministic $X_0 = x_0$). Parentheses can be omitted by associativity. Note that the function `kernel1234` corresponds to a Markov kernel itself, $\kappa_{1234} = \kappa_1\kappa_2\kappa_3\kappa_4$.

For computing the backward recursion of Doob's h-transform given observation $X_{n+1} = x_{n+1}$, assume that X_{n+1} has a distribution with a λ-density and define, recalling (3.1),

$$h_n(x) = \kappa_{n+1}(x, \mathrm{d}y)/\lambda(\mathrm{d}y) \,|_{y=x_{n+1}} \tag{5.3}$$

and

$$h_{i-1}(x) = \int h_i(y)\kappa_i(x, \mathrm{d}y), \qquad i = 1, \ldots, n. \tag{5.4}$$

If we denote by $\mathbf{B}(S)$ the Banach space of bounded measurable real valued functions on the measurable space $S = (E, \mathfrak{B})$ (equipped with the supremum norm), then this can be written as $h_{i-1} = \kappa_i h_i$ by defining the linear continuous operator $\kappa \colon \mathbf{B}(S') \to \mathbf{B}(S)$ by

$$\kappa h(\cdot) = \int_E h(y)\kappa(\cdot, \mathrm{d}y), \qquad h \in \mathbf{B}(S'). \tag{5.5}$$

By repeatedly applying this map, we obtain $h_0 = \kappa_1\kappa_2 \cdots \kappa_n h_n$.

The abstract h-transform maps (κ_i, h_i) to $\kappa_i^\star$ with h_i computed beforehand, where the conditional kernel is defined through its Radon-Nikodym derivative. For $(\kappa, h) \in \{(\kappa_1, h_1), \ldots, (\kappa_n, h_n)\}$,

$$\frac{\kappa^\star(x, \mathrm{d}y)}{\kappa(x, \mathrm{d}y)} = \frac{h(y)}{(\kappa h)(x)} =: m(x, y) \tag{5.6}$$

(not for $\kappa^\star_{n+1}$, which is just a Dirac measure in the observation). Note that equivalently $\kappa^\star$ is characterised by the relation

$$\frac{\kappa f h}{\kappa h} = \kappa^\star f, \tag{5.7}$$

for bounded measurable test functions f We will interpret m as a *message* obtained from backward filtering, to be used in forward sampling.

Definition 5.1. For a Markov kernel κ, message m (as defined in (5.6)) and measure μ define the forward map $\mathcal{F}$ by

$$\mathcal{F}_\kappa(m, \mu) = \nu, \quad \nu(\mathrm{d}y) = \int m(x, y)\mu(\mathrm{d}x)\kappa(x, \mathrm{d}y). \tag{5.8}$$

In this definition, we implicitly assume κ, m and μ are compatible. This map requires m as "message" from the backward recursion and pushes forward the measure μ by $\kappa^\star$ (conditional forward evolution) via Doob's h-transform. Note that the forward map does not necessarily return a probability measure. It does when μ is a probability measure and m is as in (5.6).

With foresight we define the corresponding backward map (assuming implicitly compatibility of κ and h):

Definition 5.2. For a Markov kernel κ and function h defined the backward map by

$$\mathcal{B}_\kappa(h) = (m, \kappa h), \quad \text{where} \quad m(x, y) = \frac{h(y)}{(\kappa h)(x)}. \tag{5.9}$$

This map returns both the appropriate message m for $\mathcal{F}_\kappa$ and the "pullback" κh for the backward recursion.

A first form of compositionality is seen from iteratively apply the forward and backward maps in the following way:

$$\begin{aligned} m_i, h_{i-1} &= \mathcal{B}_{\kappa_i}(h_i), \qquad i = 1, \ldots, n \\ \mu_i &= \mathcal{F}_{\kappa_i}(m_i, \mu_{i-1}) \qquad i = 1, \ldots, n, \end{aligned} \tag{5.10}$$

initialised from h_n and μ_0.

Remark 5.3. The message m from $\mathcal{B}_\kappa$ could equivalently be transmitted as pair $(h, \kappa h)$ and then combined in the recipient $\mathcal{F}_\kappa$, or even just as h, with κh computed in $\mathcal{F}_\kappa$. We choose the quotient form as it is convenient for the exposition and acknowledges that while other choices are sometimes convenient, they are essentially equivalent: the format of the message m is something between $\mathcal{F}_\kappa$ and $\mathcal{B}_\kappa$. Also note that the forward map is invariant under multiplying h by an arbitrary strictly positive constant c, as the constant is cancelled in the message m.

Figure 3 illustrates the structure which reminds of a physical zipper and illustrates how the output of $\mathcal{B}_\kappa$ serves as input for $\mathcal{F}_\kappa$. The diagram should be read starting from the very right, where the observation from leaf 4 comes in via h_3. Then the h-transform is computed back to the root by moving south-west towards h_0. Subsequently, the conditional marginal is propagated from μ_0 onwards by moving in north-west direction. Simulation of the conditioned process follows the same pattern. The diagram reveals the dependency structure and recursive nature of the algorithm for computing the h-transform.

A corresponding pseudo-program in unrolled form, consisting of a function accepting h_3 and μ_0 and computing the marginals of the smoothing distribution, has a similar structure:

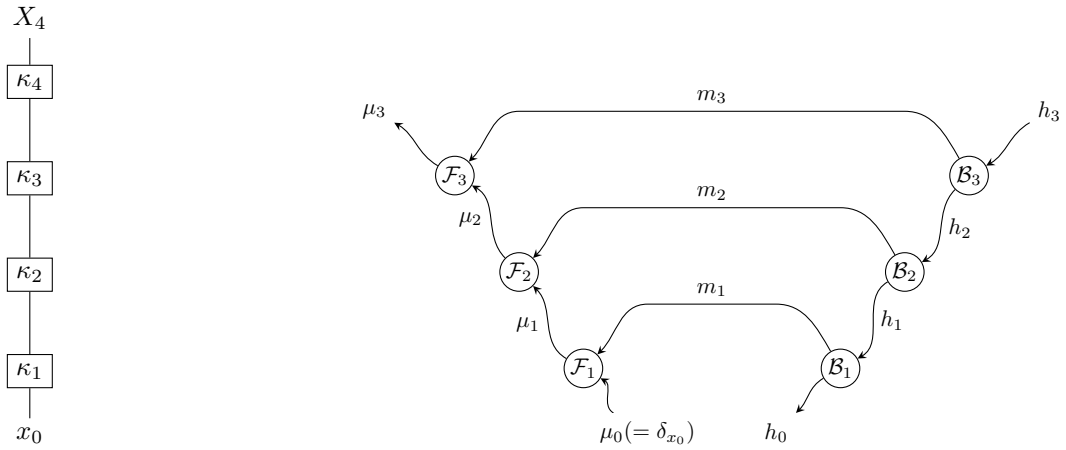


FIGURE 3: Left: String diagram of Markov kernels for a Markov process x_0, X_1, $\dots X_4$. Right: Corresponding diagram illustrating passing of arguments between backward operators $\mathcal{B}_i$ and forward operators $\mathcal{F}_i$ during backward filtering, forward smoothing for the process observed at the final state X_4. Algorithmic time runs from right to left, starting from h_3 determined by x_4 in the upper right corner. The messages m_i are passed between backward pass and forward pass from right to left.

```
function transform123(mu0, h3)
    message3, h2 = backward(kernel3, h3)
    message2, h1 = backward(kernel2, h2)
    message1, h0 = backward(kernel1, h1)

    mu1 = forward(kernel1, message1, mu0)
    mu2 = forward(kernel2, message2, mu1)
    mu3 = forward(kernel3, message3, mu2)

    return h0, mu1, mu2, mu3 # return evidence and marginals
end
```

Clearly, $\mathcal{F}_\kappa$ and $\mathcal{B}_\kappa$ work as pairs which are therefore denoted formally as $\langle \mathcal{F}_\kappa \mid \mathcal{B}_\kappa \rangle$. Then we can write the transformation of kernel to the pair of forward and backward map as

$$F(\kappa) = \langle \mathcal{F}_\kappa \mid \mathcal{B}_\kappa \rangle. \tag{5.11}$$

Instead of pushing forward probability measures μ, we can forward propagate a sample x via $\mathcal{F}_\kappa(m, \delta_x)$.

Definition 5.4. We define $\mathcal{F}^\star_\kappa(m, x)$ as a random variable with distribution $\mathcal{F}_\kappa(m, \delta_x)$, giving a sample from the transition kernel $\kappa^\star(x, \cdot)$.

This entails that for $\kappa\colon (E, \mathfrak{B}) \twoheadrightarrow (E', \mathfrak{B}')$, $\mathcal{F}^\star_\kappa(m, x)$ is a E'-valued random variable parametrised by $(m, x) \in M \times E$ where $M \times \mathbf{B}$ is the range of $\mathcal{B}_\kappa$. A program may use several random variables of the form $\mathcal{F}^\star_\kappa(m, x)$. We then assume their independence.

The following randomisation lemma gives $\mathcal{F}^\star_\kappa$ a functional form and clarifies how to draw from a kernel κ or $\kappa^\star$:

Lemma 5.5 (Kallenberg (2002), p. 56)**.** For a Markov kernel κ between a measure space $(E, \mathfrak{B})$ and a Borel space $(F, \mathfrak{C})$, there is a measurable function $f_\kappa \colon E \times [0,1]$ and a random variable $Z \sim U[0,1]$ such that the random variable $f_\kappa(x, Z)$ has law $\kappa(x, \cdot)$ for each $x \in E$. The random variable Z is called an *innovation variable*.

Further, if κ' is a kernel with source $(F, \mathfrak{C})$ into a third Borel space, there exist independent random variables Z, $Z' \sim U[0,1]$ such that $f_{\kappa'}(f_\kappa(x, Z), Z') \sim (\kappa\kappa')(x, \cdot)$.

5.2 The computational structure on a line graph for guided processes

In case $\widetilde{h}$ is derived from filtering an (auxiliary) process $\widetilde{X}$ with transition densities $\widetilde{p}$, then this effectively means we use a kernel $\widetilde{\kappa}$ instead of κ. As κ, $\widetilde{\kappa}$ distinguish the kernels used in the backward forward and backward pass, writing $\widetilde{h}$ is superfluous, the tilde can be dropped, and for that reason we will simply write h.

For a guided process, the pair of forward and backward maps can then be written as

$$\langle \mathcal{F}_\kappa \mid \mathcal{B}_{\widetilde{\kappa}} \rangle.$$

If μ is a probability measure and $\mathcal{B}_\kappa$ sends the message m, then $\mathcal{F}_\kappa(m, \mu)$ is again a probability measure. If instead, as for guided processes, $\mathcal{B}_{\widetilde{\kappa}}$ sends the message m (rather than $\mathcal{B}_\kappa$), then $\mathcal{F}_\kappa(m, \mu)$ need not be a probability measure, even if μ is. For this reason, we need to assume $\mathcal{F}$ is applied to the bounded measure $\varpi \cdot \mu$, where ϖ is such that μ is a probability measure and $\cdot$ denotes the scaling of a measure with a scalar. Then if the message m is given by $m(x, y) = h(y)/\widetilde{\kappa}h(x)$, we have

$$\mathcal{F}_\kappa(m, \varpi \cdot \mu)(\mathrm{d}y) = \varpi \int m(x, y)\mu(\mathrm{d}x)\kappa(x, \mathrm{d}y)$$

by definition of $\mathcal{F}_\kappa$ (cf. definition 5.1). The right hand side can then again be written as weighted probability measure,

$$\mathcal{F}_\kappa(m, \varpi \cdot \mu) = w_\kappa(m, \mu)\varpi \cdot \nu,$$

with

$$\nu(\mathrm{d}y) = w_\kappa^{-1}(m, \mu) \int \frac{h(y)}{(\widetilde{\kappa}h)(x)}\mu(\mathrm{d}x)\kappa(x, \mathrm{d}y) \quad \text{and} \quad w_\kappa(m, \mu) = \int \frac{(\kappa h)(x)}{(\widetilde{\kappa}h)(x)}\mu(\mathrm{d}x). \tag{5.12}$$

Note that $w_\kappa(m, \mu)$ depends on the backward map only via the message m, and that ν is a probability measure.

Crucially $\langle \mathcal{F}_\kappa \mid \mathcal{B}_{\widetilde{\kappa}} \rangle$ characterises the conditional distribution: With say (5.7), we have for a kernel $\kappa \colon S \twoheadrightarrow S'$ and an operator $\widetilde{\kappa} \colon \mathbf{B}(S') \to \mathbf{B}(S)$,

$$\begin{aligned} \int f(y)\mathcal{F}_\kappa(m, \varpi, \delta_{x_0})\,\mathrm{d}y &= \varpi \int f(y)m(x, y)\kappa(x, \mathrm{d}y)\delta_{x_0}(\mathrm{d}x) \\ &= \frac{\varpi}{(\widetilde{\kappa}h)(x_0)} \int f(y)h(y)\kappa(x_0, \mathrm{d}y) \\ &= \varpi \frac{(\kappa f h)(x_0)}{(\widetilde{\kappa}h)(x_0)} = \left(\frac{\kappa f h}{\kappa h}\right)(x_0), \end{aligned} \tag{5.13}$$

where the final equality follows upon taking $\varpi = \frac{\widetilde{\kappa}h}{\kappa h}(x_0)$.

This holds even if $\widetilde{\kappa}$ is not an operator $\mathbf{B}(S') \to \mathbf{B}(S)$ which comes from a Markov kernel via (5.5). The important property is that the numerator and denominator of messages m_1 and m_2 cancel as in the next lemma.

Lemma 5.6. If $\kappa, \widetilde{\kappa}\colon S_0 \twoheadrightarrow S_2$ where $\kappa = \kappa_1\kappa_2$ and $\widetilde{\kappa} = \widetilde{\kappa}_1\widetilde{\kappa}_2$ are the compositions of kernels to and from an intermediate space S_1,

$$(\kappa f h)(x) = (\widetilde{\kappa} h)(x) \int \left(\int f(z) m_2(y,z) \kappa_2(y, \mathrm{d}z) \right) m_1(x,y) \kappa_1(x, \mathrm{d}y) \quad \forall f \in \mathbf{B}(S_2)$$

for

$$m_1(x,y) = \frac{h_1(y)}{(\widetilde{\kappa}_1 h_1)(x)}, \qquad m_2(y,z) = \frac{h(z)}{(\widetilde{\kappa}_2 h)(y)},$$

where $h_1 = \widetilde{\kappa}_2 h$.

Proof. Plugging in m_1, m_2 and h_1, the right-hand-side is seen to be equal to

$$\int \int f(x) \frac{h(z)}{(\widetilde{\kappa}_2 h)(y)} \kappa_2(y, \mathrm{d}z) \frac{(\widetilde{\kappa}_2 h)(y)}{(\widetilde{\kappa}_1 h_1)(x)} \kappa_1(x, \mathrm{d}y).$$

The results now follows upon integrating out y. □

Instead of pushing forward the measures $\varpi \cdot \mu$, a weighted sample can be forward propagated. If x is a weighted sample from μ with weight ϖ, we can apply $\mathcal{F}_\kappa(m, \cdot)$ to $\varpi \cdot \delta_x$ to obtain a new weight $\varpi w_\kappa(m, \delta_x)$ and new distribution ν to sample from.

Definition 5.7. Define the random map

$$\mathcal{F}^\circ_\kappa(m, (\varpi, x)) = (\varpi w_\kappa(m, \delta_x), Y), \quad Y \sim \nu, \tag{5.14}$$

where ν is the probability measure $\mathcal{F}_\kappa(m, \delta_x)/w_\kappa(m, \delta_x)$ (i.e. $\mathcal{F}_\kappa(m, \delta_x)$ normalised by $w_\kappa(m, \delta_x)$).

This entails that Y is a $[0, \infty) \times E'$-valued random variable parametrised by $(m, \varpi, x) \in M \times [0, \infty) \times E$. We again require independence of the $\mathcal{F}^\circ_\kappa(m, (\varpi, x))$ used in a single diagram.

On a line graph, also the compositional structure for a guided process is the same as for Doob's h-transform (as in equation (5.10)); all that is done is replacing κ_i in the backward map by $\widetilde{\kappa}_i$.

5.3 Forward and backward operators on a tree

Up to this point, we have only considered the simple case of a line graph. In this section we aim to extend the composition ideas to a tree. A first step is to turn a tree into a string diagram. As we will shortly see, this first requires considering both product kernels and the duplication kernel. Subsequently, we study how F as defined in equation (5.11) acts on these. Figure 4 gives an example of a directed tree. At the vertex labeled 1 the process branches, leading to a leaf vertex $2b$ and another vertex, $2a$. At this vertex the process branches once more, leading to leaf vertices $3a$ and $3b$. It follows from our assumptions that after branching the evolution on the branches is independent, conditional on the value of the common parent vertex. We now make the concepts "branching" and "evolving conditionally independent" precise using the duplication kernel and product kernels respectively.

5.3.1 Product kernels and marginalisation

Given measurable spaces $S = (E, \mathfrak{B})$, $T = (F, \mathfrak{C})$, $S' = (E', \mathfrak{B}')$ and $T' = (F', \mathfrak{C}')$ consider kernels $\kappa\colon S \twoheadrightarrow T$ and $\kappa'\colon S' \twoheadrightarrow T'$. The product Markov kernel $\kappa \otimes \kappa'$ on $S \otimes S'$ is defined on the cylinder sets by

$$(\kappa \otimes \kappa')((x, x'), B \times B') = \kappa(x, B)\kappa'(x', B'),$$

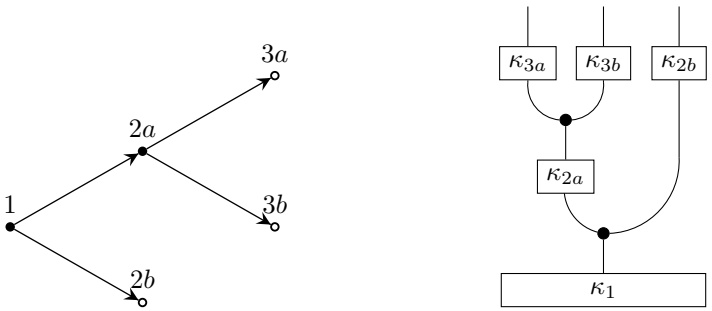


FIGURE 4: A simple DAG with three leaves and the corresponding string diagram with 3 observables reaching the upper figure boundary. The root 0 is not drawn. The black bullet symbol denotes the duplication (copy) Δ.

(where $x \in E$, $x' \in E'$, $B \in \mathfrak{C}$, $B' \in \mathfrak{C}'$) and then extended to a kernel on the product measure space.

If κ_{n+1} of product form, $\kappa_{n+1} = \kappa_{(n+1)a} \otimes \kappa_{(n+1)b}$ we now allow for observation of only $X_{(n+1)a} = x_{(n+1)a}$ (say) and set

$$h_n(x) = \kappa_{(n+1)a}(x, \mathrm{d}y)/\lambda(\mathrm{d}y) \mid_{y=x_{(n+1)a}} . \tag{5.15}$$

This facilitates to reason about the distribution of intermediate states $X_2, \cdots, X_n$ by making a duplicate of them and making them part of X_{n+1}.

Product kernels preserve product measures:

$$(\mu \otimes \mu')(\kappa \otimes \kappa') = (\mu\kappa) \otimes (\mu'\kappa').$$

Note that if $\mu \in \mathcal{M}(S \otimes S')$ and ν is defined by $\nu = \mu(\kappa \otimes \kappa')$, then ν is not necessarily a product measure.

The product structure simplifies sampling from $\nu = \mu(\kappa \otimes \kappa')$ using samples (X, X') from μ. Here X and X' are not necessarily independent. Then indeed, if Z, Z' are independent and uniformly distributed on $[0, 1]$, for f_κ, $f_{\kappa'}$ and $f_{\kappa\otimes\kappa'}$ as in Lemma 5.5, we have

$$f_{\kappa\otimes\kappa'}((X, X'), Z) \sim \mu(\kappa \otimes \kappa')$$

and *also*

$$(f_\kappa(X, Z), f_{\kappa'}(X', Z')) \sim \mu(\kappa \otimes \kappa').$$

The product structure also simplifies the computation of the marginals of ν (cf. Jacobs and Zanasi (2020)).

Proposition 5.8. Let M be the marginalisation operator acting on measures $\mu \in \mathcal{M}(S \otimes S')$ by (using postfix notation for M)

$$(\mu M)(B \times B') = \frac{\int \mu(B, \mathrm{d}x')}{\sqrt{\int \mathrm{d}\mu}} \otimes \frac{\int \mu(\mathrm{d}x, B')}{\sqrt{\int \mathrm{d}\mu}}, \qquad B \in \mathfrak{B}, B' \in \mathfrak{B}'.$$

Then for Markov kernels $\kappa\colon S \to T$ and $\kappa'\colon S' \to T'$

$$\mu(\kappa \otimes \kappa')M = \mu M(\kappa \otimes \kappa').$$

Proof. See appendix B. □

The product structure for Markov kernels extends to their application as pullbacks. For $h \in \mathbf{B}(S), h' \in \mathbf{B}(S')$, define the pointwise product

$$(h \odot h')(x, y) = h(x) \cdot h'(y), \qquad x \in E, y \in E'.$$

Define the corresponding pointwise product for messages by

$$(m \mathbin{\widetilde{\odot}} m')((x, x'), (y, y')) = m(x, y) \cdot m(x', y').$$

We occasionally use $\otimes$ as postfix operator (as it acts on products of measures) and $\odot$ as prefix operator (as it acts on functions.) The following lemma shows product rules for the forward and backward map obtained from a product of kernels.

Lemma 5.9. Let $h \in \mathbf{B}(T)$ $h' \in \mathbf{B}(T')$ and assume Markov kernels $\kappa\colon S \nrightarrow T$, $\kappa\colon S' \nrightarrow T'$. Suppose $\mu \in \mathcal{M}(S)$, $\mu' \in \mathcal{M}(S')$, $m \in \mathbf{B}(S \otimes T)$ and $m' \in \mathbf{B}(S' \otimes T')$. Then

$$\begin{aligned}(\kappa \otimes \kappa')(h \odot h') &= (\kappa h) \odot (\kappa' h') \\ \mathcal{B}_{\kappa\otimes\kappa'}(h \odot h') &= \Phi(\mathcal{B}_\kappa(h), \mathcal{B}_{\kappa'}(h')) \\ \mathcal{F}_{\kappa\otimes\kappa'}(m \mathbin{\widetilde{\odot}} m', \mu \otimes \mu') &= \mathcal{F}_\kappa(m, \mu) \otimes \mathcal{F}_{\kappa'}(m', \mu')\end{aligned}$$

where $\Phi((m, h), (m', h')) = (m \mathbin{\widetilde{\odot}} m', h \odot h')$.

Proof. The first statement follows from Fubini's theorem. If

$$m(x, y) = \frac{h(y)}{(\kappa h)(x)} \quad \text{and} \quad m'(x, y) = \frac{h'(y)}{(\kappa' h)(x)},$$

then

$$\mathcal{B}_{\kappa\otimes\kappa'}(h \odot h') = (m \mathbin{\widetilde{\odot}} m', (\kappa \otimes \kappa')(h \odot h')) = (m \mathbin{\widetilde{\odot}} m', (\kappa h) \odot (\kappa' h')).$$

Further, if $\bar{\nu} = \mathcal{F}_{\kappa\otimes\kappa'}(m \mathbin{\widetilde{\odot}} m', \mu \otimes \mu')$, then

$$\begin{aligned}\bar{\nu}(\mathrm{d}(y, y')) &= \int m(x, y) m(x', y')(\mu \otimes \mu')(\mathrm{d}(x, x'))(\kappa \otimes \kappa')((x, x'), \mathrm{d}(y, y')) \\ &= \nu(\mathrm{d}y)\nu'(\mathrm{d}y') = (\nu \otimes \nu')(\mathrm{d}(y, y')),\end{aligned}$$

where $\nu = \mathcal{F}_\kappa(m, \mu)$ and $\nu' = \mathcal{F}_{\kappa'}(m', \mu')$. □

This result has a striking algebraic nature, and is a first taste of the line of reasoning in section 6, that yields an *implementable* smoothing transformation on DAGs defining backward filtering, forward guiding via elementary transformations and composition rules. There, it is seen that the pairs $\langle \mathcal{F}_\kappa \mid \mathcal{B}_\kappa \rangle$ are instances of optics introduced in category theory (Riley, 2018). We will establish the validity of the composition rule by studying the functorial nature of the map $\kappa \mapsto \langle \mathcal{F}_\kappa^\circ \mid \mathcal{B}_{\bar{\kappa}} \rangle$.

5.3.2 Duplication kernel

The second concept we additionally introduce allows for representing branches. A value is duplicated by the Markov kernel Δ defined by

$$\Delta(x, \mathrm{d}(y_1, y_2)) = \delta_x(\mathrm{d}y_1)\delta_x(\mathrm{d}y_2).$$

This allows two different kernels κ_1, κ_2 to apply independently to the same argument, giving the probability kernel $(\Delta(\kappa_1 \otimes \kappa_2))(x, \cdot) = (\kappa_1 \otimes \kappa_2)((x, x), \cdot)$ representing a conditionally independent transition to two children (as part of a DAG).

Figure 4 illustrates the interplay between Δ and $\otimes$ and the correspondence of string diagrams with DAGs as shown in Figure 1. To the transition densities $p_{\to 1}, p_{\to 2a}$, $p_{\to 2b}$, $p_{\to 3a}$ and $p_{\to 3b}$ correspond four Markov kernels $\kappa_s(x_{\mathrm{pa}(s)}; \mathrm{d}x_s) = p_{\to s}(x_{\mathrm{pa}(s)}; x_s)\lambda(\mathrm{d}x_s)$, $s \in \{1, 2a, 2b, 3a, 3b\}$. The string diagram on the right then illustrates the equivalent composition

$$\kappa_1 \Delta(\kappa_{2a} \otimes \mathrm{id})(\Delta \otimes \mathrm{id})(\kappa_{3a} \otimes \kappa_{3b} \otimes \kappa_{2b}),$$

where the probabilistic evolution of the process is represented by a sequence of kernels and the structure comes from some of these having product form acting independently on parts of the common state. The black bullet symbol in Figure 4 denotes the duplication (copy) Δ. Recall id denotes the identity kernel.

Corollary 5.10. Define the k-fold duplication kernel $\Delta_k(x, \mathrm{d}y) = \delta_x(\mathrm{d}y_1) \dots \delta_x(\mathrm{d}y_k)$. Then

$$\mathcal{B}_{\Delta_k}(h_1 \odot \cdots \odot h_k) = \left(m, \prod_{i=1}^{k} h_i \right), \quad m(x, y) = \frac{(h_1 \odot \cdots \odot h_k)(y)}{\prod_{i=1}^{k} h_i(x)},$$

and

$$\mathcal{F}_{\Delta_k}(m, \mu) = f_{\Delta_k}(\mu),$$

with $f_{\Delta_k}(\mu)$ the image measure of μ under the diagonal map $f_{\Delta_k} \colon x \mapsto (x, \dots, x)$ (k elements). If μ is a probability measure, then so is ν.

Proof. As $(\Delta_k(h_1 \odot \cdots \odot h_k))(x) = \prod_{i=1}^{k} h_i(x)$, the relation for $\mathcal{B}_{\Delta_k}$ follows directly from the definition of the backward map. If $\nu = \mathcal{F}_{\Delta_k}(m, \mu)$ then

$$\begin{aligned} \nu(\mathrm{d}(y_1, \dots, y_k)) &= \int \frac{(h_1 \odot \cdots \odot h_k)(y_1, \dots, y_k)}{(\Delta_k(h_1 \odot \cdots \odot h_k))(x)} \mu(\mathrm{d}x) \Delta_k(x, \mathrm{d}(y_1, \dots, y_k)) \\ &= \int \mu(\mathrm{d}x) \delta_x(\mathrm{d}y_1) \dots \delta_x(\mathrm{d}y_k), \end{aligned}$$

where the second equality follows by cancellation of the terms in numerator and denominator. □

Note in particular that $\Delta_2 \equiv \Delta$ and sampling $\mathcal{F}_{\Delta_k}(hm\mu)$ means drawing z from $\mu / \int \mathrm{d}\mu$ and setting $y_1 = \cdots = y_k = z$ with weight equal to $(\int \mu \, \mathrm{d}x)^{1/k}$.

5.3.3 Forward-backward diagram on a directed tree

Consider the directed tree and string diagram in Figure 5. Assume we backward filter with kernels $\widetilde{\kappa}_i$. Abbreviate $\mathcal{B}_i$ for $\mathcal{B}_{\widetilde{\kappa}_i}$ and $\mathcal{F}_i$ for $\mathcal{F}_{\kappa_i}$. Corresponding to the string-diagram we have the following the following dependency diagram which shows the compositional structure of forward and backward maps for this example:

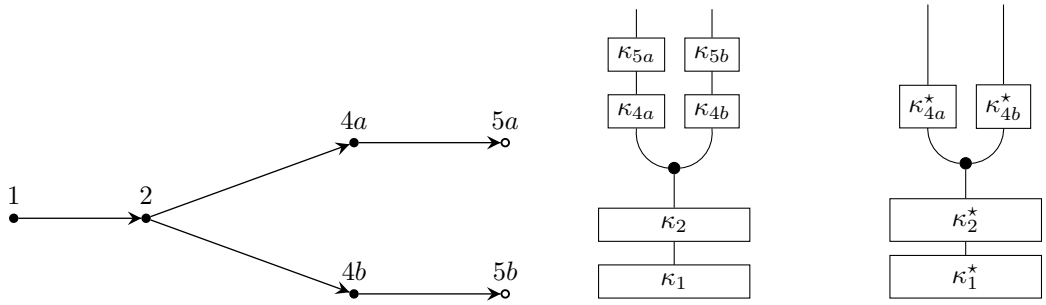
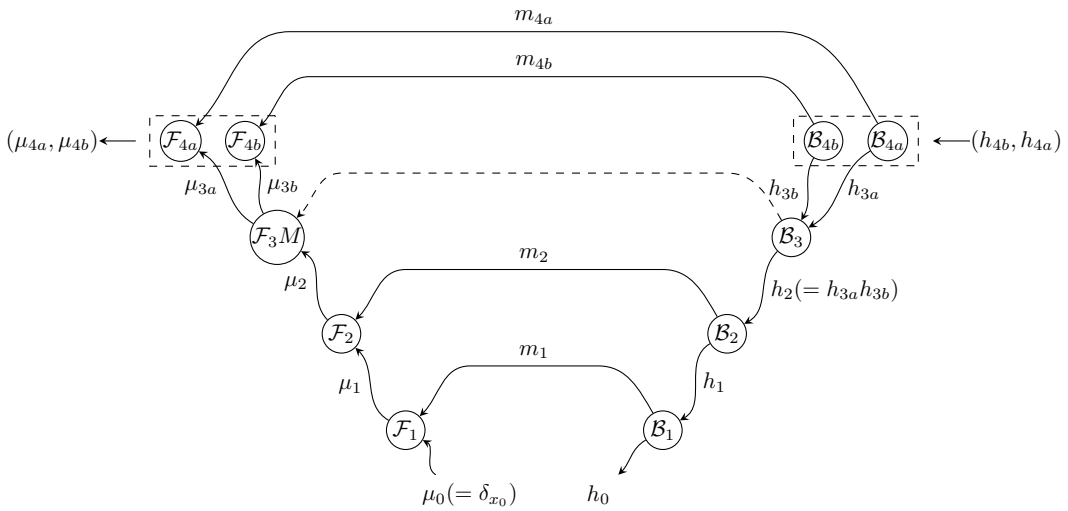


FIGURE 5: A tree with two leaves and vertices numbered according to level in the corresponding string diagram of the composition of $\kappa_1\kappa_2\kappa_3\kappa_4\kappa_5 = \kappa_1\kappa_2\Delta(\kappa_{4a}\otimes\kappa_{4b})(\kappa_{5a}\otimes\kappa_{5b})$. $\kappa^\star_{5a}(x,\cdot) = \delta_{x_{5a}}$ and $\kappa^\star_{5b}(x',\cdot) = \delta_{x_{5b}}$ which are Dirac measures independent of their argument recovering the observations, are not drawn.



(5.16)

This diagram should be read starting from the very right, where observations from the leaves $5a, 5b$ come in via (h_{4a}, h_{4b}). Then the $h/\widetilde{h}$-transform is computed back to the root by moving south-west towards h_0. Subsequently, the conditional marginal is propagated from μ_0 onwards by moving in north-west direction. Simulation of the conditioned process follows the same pattern, but then we would also include the weights explicitly in the diagram. The map $\mathcal{F}_3$ applies marginalisation of μ_2.

Note that if h functions need to be fused, they only need to be sent as message after the fusion step. Duplication is deterministic, so the message is ignored (and doesn't need to be send) in case of the duplication kernel. Here, $\kappa_3 = \Delta$ and hence there is no meaningful message m_3. But the fused message m_2 is sent to $\mathcal{F}_2$.

The corresponding program is as follows:

```
function transformfig5(mu0, h4a, h4b)
    message4a, h3a = backward(kernel4a, h4a)
    message4b, h3b = backward(kernel4b, h4b)
```

```
    _, h2 = backward(kernel3, (h3a, h3b)) # fusion
    message2, h1 = backward(kernel2, h2)
    message1, h0 = backward(kernel1, h1)

    mu1 = forward(kernel1, message1, mu0)
    mu2 = forward(kernel2, message2, mu1)
    mu3 = forward(kernel3, _, mu2) # copy
    mu3a, mu3b = marginals(mu3)
    mu4a = forward(kernel4a, message4a, mu3a)
    mu4b = forward(kernel4b, message4b, mu3b)

    return h0, mu1, mu2, mu3, mu4a, mu4b # return evidence and marginals
end
```

The notation $\mathcal{F}_3 M$ indicates that $\mathcal{F}_3$ is followed by marginalisation with operator M, as defined in Proposition 5.8. In the program we call this `marginals`.

5.3.4 Remarks

Remark 5.11. As we require that $\kappa_{n+1}(x,\cdot)$ has a λ-density for the definition of h_n in (5.3) we choose in Figure 4 (where $n=3$) the composition $\kappa_1\Delta(\kappa_2\otimes\mathrm{id})(\Delta\otimes\mathrm{id})(\kappa_{3a}\otimes\kappa_{3b}\otimes\kappa_{2a})$ using $\kappa_{n+1}=\kappa_{3a}\otimes\kappa_{3b}\otimes\kappa_{2b}$ over the otherwise equivalent composition $\kappa_1\Delta(\kappa_{2a}\otimes\kappa_{2b})(\Delta\otimes\mathrm{id})(\kappa_{3a}\otimes\kappa_{3b}\otimes\mathrm{id})$ using the singular kernel $\kappa_{n+1}=\kappa_{3a}\otimes\kappa_{3b}\otimes\mathrm{id}$.

Remark 5.12. So far we have assumed conditional independent observations. If $X_{n+1}=(Y,Y')$ with $\kappa_{n+1}\colon S_n \to T\otimes T'$ and only $Y'=y'$ was observed, then with $\widetilde{\kappa}\colon S\to T$, $\widetilde{\kappa}'\colon S\to T'$, and distribution of $(\widetilde{Y}_x,\widetilde{Y}'_x)$ given by $\widetilde{\kappa}(x,\cdot)\otimes\widetilde{\kappa}'(x,\cdot)$

$$\mathbb{E}[f(X_{n+1})\mid X_n=x,Y'=y'] = \frac{\mathbb{E}\ f(\widetilde{Y}_x,y')\Phi(x,\widetilde{Y}_x,y')}{\mathbb{E}\ \Phi(x,\widetilde{Y}_x,y')},$$

where

$$\Phi(x,y,y') = \frac{\mathrm{d}\kappa(x,\cdot)}{\mathrm{d}\widetilde{\kappa}(x,\cdot)\otimes\widetilde{\kappa}'(x,\cdot)}(y,y').$$

Thus, (5.3) can be replaced by

$$h_n(x) = \mathbb{E}\ \Phi(x,\widetilde{Y}_x,y'). \tag{5.17}$$

5.4 Forward and backward operators on a DAG with backward marginalisation

In this section we extend our approach to a DAG or its diagrammatic equivalent, as it was done in section 4.2. Whereas on a directed tree each vertex has a unique parent vertex, on a DAG a vertex can have multiple parent vertices. Correspondingly, in the diagram a box with several input edges denotes a Markov kernels whose domain is a tensor product.

Note that if $\kappa=\kappa_1\kappa_2$, then $\kappa^\star=\kappa_1^\star\kappa_2^\star$. Moreover, if $\kappa_2=\kappa_{2a}\otimes\kappa_{2b}$ has product form, we also get

$$(\kappa_1\kappa_2)^\star = (\kappa_1(\kappa_{2a}\otimes\kappa_{2b}))^\star = \kappa_1^\star(\kappa_{2a}^\star\otimes\kappa_{2b}^*).$$

This corresponds to our earlier observation, that the h-transform can be computed in the tree one vertex and its children at a time. It illustrates that the transform as we have considered it until now is structure preserving on a directed tree. This property is lost in case of a general DAG.

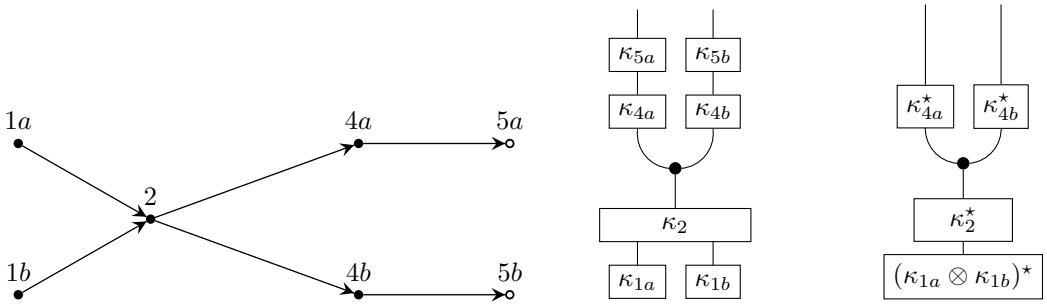


FIGURE 6: A DAG with two leaves and two roots and vertices numbered according to level in the corresponding string diagram of the composition of $\kappa_1\kappa_2\kappa_3\kappa_4\kappa_5 = (\kappa_{1a} \otimes \kappa_{1b})\kappa_2\Delta(\kappa_{4a} \otimes \kappa_{4b})(\kappa_{5a} \otimes \kappa_{5b})$. Conditioning on the end values introduces dependence between roots. $\kappa^\star_{5a}(x, \cdot) = \delta_{x_{5a}}$ and $\kappa^\star_{5b}(x', \cdot) = \delta_{x_{5b}}$ which are Dirac measures independent of their argument recovering the observations, are not drawn.

To illustrate the issue, we consider Doob's h-transform for the DAG in Figure 6. The rightmost picture shows the situation corresponding to a collider in graphical models. Here $\kappa_1 = \kappa_{1a} \otimes \kappa_{1b}$ has product form, but κ_2 does not. The transformed kernel $\kappa^\star_1$ typically won't have product form, the states become entwined (Jacobs, 2019). The solution is to break this dependency through backward marginalisation, was touched upon at the end of subsection 4.2:

In applying the backward map, one generic way to propagate h to its parents consists of splitting both h and m by marginalisation. We call this *backward marginalisation*.

Definition 5.13. For a reference measure of product form $\lambda \otimes \lambda' \in \mathcal{M}(S \otimes S')$, define the operator $M^* = M^*_{\lambda\otimes\lambda'}$ on $\mathbf{B}(S \otimes S')$ by

$$M^*h = \frac{\int h(\cdot, x')\lambda'(\mathrm{d}x')}{\sqrt{\int h \,\mathrm{d}(\lambda \otimes \lambda')}} \odot \frac{\int h(x, \cdot)\lambda(\mathrm{d}x)}{\sqrt{\int h \,\mathrm{d}(\lambda \otimes \lambda')}},$$

taking functions h to product form by marginalisation with respect to the reference measure.

Remark 5.14. $\widetilde{\kappa}h$ can be regularised for a suitable reference measure λ by setting $h'_\to(x) = \int q(x)h(y)\widetilde{\kappa}(x, \mathrm{d}y)$ for a chosen regularising density q. Then

$$\int h_\to(x) \,\mathrm{d}\lambda(x) < \infty.$$

This way, $h'_\to$ can be understood as unnormalised λ probability density. Note that

$$h'_\to = \Delta(\widetilde{\kappa}' \otimes \widetilde{\kappa})(h' \cdot h)$$

where $h' \equiv 1$ and $\widetilde{\kappa}'$ is chosen such that $\int \widetilde{\kappa}'(x, \mathrm{d}y) = q(x)$. This is essentially adding an artificial observation (creating an extra leaf in the graph). The bias of the artificial observation can be removed through sampling, just as in Theorem 4.3. This is closely related to Algorithm 5.4 in Mider et al. (2020) with the modified acceptance rate correcting for regularisation in the initialisation of the backward filter.

Under additional assumptions, the backward marginalisation commutes with pullback through product kernels.

Proposition 5.15. If the reference measures and Markov kernels $\kappa\colon S_1 \nrightarrow S_2$ and $\kappa'\colon S_1' \nrightarrow S_2'$, $\lambda_i \in \mathcal{M}(S_i)$, $\lambda_i' \in \mathcal{M}(S_i')$ $(i = 1, 2)$, satisfy

$$\lambda_1\kappa = \lambda_2, \quad \lambda_1'\kappa' = \lambda_2'$$

then for $h \in \mathbf{B}(S_2 \otimes S_2')$

$$(\kappa \otimes \kappa')M^*_{\lambda_2 \otimes \lambda_2'}h = M^*_{\lambda_1 \otimes \lambda_1'}(\kappa \otimes \kappa')h.$$

Proof. See appendix B. □

5.4.1 Backward filtering, forward guiding with backward marginalisation

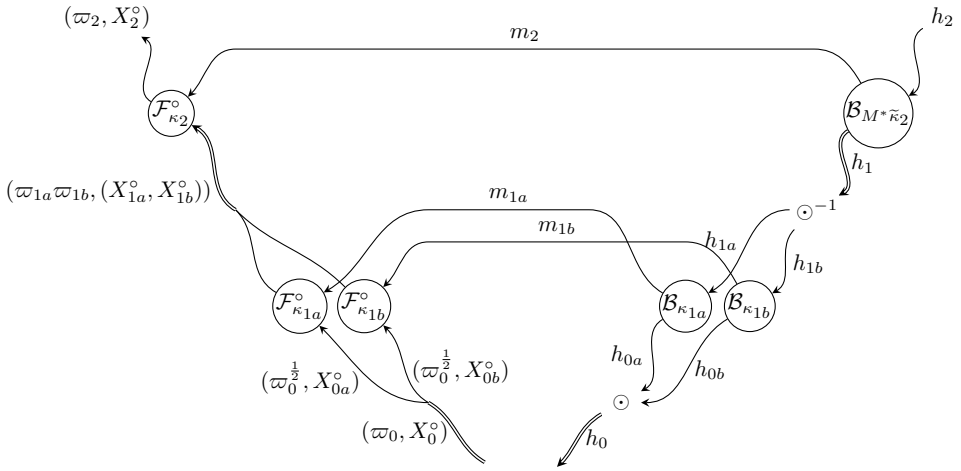
We now have all constituents for a universal procedure of backward filtering with marginalisation and forward guiding (by sampling), which doesn't put constraints on the program architecture, because it is applicable to any composition of (product-) kernels and it preserves the program architecture. The transformed program is a two-pass algorithm where the forward pass has the same architecture as the program: it has the same Markovian structure, the same function calls, the same diagrammatical form. Also the backward pass has the same architecture, though traversed backwards. Forwards, kernels κ or *factors* κ *of product kernels* $\kappa \otimes \kappa'$ are replaced by $\mathcal{F}^\circ_\kappa$, and backwards, kernels κ or *factors* κ *of product kernels* $\kappa \otimes \kappa'$ are replaced by $\mathcal{B}_{\bar\kappa}$, where $\bar\kappa$ may be the approximation $\widetilde\kappa$ of κ or the composition of M^* with $\widetilde\kappa$: $\bar\kappa = M^*\widetilde\kappa$. The entire program consists of pairs

$$\kappa \mapsto \langle \mathcal{F}^\circ_\kappa \mid \mathcal{B}_{\bar\kappa} \rangle$$

defined locally and combined together by message passing (in writing $\kappa \mapsto \langle \mathcal{F}^\circ_\kappa \mid \mathcal{B}_{\bar\kappa} \rangle$ we implicitly assume that we choose $\bar\kappa$ depending on κ).

In particular, backward marginalisation allows to transform factors of products separately, even if they are composed with a kernel which is not of product from (e.g. on a DAG which is not a tree). In this way, for X on any DAG (translated to the corresponding string diagram of kernel compositions) an approximation X° of the conditional $X^\star$ is obtained that can be sampled from. Note that X°, unlike $X^\star$, preserves the Markovian structure of X and does not introduce new edges in the dependency graph (as it was the case in Figure 6). This is of importance: for a probabilistic program which is able to sample X, it is easier to adapt it to sample from X° with weights, than to adapt it to sample from $X^\star$ directly (which has a different dependence structure). The former is possible as automatic program transform.

The following string diagram illustrates the message flow in the particular example of the 'probabilistic program' $(\kappa_{1a} \otimes \kappa_{1b})\kappa_2$:



The corresponding program, with $\odot$ and $\odot^{-1}$ denoted by `join`* and `split`* respectively, is as follows:

```
function transform((w0, x0), h2)
    message2, h1 = backward_with_marginalisation(approx_kernel2, h2)
    h1a, h1b = split*(h1)
    message1a, h0a = backward(approx_kernel1a, h1a)
    message1b, h0b = backward(approx_kernel1b, h1b)
    h0 = join*(h0a, h0b)

    w0a = w0b = sqrt(w0) # split weighted samples
    (x0a, x0b) = x0

    (w1a, x1a) = forwardsample(kernel1a, message1a, (w0a, x0a)))
    (w1b, x1b) = forwardsample(kernel1b, message1b, (w0b, x0b)))

    (w1, x1) = (w1a*w1b, (x1a, x1b)) # join weighted samples
    (w2, x2) = forwardsample(kernel2, message2, (w1, x1))

    return h0, (w1, x1)
end
```

Though the use of backward marginalisation $M^\star\widetilde{\kappa}$ in $\mathcal{B}$ *changes* the message and the proposal law, it preserves the invariant of the algorithm, namely sampling the conditional distribution with weighted samples in a structure preserving way. The following proposition makes this precise for the example:

Proposition 5.16. If $\kappa = (\kappa_{1a} \otimes \kappa_{1b})\kappa_2$ and $\widetilde{\kappa} = (\widetilde{\kappa}_{1a} \otimes \widetilde{\kappa}_{1b})\ M^*\widetilde{\kappa}_2$, then for measurable bounded f,

$$(\kappa f h)(x) = (\widetilde{\kappa} h)(x) \int \left(\int f(z) m_2(y, z) \kappa_2(y, \mathrm{d}z) \right) m_{1a}(x_a, y_a) \kappa_1(x_a, \mathrm{d}y_a) m_{1b}(x_b, y_b) \kappa_{1b}(x_b, \mathrm{d}y_b),$$

where

$$m_{1a}(x_a, y_a) = \frac{h_a(y_a)}{(\widetilde{\kappa}_{1a} h_a)(x_a)}, \qquad m_{1b}(x_b, y_b) = \frac{h_b(y_b)}{(\widetilde{\kappa}_{1b} h_b)(x_b)}, \qquad m_2(y, z) = \frac{h(z)}{(M^*\widetilde{\kappa}_2 h)(y)}$$

with $y = (y_a, y_b)$ and $h_a(y_a)h_b(y_b) = (M^*\widetilde{\kappa}_2 h)((y_a, y_b))$.

This proposition takes a particular, though representative, composition $(\kappa_{1a} \otimes \kappa_{1b})\kappa_2$ as example. How this applies to any program structure is topic of the next section, where we also formalise the rules underlying the wiring the string diagrams consisting of $\kappa \mapsto \langle \mathcal{F}_\kappa^\circ \mid \mathcal{B}_{\bar{\kappa}} \rangle$ with $\bar{\kappa}$ as above using category theory.

6 Backward filtering forward guiding from the perspective of category theory

Introducing a categorical perspective is motivated by the admission that we have already used the concepts introduced shortly in all but name in section 5 and by the following testament:

> "We should approach the problem of statistical modelling and computation in a modular, composable, functional way, guided by underpinning principles from category theory." Wilkinson (2017)

We show that composing two kernels and subsequently running the backward recursion for h, following by repeatedly applying the forward map is equivalent to a composition rule for the backward recursion – forward map. To formalise this setup, we use the optic, a construction from category theory. Optics, originally introduced as Getter/Setters and then generalised, have recently been discovered as devise to formalise the task of automatic (reverse mode) differentiation of computer programs (communicated by Keno Fischer, see also Fischer (2020)).

6.1 Categories of Markov kernels

A key element in our string diagrams are Markov kernels. Markov kernels $\kappa\colon S \nrightarrow S'$ are the morphisms of the Markov category **BorelStoch** with objects standard Borel spaces S, S'. Composition of morphisms is the composition of Markov kernels (as defined by the Chapman-Kolmogorov equations in equation (5.2)).

The symmetric monoidal structure on **BorelStoch** is given by product measure spaces at the level of objects, and the product Markov kernel $\kappa \otimes \kappa'$ at the level of morphisms.

BorelStoch is a symmetric monoidal category extended by a comultiplication, that is identically duplicating something, $\Delta\colon S \nrightarrow S \otimes S$ see Fritz (2020) and Jacobs (2019).[3] $\mathcal{I}$ is the monoidal unit object. See also Panangaden (1999).

6.2 Actions of Markov kernels

A Markov kernel in **BorelStoch** can act as pushforwards on measures and as pullbacks on functions (note that these are not pushforward and pullback as defined in category theory). These actions constitute two functors G (pushforward of measures) and H (pullback of bounded measurable functions) from **BorelStoch** into the category $(\mathbf{Set}, \times, I)$ with monoidal unit I (we denote functions $f\colon A \to B$ seen as morphisms in **Set** also by $f\colon A \nrightarrow B$.)

More precisely, if S and T are Borel spaces, then G is the functor from **BorelStoch** to **Set** with

objects:	$\mathcal{M}(S)$, the set of bounded measures on S
morphisms:	functions $G(\kappa)\colon \mathcal{M}(S) \nrightarrow \mathcal{M}(T)$ with $[G(\kappa)](\mu) = \mu\kappa$ (equation (5.1))

[3] and a counit $\mathrm{del}\colon S \nrightarrow \mathcal{I}$, for each object $(E, \mathfrak{B})$, that is forgetting something.

and H is the functor H from **BorelStoch** to **Set** with

objects:	$\mathbf{B}(S)$, the set of bounded measurable functions on S
morphisms:	functions $H(\kappa)\colon \mathbf{B}(S) \to \mathbf{B}(T)$ with $[H(\kappa)](h) = \kappa h$ (equation (5.5))

Before we denoted $G(\kappa)$ and $H(\kappa)$ by κ as well. Note that $G(\mathrm{del})$, the counit as Markov kernel acting on a measure, preserves one piece of information: the weight, and algorithmically, in sums over the weights these weights may need to be included.

From Lemma 5.9 we obtain the following proposition (communicated by Evan Patterson)

Proposition 6.1. (i) Product measures can be pushed forward in parallel by parallel kernels: Write $S = (E, \mathfrak{B})$, $S' = (E', \mathfrak{B}')$, $T = (F, \mathfrak{C})$, $T' = (F', \mathfrak{C}')$ etc. The functor G equipped with the natural transformation $\otimes_{S,S'}\colon G(S) \times G(S') \to G(S \otimes S')$ which takes pairs of measures to their product is a lax monoidal functor from $(\mathbf{BorelStoch}, \otimes, \mathcal{I})$ to $(\mathbf{Set}, \times, I)$.

(ii) Product form functions h can be pulled back in parallel by parallel kernels: The functor H equipped with the natural transformation $\odot_{S,S'}\colon H(S) \times H(S') \to H(S \otimes S')$ which takes pairs of functions to their pointwise product is a lax monoidal functor from $(\mathbf{BorelStoch}^{\mathrm{op}}, \otimes, \mathcal{I})$ to $(\mathbf{Set}, \times, I)$.

Proof. See appendix B. □

Lax reflects that $\otimes$ lacks invertibility on $S \otimes S'$ and $\odot$ lacks invertibility on $H(S) \odot H(S')$.

Though one cannot push forward a measure which is not of product form in parallel, one can after marginalisation (with Proposition 5.8), as illustrated by the following diagram:

$$\begin{array}{ccccc} G(S \otimes S') & \xrightarrow{M_{S,S'}} & M(G(S \otimes S')) & \xrightarrow[\simeq]{\otimes^{-1}_{S,S'}} & G(S) \times G(S') \\ \downarrow{\scriptstyle G(\kappa_1 \otimes \kappa_2)} & & \downarrow{\scriptstyle G(\kappa_1 \otimes \kappa_2)} & & \downarrow{\scriptstyle G(\kappa_1) \times G(\kappa_2)} \\ G(T \otimes T') & \xrightarrow{M_{T,T'}} & M(G(T \otimes T')) & \xrightarrow[\simeq]{\otimes^{-1}_{T,T'}} & G(T) \times G(T') \end{array}$$

Likewise, one cannot pull backward a function h which is not of product form in parallel, one can after backward marginalisation (under the assumptions of Proposition 5.15), as illustrated by the following diagram:

$$\begin{array}{ccccc} H(T \otimes T') & \xrightarrow{M^*_{T,T'}} & M^*(H(T \otimes T')) & \xrightarrow[\simeq]{\odot^{-1}_{T,T'}} & H(T) \times H(T') \\ \downarrow{\scriptstyle H(\kappa_1 \otimes \kappa_2)} & & \downarrow{\scriptstyle H(\kappa_1 \otimes \kappa_2)} & & \downarrow{\scriptstyle H(\kappa_1) \times H(\kappa_2)} \\ H(S \otimes S') & \xrightarrow{M^*_{S,S'}} & M^*(H(S \otimes S')) & \xrightarrow[\simeq]{\odot^{-1}_{S,S'}} & H(S) \times H(S') \end{array}$$

6.3 Sampling

The limitation in pushing forward product measures does not apply to sampling: one can sample forward in parallel from a sample of a joint distribution.

Recall the definition of $\kappa \mapsto f_\kappa$ from definition 5.5. Such randomisations (or randomness pushback in the terminology of Fritz (2020)) form a monoidal category **Sample**. Let $\Lambda = ([0,1], \mathfrak{B}([0,1]))$ equipped with Lebesgue measure λ_{Leb}. The construction relies on the existence of reproduction

functions $r, r' : \Lambda \to \Lambda$ such that r and r' are independent uniformly distributed random variables on Λ. If $z \in [0,1]$ has binary expansion $z = \sum_{j=1}^{n} 2^{-j} z_i$,[4]

$$r(z) = \sum_{j=1}^{\infty} 2^{-2j+1} z_{2j-1}, \quad r'(z) = \sum_{j=1}^{\infty} 2^{-2j} z_{2j}. \tag{6.1}$$

The proof follows along the lines of Lemma 3.21 in Kallenberg (2002).

Definition 6.2. Objects of the category **Sample** are Borel measure spaces $S = (E, \mathfrak{B})$, $S' = (E', \mathfrak{B}')$... Morphisms $f : S \to S'$ are equivalence classes of measurable functions $f : S \otimes \Lambda \to S'$ under the equivalence relation

$$f \sim f' \quad \Leftrightarrow \quad \int_A f(x,y) \lambda_{\text{Leb}}(\,\mathrm{d}y) = \int_A f'(x,y) \lambda_{\text{Leb}}(\,\mathrm{d}y) \qquad \text{for } x \in E \text{ and } A \in \mathfrak{B}' \tag{6.2}$$

for $f, f' : E \times [0,1] \to E'$.

A representative of the identity $\mathcal{I}_{S,S}$ is $f(x,z) = x, x \in E$. Composition $f' \circ f$ of $f : S \to S'$ and $f : S' \to S''$is defined for representatives of f, f' by

$$(f' \circ f)(x,z) = f'(f(x, r(z)), r'(z)).$$

The product $\otimes$ is defined on objects a product of measure spaces $S \otimes S'$ and a representative of $f \otimes f' : S \otimes S' \to T \otimes T'$ given by

$$(f \otimes f')((x,x'),z) = (f(x, r(z)), f'(x', r'(z))).$$

Proposition 6.3. The functor $f_\kappa = J(\kappa)$ is a *strong* monoidal functor.

Proof. For $\kappa : S \to T$, $\kappa' : S' \to T'$ the following diagram commutes, the natural transformation is the identity isomorphism.

$$\begin{array}{ccc} J(S) \otimes J(S') & \xrightarrow[\simeq]{} & J(S \otimes S') \\ {\scriptstyle J(\kappa)\otimes J(\kappa')}\downarrow & & \downarrow{\scriptstyle J(\kappa\otimes\kappa')} \\ J(T) \otimes J(T') & \xrightarrow[\simeq]{} & J(T) \otimes J(T') \end{array}$$

The remaining conditions for strong monoidal functors (cf. the entry monoidal functor in nLab authors (2021)) are easily checked. □

This means: as long as we are sampling, if $\kappa = \kappa_a \otimes \kappa_b$ we may freely switch between sampling $f_\kappa((x_a, x_b), Z)$ and creating a pair from $f_{\kappa_a}(x_a, Z_a)$ and $f_{\kappa_b}(x_b, Z_b)$ sampled in parallel with independent uniform innovations $Z_a = r(Z), Z_b = r'(Z)$ up to equivalence with respect to (6.2).

6.4 Optics

The pair $\langle \mathcal{F} \mid \mathcal{B} \rangle$ (with the backward and forward maps defined in definition 5.2 and definition 5.1 respectively) has a categorical interpretation as morphism in a corresponding category. The defining characteristic of a category is its composition rule, in this case specifying how to obtain

[4] Opting for a non-terminating representation instead of a terminating one when both are available.

$\langle \mathcal{F}_{12} \mid \mathcal{B}_{12} \rangle$ as the composition of $\langle \mathcal{F}_1 \mid \mathcal{B}_1 \rangle \langle \mathcal{F}_2 \mid \mathcal{B}_2 \rangle$. We already know how "our" optics compose: Suppose the kernels κ_1 and κ_2 are composed to $\kappa_{12} = \kappa_1 \kappa_2$. By definition

$$\begin{aligned} \mathcal{B}_{\kappa_{12}}(h) &= (m, \kappa_{12} h) \\ \mathcal{F}_{\kappa_{12}}(m, \mu) &= \nu, \qquad \text{where} \quad \nu(\mathrm{d}y) = \int m(x,y)\mu(\mathrm{d}x)\kappa_{12}(x, \mathrm{d}y), \end{aligned} \tag{6.3}$$

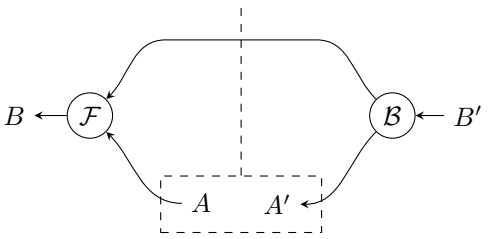
with m as defined in (5.9). Similarly we may rewrite function `transform123` to a pair $\langle$`forward`|`backward`$\rangle$, elucidating the composition structure.

```
function backward(::typeof(kernel123), h3)
    message3, h2 = backward(kernel3, h3)
    message2, h1 = backward(kernel2, h2)
    message1, h0 = backward(kernel1, h1)
    m = (message1, message2, message3)
    return m, h0 # return message, evidence
end

function forward(::typeof{kernel123}, m, mu0)
    (message1, message2, message3) = m
    mu1 = forward(kernel1, message1, mu0)
    mu2 = forward(kernel2, message2, mu1)
    mu3 = forward(kernel3, message3, mu2)
    return mu3 # return marginal
end

message, h0 = backward(kernel1234, h3)
mu3 = forward(kernel1234, message, mu0)
```

The syntax `function backward(::typeof(kernel1234), ...)` indicates the definition of a function specifically being executed when `backward(kernel1234, h3)` is being called, just as `backward(kernel3, h3)` executes some other version of the function, not reproduced here.

In this section we will introduce the category of optics, with a composition rule that is equivalent to (6.3) or $\langle$`forward`|`backward`$\rangle$. With guided processes of previous section in mind, we allow for different kernels in the backward and forward map. In particular we will consider $\mathcal{B}_{\widetilde{\kappa}_{12}}(h) = (m, \widetilde{\kappa}_{12} h)$, where $\widetilde{\kappa}_{12} = \widetilde{\kappa}_1 \widetilde{\kappa}_2$.

Essentially an optic is a diagrammatic scheme as shown below



It is the essential building block of diagram 3. The circled vertices in the diagram correspond to two maps $\mathcal{B}\colon B' \to M \times A'$, $\mathcal{F}\colon M \times A \to B$, representing a backward pass followed by a forward pass. They are written symbolically as pair $\langle \mathcal{F} \mid \mathcal{B} \rangle$,[5] where $\mathcal{B}$ takes as input a value $b' \in B'$ and

[5]Our optics are right-to-left. In the literature, the same optic would be written $\langle \mathcal{B} \mid \mathcal{F} \rangle$ and drawn from left to the right.

produces as output a message $m \in M$ and a value $a' \in A'$. The message can be consumed by $\mathcal{F}$ to produce an output $b \in B$ from an input $a \in A$. Here the forward pass depends on the message of the backward pass.

For the h-transform, $\mathcal{B}$ and $\mathcal{F}$ are specified in equations (5.9) and (5.8) respectively. We may consider the maps $\mathcal{F}$ and $\mathcal{B}$ as morphisms in the category $\mathcal{D} \subset \mathbf{Set}$.

Then the pairs $\langle \mathcal{F} \mid \mathcal{B} \rangle$ are morphisms in a corresponding category of right-to-left optics $\mathbf{Optic}_{\mathcal{D}}$ derived from $\mathcal{D}$. In the diagram, the circled vertices in the diagram correspond to two morphisms, and the sources and targets of the morphisms (objects of the category) are annotated in the diagram. (If the morphisms are functions and the category is **Set**, then the sources and targets are domains and codomains.)

Following Riley (2018), this category has objects given by pairs (A, A') , (B, B') of objects in a symmetric monoidal category $(\mathcal{D}, \otimes, \mathcal{I})$ and morphisms represented by pairs $(\mathcal{F}, \mathcal{B})$ of morphisms in $\mathcal{D}$.[6] M is interpreted as message (domain). Optics are equivalent modulo changes in representation of the same message. For a morphism $g \colon M \to M$ both $((g \otimes \mathrm{id})\mathcal{F}, \mathcal{B})$ and $(\mathcal{F}, \mathcal{B}(g \otimes \mathrm{id}))$ are representatives of the same optic and a transformation g of a message can be equivalently applied before or after sending the message ($(g \otimes \mathrm{id})$ acts on the message alone.)

These morphisms are denoted as $\langle \mathcal{F} \mid \mathcal{B} \rangle \colon (A, A') \nrightarrow (B, B')$, with the arrow accented to distinguish from morphisms in the category $\mathcal{D}$. We write

$$\langle (g \otimes \mathrm{id}_{A'})\mathcal{F}, \mathcal{B} \rangle = \langle \mathcal{F}, \mathcal{B}(g \otimes \mathrm{id}_{B'}) \rangle \tag{6.4}$$

For two morphisms $f \colon A \to B$, $b \colon B' \to A'$, define

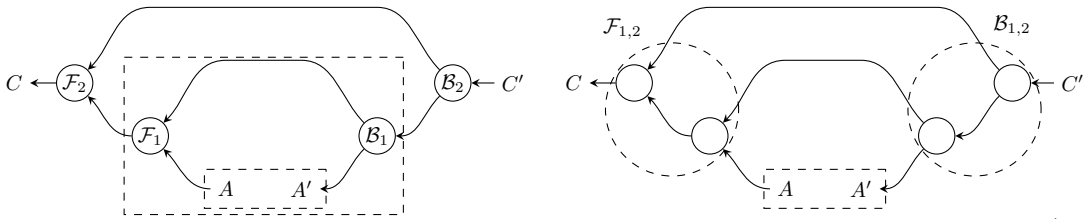
$$\iota(f, b) = \langle f \circ \lambda, \lambda^{-1} \circ b \rangle \tag{6.5}$$

as the optic which applies b and f and sends an empty/void message (here λ is the unitor $\lambda \colon \mathcal{I} \times A \to A$, in case of functions as morphisms in **Set**, $\lambda(m, a) = a, m \in I$, $m \in A$.)

Sequential (left-to-right) composition of these optics,

$$\langle \mathcal{F}_1 \mid \mathcal{B}_1 \rangle \ \langle \mathcal{F}_2 \mid \mathcal{B}_2 \rangle = \langle \mathcal{F}_{1,2} \mid \mathcal{B}_{1,2} \rangle$$

into a new optic defined by the morphisms $\mathcal{F}_{1,2}$ and $\mathcal{B}_{1,2}$ is best understood in a diagrammatic sense, as shown in the following diagram. Here, the two optics are composed inserting $\langle \mathcal{F}_1 \mid \mathcal{B}_1 \rangle$ into the dashed box of $\langle \mathcal{F}_2 \mid \mathcal{B}_2 \rangle$, and connecting the morphism to form new $\mathcal{F}_{1,2}$ and $\mathcal{B}_{1,2}$ (dashed circles):



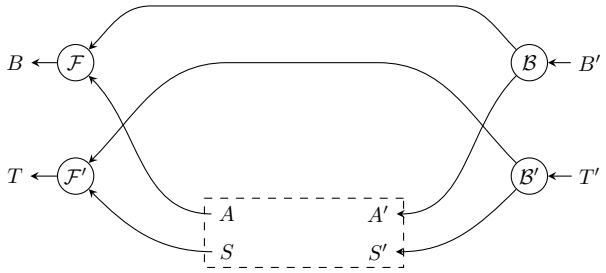
(6.6)

The composition on the left-hand-side is – as the right hand side illustrates – another optic $\langle \mathcal{F}_{1,2} \mid \mathcal{B}_{1,2} \rangle$. Algebraically, the new morphisms, written as functions, are given by

$$\begin{aligned} \mathcal{B}_{1,2}(c') &:= (m_1 \otimes m_2, a') \text{ where } (m_2, b') = \mathcal{B}_2(c'), (m_1, a') = \mathcal{B}_1(b'), \\ \mathcal{F}_{1,2}(m_1 \otimes m_2, a) &:= \mathcal{F}_2(m_2, \mathcal{F}_1(m_1, a)). \end{aligned} \tag{6.7}$$

[6] Where $\mathcal{B} \colon B' \to M' \otimes A'$, $\mathcal{F} \colon M' \otimes A \to B$ for M.

Secondly, optics can be applied in parallel. With the product $\otimes$ the category becomes symmetric monoidal, where $(B, B') \otimes (T, T') = (B \otimes T, B' \otimes T')$, the unit object is $(\mathcal{I}, \mathcal{I})$ and $\langle \mathcal{F} \mid \mathcal{B} \rangle \otimes \langle \mathcal{F}' \mid \mathcal{B}' \rangle$ is defined by the diagram



Algebraically, the new morphisms, written as functions, are given by

$$\begin{aligned} \mathcal{B}_{1\times 2}(b' \otimes t') &:= (m_1 \otimes m_2, a' \otimes s') \text{ where } (m_1, a') = \mathcal{B}_1(b'), (m_2, s') = \mathcal{B}_2(t'), \\ \mathcal{F}_{1\times 2}(m_1 \otimes m_2, a \otimes s) &:= \mathcal{F}_2(m_2, a) \otimes \mathcal{F}_1(m_1, s). \end{aligned} \tag{6.8}$$

The following program reflects this structure.

```
function backward(::typeof{kernelab}, (ha, hb))
    messagea, ha = backward(kernela, ha)
    messageb, hb = backward(kernelb, hb)
    m = (messagea, messageb)
    h = (ha, hb)
    return m, h
end

function forward(::typeof{kernelab}, m, (mua, mub))
   (messagea, messageb) = m
    mua = forward(kernela, messagea, mua)
    mub = forward(kernelb, messageb, mub)
    mu = (mua, mub)
    return mu
end
```

6.5 Optics for guided processes

Kernels can be composed sequentially and in parallel, and those compositions make up the structure of a probabilistic program. In this section we show that the optics can be composed likewise, and that composition of optics (in sequentially and in parallel) preserves the structure of the program.

We now consider $\langle \mathcal{F} \mid \mathcal{B} \rangle \colon (S, \mathbf{B}(S)) \nrightarrow (T, \mathbf{B}(T))$ as morphism in $\mathbf{Optic}_{(\mathbf{Set}, \times, I)}$. The space of messages M is a subspace of $\mathcal{M}(S \times T)$.

6.5.1 Sequential composition

Consider the optic in (6.3), with the backward map with $\widetilde{\kappa}_{1,2}$ and the forward map with $\kappa_{1,2}$. As an alternative to composing kernels and looking a the resulting optic, we can also compose

optics $\langle \mathcal{F}_{\kappa_1} \mid \mathcal{B}_{\widetilde{\kappa}_1} \rangle$ and $\langle \mathcal{F}_{\kappa_2} \mid \mathcal{B}_{\widetilde{\kappa}_2} \rangle$ corresponding to $\kappa_1, \kappa_2, \widetilde{\kappa}_1, \widetilde{\kappa}_2$ by

$$\langle \mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2} \mid \mathcal{F}_{\kappa_1,\kappa_2} \rangle := \langle \mathcal{B}_{\widetilde{\kappa}_1} \mid \mathcal{F}_{\kappa_1} \rangle \langle \mathcal{B}_{\widetilde{\kappa}_2} \mid \mathcal{F}_{\kappa_2} \rangle$$

More precisely, with $m_2(x,y) = \dfrac{h_2(y)}{(\widetilde{\kappa}_2 h_2)(x)}$ and $m_1(z,x) = \dfrac{(\widetilde{\kappa}_2 h_2)(x)}{(\widetilde{\kappa}_1 \widetilde{\kappa}_2 h_2)(z)}$,

$$\begin{aligned} \mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2}(h_2) &= ((m_1, m_2), \widetilde{\kappa}_1 \widetilde{\kappa}_2 h_2), \\ \mathcal{F}_{\kappa_1,\kappa_2}((m_1,m_2),\mu) &= \mathcal{F}_{\kappa_2}(m_2, \mathcal{F}_{\kappa_1}(m_1,\mu)). \end{aligned} \tag{6.9}$$

The optic corresponding to a composition of kernels can be obtained as composition of optics of the kernels:

Theorem 6.4. $\mathcal{F}_{\kappa_{12}}$ and $\mathcal{B}_{\widetilde{\kappa}_{12}}$ represent the same optic as $\mathcal{F}_{\kappa_1,\kappa_2}$ and $\mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2}$, that is

$$\langle \mathcal{F}_{\kappa_{12}}, \mathcal{B}_{\widetilde{\kappa}_{12}} \rangle = \langle \mathcal{F}_{\kappa_1} \mid \mathcal{B}_{\widetilde{\kappa}_1} \rangle \langle \mathcal{F}_{\kappa_2} \mid \mathcal{B}_{\widetilde{\kappa}_2} \rangle.$$

Proof. By (6.4) we need to find g, with $g \times \mathrm{id}$ invertible on the range of $\mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2}$, such that

$$\begin{aligned} \mathcal{F}_{\kappa_{12}}(m',\mu) = \mathcal{F}_{\kappa_1,\kappa_2}(g(m'),\mu), \quad m'(z,y) &= \frac{h_2(y)}{\kappa_{12} h_2(z)} \\ \mathcal{B}_{\widetilde{\kappa}_{12}}(h_2) = (g^{-1}(m_1,m_2), h_0) \text{ where } ((m_1,m_2),h_0) &= \mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2}(h_2). \end{aligned} \tag{6.10}$$

Define $g(m') = \Phi^{-1} \circ m_g$ where $\Phi(m_1,m_2)((z,x),(x',y)) = m_2(x',y) m_1(z,x)$.

and $m_g((z,x),(x',y)) = m'(z,y) \cdot \frac{\widetilde{\kappa}_2 h_2(x')}{\widetilde{\kappa}_2 h_2(x)}$. For the second relation, by the first equation of display (6.3), $\mathcal{B}_{\widetilde{\kappa}_{12}}(h_2) = (m', h_0)$. Now

$$m_g((z,x),(x',y)) = \frac{h_2(y)}{(\widetilde{\kappa}_2 h_2)(x)} \frac{(\widetilde{\kappa}_2 h_2)(x')}{(\widetilde{\kappa}_1 \widetilde{\kappa}_2 h_2)(z)} = m_2(x',y) m_1(z,x) = \Phi(m_1,m_2)((z,x),(x',y)).$$

Thus $(g(m'), h_0) = ((m_1,m_2),h_0)$ equals $\mathcal{B}_{\widetilde{\kappa}_1,\widetilde{\kappa}_2}(h_2)$ and the relation follows.

To verify the first relation, define $\nu_1 = \mathcal{F}_{\kappa_1}(m_1,\mu)$ and $\nu_2 = \mathcal{F}_{\kappa_2}(m_2,\nu_1)$ so

$$\nu_2 = \mathcal{F}_{\kappa_1,\kappa_2}((m_1,m_2),\mu).$$

By applying the definition of the forward map in (5.8) twice, we get

$$\nu_2(\mathrm{d}y) = \int\int m_2(x,y) m_1(z,x) \mu(\mathrm{d}x) \kappa_1(z,\, \mathrm{d}x) \kappa_2(x,\, \mathrm{d}y)$$

and substituting $h_1 = \widetilde{\kappa}_2 h_2$ gives cancellation of terms in the numerator of the second term and denominator of the first term. Hence

$$\begin{aligned} \nu_2(\mathrm{d}y) &= \int\int \frac{h_2(y)}{(\widetilde{\kappa}_1 \widetilde{\kappa}_2 h_2)(z)} \kappa_1(z,\, \mathrm{d}x) \kappa_2(z,\, \mathrm{d}y) \mu(\mathrm{d}z) \\ &= \int \frac{h_2(y)}{(\widetilde{\kappa}_{12} h_2)(z)} \kappa_{12}(z,\, \mathrm{d}y) \mu(\mathrm{d}z) \\ &= \int m'(z,y) \kappa_{12}(z,\, \mathrm{d}y) \mu(\mathrm{d}z). \end{aligned}$$

This is exactly as in the second equation of display (6.3). □

Corollary 6.5. $F(\kappa,\widetilde{\kappa}) = \langle \mathcal{F}_\kappa \mid \mathcal{B}_{\widetilde{\kappa}} \rangle$ defined for parallel morphisms $\kappa, \widetilde{\kappa} \colon S \nrightarrow T$ is a lax monoidal (bi-)functor on the category of parallel **BorelStoch** morphisms into $\mathbf{Optic}_{(\mathbf{Set},\times,I)}$.

Proof. Combine the preceding theorem with Lemma 5.9 and Proposition 6.1. The main missing piece is that $\mathcal{B}_{\widetilde{\kappa}_a \otimes \widetilde{\kappa}_b}$ applied to a product $h_a \odot h_b$ produces a message of product form $m_a \mathbin{\widetilde{\otimes}} m_b$ and $\mathcal{F}_{\kappa_a \otimes \kappa_b}(m_a \mathbin{\widetilde{\otimes}} m_b, \cdot)$ is a product kernel for fix messages of product form. □

6.5.2 Products

What is missing is obtaining the optic of a product kernel as product of the optics of its parts. The caveat is that in the backward pass we require backward marginalisation. In the forward pass, we can apply sampling. The difficulty is how to replace the optic of a product kernel with its product of optics within sequential composition.

Recall definition 5.7, where we defined the random map

$$\mathcal{F}^{\circ}_{\kappa}(m, (\varpi, x)) = (\varpi w_{\kappa}(m, \delta_x), Y), \quad Y \sim \nu,$$

where $w_{\kappa}(m, \delta_x) = \int \mathrm{d}\mathcal{F}_{\kappa}(m, \delta_x)(\,\mathrm{d}y)$ (see (5.12)) and ν is the probability measure equal to $\mathcal{F}_{\kappa}(m, \delta_x)/w_{\kappa}(m, \delta_x)$. To reason about forward samplers as composable morphisms in **Set** we construct a variant of $\mathcal{F}^{\circ}$ that we denote by $\mathcal{F}^{\diamond}$ and which is a composable function.

The first idea is to chose a randomization of the kernel $\mathcal{F}_{\kappa}(m, \delta_x)/w_{\kappa}(m, x)$ by Lemma 5.5. With that we represent Y by $f^m_{\kappa}(x, Z)$, where f^m_{κ} is deterministic randomisation function and Z the innovation random variable.

However, to compose, we also wish to construct a new innovation variable from the output of the new forward map $\mathcal{F}^{\diamond}$ to be used as input to the subsequent randomisation function..

For that purpose, use the construction from equation (6.1) in terms of the maps r and r'. Define the function $\mathcal{F}^{\diamond}_{\kappa} \colon M \times \mathbb{R} \times E \times [0,1] \to \mathbb{R} \times F \times [0,1]$

$$\mathcal{F}^{\diamond}_{\kappa}(m, (\varpi, x), z) = (\varpi w_{\kappa}(m, \delta_x), f^m_{\kappa}(x, r(z)), r'(z)).$$

Here the innovation variable is split into two components, which both are independent uniform innovations, one to be use as innovation for f^m_{κ}, and one made part of the output of $\mathcal{F}^{\diamond}$ to be used as input in a subsequent forward sampling step, where it is split up again and partly used as innovation.

In some sense $\mathcal{F}^{\diamond}_{\kappa}$ is a variant of $\mathcal{F}^{\circ}_{\kappa}$ which carries a random number generator with it, cf. the concept of a monad. Here z is a *random seed* value in $[0,1]$, which through splitting with r, r' becomes an infinite sequence of numbers which are independently uniformly distributed, if the initial seed $z = Z$ is drawn from a uniform.

Equation (6.2) defines an congruence relation R on optics $\langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle$ by considering two optics to be equivalent if they only differ by equivalent choice of each f^m_{κ}, $m \in M$.

Proposition 6.6. $\widetilde{F}(\kappa, \widetilde{\kappa}) = \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle$ is a functor into the category $\mathbf{Optic}_{(\mathbf{Set}, \times, I)}$ quotiented by congruence relation R.

For the remainder, instead of introducing new category theoretic concepts, we describe the properties using more elementary terms.

In the following definition, we clarify the meaning of the composed forward-backward map $\langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle$. The starting point is that the backward pass receives a given h and the forward pass samples conditional on a given value x.

Definition 6.7. For $x \in E, h \in \mathbf{B}$ and a measurable function f, define

$$\mathcal{E}_{x,f,h} \ \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle = \mathbb{E}\ \varpi' f(X'),$$

where

$$\varpi', X', Z' = \mathcal{F}^{\diamond}_{\kappa}(m, (\varpi, x), Z), \qquad m, h_{-} = \mathcal{B}_{\widetilde{\kappa}} h,$$

$\varpi = \widetilde{\kappa} h(x)$ and $Z \sim U[0,1]$.

Hence $\mathcal{E}$ computes the expectation of a functional with respect to the weighted samples at the end of the forward pass, multiplied by $\widetilde{\kappa}h$. The next proposition captures that this means weighted sampling from measure proportional to the conditional, with proportionality constant $(\kappa h)(x_0)$ independent from the choice of $\widetilde{\kappa}$.

Proposition 6.8.

$$\mathcal{E}_{x,f,h}\ \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle = (\kappa f h)\,(x), \quad \forall f \in \mathbf{B}$$

Proof. By definition of $\mathcal{E}$ and (5.13)

$$\mathcal{E}_{x,f,h}\ \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle = \int f(y)\mathcal{F}_{\kappa}(m, (\widetilde{\kappa}h)(x), \delta_x)\,\mathrm{d}y = (\kappa f h)\,(x).$$

□

Equality under $\mathcal{E}_{x,f,h}$, that is sampling from measure proportional to the conditional, can be considered the invariant which is preserved by the guided process construction.

We now give the main result of this section. It says that both forward sampling and backward marginalisation operations preserve the invariant of guided processes.

Theorem 6.9. Define the split and join operators for samples by

$$\begin{aligned} \mathrm{split}(\varpi, (x, x'), z) &= ((\sqrt{\varpi}, x, r(z)), (\sqrt{\varpi}, x', r'(z))) \\ \mathrm{join}((\varpi, x, z), (\varpi', x', z')) &= (\varpi\varpi', (x, x'), z) \end{aligned}$$

respectively. Then

1. for $\kappa = \kappa_1(\kappa_{2a} \otimes \kappa_{2b})$ and $\widetilde{\kappa} = \widetilde{\kappa}_1(\widetilde{\kappa}_{2a} \otimes \widetilde{\kappa}_{2b})$

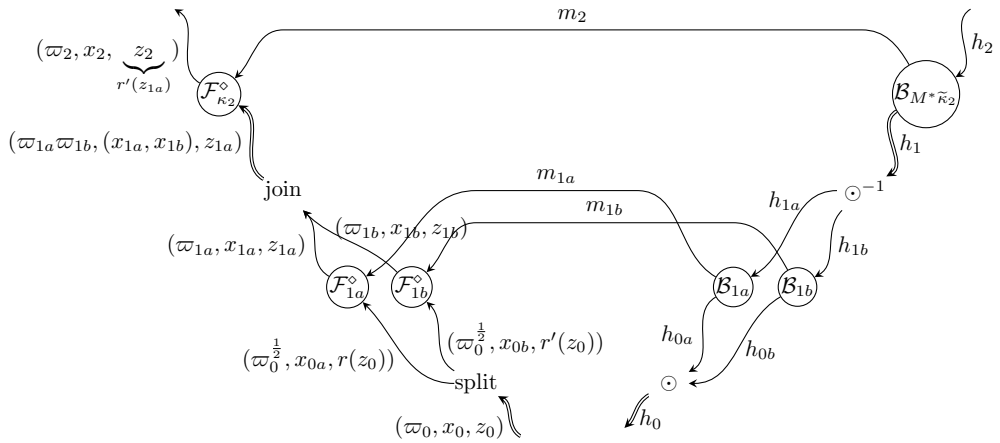
$$\mathcal{E}_{x,f,h}\ \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{\widetilde{\kappa}} \rangle = \mathcal{E}_{x,f,h}\ \langle \mathcal{F}^{\diamond}_{\kappa_1} \mid \mathcal{B}_{\widetilde{\kappa}_1} \rangle \iota(\mathrm{split}, \odot) \left(\langle \mathcal{F}^{\diamond}_{\kappa_{2a}} \mid \mathcal{B}_{\widetilde{\kappa}_{2a}} \rangle \times \langle \mathcal{F}^{\diamond}_{\kappa_{2b}} \mid \mathcal{B}_{\widetilde{\kappa}_{2b}} \rangle \right) \iota(\mathrm{join}, \odot^{-1}),$$

2. and for $\kappa = (\kappa_{1a} \otimes \kappa_{1b})\kappa_2$ and $\widetilde{\kappa} = (\widetilde{\kappa}_{1a} \otimes \widetilde{\kappa}_{1b})\widetilde{\kappa}_2$

$$\mathcal{E}_{x,f,h}\ \langle \mathcal{F}^{\diamond}_{\kappa} \mid \mathcal{B}_{M^*\widetilde{\kappa}} \rangle = \mathcal{E}_{x,f,h}\ \iota(\mathrm{split}, \odot) \left(\langle \mathcal{F}^{\diamond}_{\kappa_{1a}} \mid \mathcal{B}_{\widetilde{\kappa}_{1a}} \rangle \times \langle \mathcal{F}^{\diamond}_{\kappa_{1b}} \mid \mathcal{B}_{\widetilde{\kappa}_{1b}} \rangle \right) \iota(\mathrm{join}, \odot^{-1}) \langle \mathcal{F}^{\diamond}_{\kappa_2} \mid \mathcal{B}_{M^*\widetilde{\kappa}_2} \rangle,$$

with ι defined as in (6.5).

On the right hand side of the last equation, in words, we backward marginalise $h_m = M^*\widetilde{\kappa}_2 h$, send a first message $h(y)/h_m(x)$, split $(h_a, h_b) = \odot^{-1} h_m$ to obtain two functions h_a, h_b which are send as pair to the backward side of the product optic $\langle \mathcal{F}^{\diamond}_{\kappa_{1a}} \mid \mathcal{B}_{\widetilde{\kappa}_{1a}} \rangle \times \langle \mathcal{F}^{\diamond}_{\kappa_{1b}} \mid \mathcal{B}_{\widetilde{\kappa}_{1b}} \rangle$, pulled back and joined again into a single function with $\odot$. On the forward side of the product optic, we split the weighted state ϖ, (x, x') and the random number z with φ before we send it to the forward side of the product optic as pair, apply $\mathcal{F}^{\diamond}_{\kappa_{1a}}$ and $\mathcal{F}^{\diamond}_{\kappa_{1a}}$ separately and join the samples, their weights and their random numbers with ψ. The following diagram illustrates, double lines signify domains of product form.



(6.11)

Note just before $\mathcal{F}_{\kappa_2}$ it says z_{1a} as we have defined join to pick the random seed of its first argument (either would do.)

Remark 6.10. A right-to-left optic can programmatically be transformed into a left-to-right optic by a continuation style transformation, setting $l(a) = (m' \mapsto (a \mapsto \mathcal{F}(m', a)), b)$ and $r(m, b') = (m(m'), a')$ where $(m', a') = \mathcal{B}(b')$ where we set $b = b(a)$ by some rule.

This allows in principle to adapt rule-based code for automatic differentiation running a forward-backward pass to the task of computing the h-transform backward-forward. See appendix D.

7 Examples of guided processes

In this section we provide various examples to illustrate our approach.

The examples make use of the idea that the composition of backward steps is computationally cheap if it preserves a parametrised form of h. Hence, suppose h can be parametrised by $h = U(\zeta)$ for a parameter ζ and a parametrisation U. Backward filtering is tractable when

- $\widetilde{\kappa}h = U(\zeta')$ for some ζ';
- $\kappa_{\Delta_k}(h_1 \odot \cdots \odot h_k) = U(\zeta')$ for some ζ' (this corresponds to the fusion step).

In some models the (otherwise intractable) fusion step can be bypassed by only applying the backward map on a line graph, see section 7.3.

In applying the forward map for sampling less tractability is required, as for example the pseudo-marginal Metropolis-Hastings algorithm can be used to unbiasedly estimate $(\kappa h)(x)$. Details are given ahead in section 8.3.

7.1 Gaussian filtering and smoothing

Consider the stochastic process on $\mathcal{G}$ defined by

$$X_t \mid X_s = x \sim N(\mu_t(x), Q_t(x)), \qquad t \in \mathrm{ch}(s). \tag{7.1}$$

Unfortunately, for this class of models (parametrised by (μ_t, Q_t)), only in very special cases Doob's h-transform is tractable. For that reason, we look for a tractable $\widetilde{h}$-transform to define a guided process X°. This is obtained in the specific case where

$$\widetilde{X}_t \mid \widetilde{X}_s = x \sim N(\Phi_t x + \beta_t, Q_t), \qquad t \in \mathrm{ch}(s).$$

Below we show that backward filtering, sampling from the forward map and computing the weights are all fully tractable.

In the following, we write $\varphi(x; \mu, \Sigma)$ for the density of the $N(\mu, \Sigma)$-distribution, evaluated at x. Similarly, we write $\varphi^{\mathrm{can}}(x; F, H)$ for the density of canonical Normal distribution with potential $F = \Sigma^{-1}\mu$ and precision matrix $H = \Sigma^{-1}$, evaluated at x.

The h functions can be parametrised by the triple (c, F, H):

$$\begin{aligned} h(y) &= \exp\left(c - \frac{1}{2} y' H y + y' F\right) \\ &= \varpi(c, F, H)\varphi^{\mathrm{can}}(y; F, H) = \varpi(c, F, H)\varphi(y; H^{-1}F, H^{-1}), \end{aligned} \tag{7.2}$$

where

$$\log \varpi(c, F, H) = c - \log \varphi^{\mathrm{can}}(0, F, H).$$

We write $h(x) = U(c, F, H)(x)$ for h and parameters as in (7.2).

Theorem 7.1. Let $\kappa(x, \mathrm{d}y) = \varphi(y; \mu(x), Q(x))\,\mathrm{d}y$ and $\widetilde{\kappa}(x, \mathrm{d}y) = \varphi(y; \Phi x + \beta, Q)\,\mathrm{d}y$. Assume that $h = U(c, F, H)$ with invertible H.

(i) Pullback for κ: With $C(x) = Q(x) + H^{-1}$ invertible,

$$(\kappa h)(x) = \varpi(c, F, H)\varphi(H^{-1}F; \mu(x), C(x)).$$

(ii) Initialisation: if y is observed at a leaf, then

$$h(x) = U(\bar{c}, \bar{F}, \bar{H})(x) \quad \text{where} \quad \begin{cases} \bar{H} = \Phi' Q^{-1} \Phi \\ \bar{F} = \Phi' Q^{-1}(y - \beta) \\ \bar{c} = \log \varphi(\beta; y, Q). \end{cases}$$

(iii) Pullback for $\widetilde{\kappa}$: With $C = Q + H^{-1}$ invertible,

$$\widetilde{\kappa} h = U(\bar{c}, \bar{F}, \bar{H}) \quad \text{where} \quad \begin{cases} \bar{H} = \Phi' C^{-1} \Phi \\ \bar{F} = \Phi' C^{-1}(H^{-1}F - \beta) \\ \bar{c} = c - \log \varphi^{\mathrm{can}}(0, F, H) + \log \varphi(\beta; H^{-1}F, C) \end{cases}$$

and

$$\mathcal{B}_{\widetilde{\kappa}} = (m, \widetilde{\kappa} h), \quad m(x, y) = \frac{U(c, F, H)(y)}{U(\bar{c}, \bar{F}, \bar{H})(x)}.$$

If Φ is invertible, then alternatively

$$\log \varpi(\bar{c}, \bar{F}, \bar{H}) = \log \varpi(c, F, H) - \log |\Phi|.$$

(iv) Forward map:

$$\mathcal{F}_\kappa(m, \mu) = \nu, \quad \nu(\mathrm{d}y) = \int w(x) \varphi^{\mathrm{can}}(y; F + Q(x)^{-1}\mu(x), H + Q(x)^{-1}) \mu(\mathrm{d}x)\,\mathrm{d}y,$$

where the weight at x is given by

$$w(x) = \frac{(\kappa h)(x)}{(\widetilde{\kappa} h)(x)} = \frac{(\kappa h)(x)}{U(\bar{c}, \bar{F}, \bar{H})(x)} = \frac{\varphi(H^{-1}F; \mu(x), Q(x) + H^{-1})}{\varphi^{\mathrm{can}}(0, F, H) U(\bar{c} - c, \bar{F}, \bar{H})(x)}.$$

In particular, if $\mu = \varpi \delta_x$, the forward map gives the importance weight and conditional distribution for the next random draw,

$$\varpi w(x) \cdot N^{\mathrm{can}}\left(F + Q(x)^{-1}\mu(x), H + Q(x)^{-1}\right).$$

(v) Fusion: if $h_i = U(c_i, F_i, h_i)$, $i = 1, \ldots, k$, then

$$\Delta_k(h_1 \odot \cdots \odot h_k) = U\left(\sum_{i=1}^k c_i, \sum_{i=1}^k F_i, \sum_{i=1}^k H_i\right).$$

In particular, $\mathcal{F}_{\Delta_k}(\cdot, \varpi \delta_x)$, where ϖ is a weight, makes k weighted copies $\varpi^{1/k}\delta_x$

(vi) Backward marginalisation: suppose $h(y) = \varpi(c, F, H)\varphi(y; \mu, P)$, with $\mu = H^{-1}F$ and $P = H^{-1}$. Consider the partitioning

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu^{(1)} \\ \mu^{(2)} \end{bmatrix} \qquad P = \begin{bmatrix} P^{(11)} & P^{(12)} \\ P^{(21)} & P^{(22)} \end{bmatrix}.$$

Then

$$M^\star h(y^{(1)}, y^{(2)}) = \sqrt{\varpi(c, F, H)}\varphi(y^{(1)}; \mu^{(1)}, P^{(11)}) \odot \sqrt{\varpi(c, F, H)}\varphi(y^{(2)}; \mu^{(2)}, P^{(22)}).$$

Proof. See appendix B. □

Note that $\nu = H^{-1}F$ and $\Sigma = H^{-1}$ are the mean and covariance of the backward filtered adjoint process respectively, to be introduced more formally in section 11. Taking an inverse of H respective C can be avoided using $(Q + H^{-1})^{-1}H^{-1} = (QH + I)^{-1}, (Q + H^{-1})^{-1} = H - H(H + Q^{-1})^{-1}H$. While backward filtering for a linear Gaussian process on a tree is well known (see e.g. Chou et al. (1994), section 3), results presented in the literature often don't state update formulas for the constant ϖ (or equivalently c). If the mere goal is smoothing with known dynamics, this suffices. However, we will also be interested in estimating parameters in μ and or Q then the constant cannot be ignored.

In case the dynamics of X itself are linear, it simply follows that the weights equal 1. Moreover, if additionally the process lives on a "line graph with attached observation leaves", where each non-leaf vertex has one child in $\mathcal{S}$ and at most one child in $\mathcal{V}$, our procedure is essentially equivalent to Forward Filtering Backward Sampling (FFBS, Carter and Kohn (1994)), the difference being that our procedure applies in time-reversed order. On a general directed tree however the ordering cannot be changed and the filtering steps must be done backwards, as we propose, just like in message passing algorithms in general.

See appendix C for details how Theorem 7.1 is implemented in `Mitosis.jl`.

7.2 Discrete state-space Markov chains and particle systems

7.2.1 Branching particle on a tree

We start with a very simple example, where a "particle" takes values in a finite state space $E = \{1, \ldots, R\}$ according to the $R \times R$ transition matrix K. At each vertex, the particle is allowed to copy itself and branch.

The algorithmic elements can easily be identified. The forward kernel $\kappa(x, \mathrm{d}y)$ can be identified with the matrix K. Furthermore, the map $x \mapsto h(x)$ can be identified with the column vector $\boldsymbol{h} = [h_1, \ldots, h_R]'$, where $h_i = h(i)$. Hence h is parametrised by $\boldsymbol{h}$ and we write $h = U(\boldsymbol{h})$.

The following theorem is easily proved. It gives the necessary ingredients for applied the backward and forward map.

Theorem 7.2. Let $\kappa(x, \mathrm{d}y)$ and $\mathrm{d}\,\widetilde{\kappa}(x, \mathrm{d}y)$ be represented by the stochastic matrices K and $\widetilde{K}$ respectively. Assume that $h = U(\boldsymbol{h})$.

(i) Pullback for κ: $\kappa h = U(K\boldsymbol{h})$.

(ii) Pullback for $\widetilde{\kappa}$: $\widetilde{\kappa} h = U(\widetilde{K}\boldsymbol{h})$ and

$$\mathcal{B}_{\widetilde{\kappa}} = (m, \widetilde{\kappa} h), \quad m(x, y) = \frac{U(Kh)(y)}{U(\widetilde{K}h)(x)}.$$

Denote by M the matrix with elements $m(x, y)$, $x, y \in E$.

(iii) Fusion: if $h_i = U(\boldsymbol{h}_i)$, $i = 1, \ldots, k$, then

$$\Delta_k(h_1 \odot \cdots \odot h_k) = U\left(\bigcirc_{i=1}^{k} \boldsymbol{h}_i\right),$$

where $\bigcirc$ denotes the Hadamard (entrywise) product.

(iv) Forward map: Let μ be a (row) probability vector. If $\mathcal{F}_\kappa(m, \mu) = \nu$, then for $k \in E$

$$\nu(\{k\}) = \mu(M \bigcirc K)e_k,$$

with e_k the k-th standard basis-vector in $\mathbb{R}^R$.

(v) The weight at $x \in E$ is given by

$$w(x) = \frac{(\kappa h)(x)}{(\widetilde{\kappa} h)(x)} = \frac{\langle K\boldsymbol{h}\rangle_x}{\langle \widetilde{K}\boldsymbol{h}\rangle_x}$$

where we denote the j-th element of the vector a by $\langle a\rangle_j$.

(vi) Initialisation at the leaves: at a leaf vertex with observation $k \in E$, set $\boldsymbol{h} = e_k$.

(vii) Backward marginalisation: suppose $E = \{(x, y) \colon x = 1, \ldots, R_1, y = 1, \ldots, R_2\}$. Then h has the form

$$h = [h_{11} \cdots h_{1R_2} h_{21} \cdots h_{2R_2} \cdots\cdots h_{R_1 1} \cdots h_{R_1 R_2}]'.$$

Let

$$h_i^{(1)} = \sum_{j=1}^{R_2} h_{ij} \in \mathbf{B}(\{1, \ldots, R_1\}), \qquad h_i^{(2)} = \sum_{j=1}^{R_1} h_{ji} \in \mathbf{B}(\{1, \ldots, R_2\})$$

and $c = \sum_{i,j} h_{ij}$, then $M^\star h = h^{(1)}/\sqrt{c} \odot h^{(2)}/\sqrt{c}$.

While we can simply take $\widetilde{\kappa} = \kappa$ in this example, yielding unit weights, in case R is very large, it can nevertheless be advantageous to use a different (simpler) map $\widetilde{\kappa}$. To see, this, note that we only need to compute one element of the matrix vector product Kh, but need to compute the full vector $\widetilde{K}h$ in the backward map. Hence, choosing $\widetilde{K}$ sparse can give computational advantages.

7.2.2 Interacting particles – cellular automata – agent based models

Now consider a discrete time interacting particle process, say with n particles, where each particle takes values in $\{1, \dots, R\}$. Hence, a particle configuration x at a particular time-instant takes values in $E = \{1, \dots, R\}^n$.

A simple example consists of particles with state $E = \{1 \equiv \mathbf{S}, 2 \equiv \mathbf{I}, 3 \equiv \mathbf{R}\}$ that represent the health status of individuals that are either **S**usceptible, **I**nfected or **R**ecovered. Then the probability to transition from state **S** to **I** may depend on the number of nearby particles (individuals) that are infected, hence the particles are "interacting". A similar example is obtained from time-discretisation of the contact process (cf. Liggett (2005)). A visualisation of the DAG is given in Figure 7: time moves from left to right, vertical connects represent state changes of particles and the diagonal connections represent local interactions. Note that each particle has multiple parents defined by a local neighbourhood and that a model like this is sometimes referred to as a cellular automaton.

Let $x \in E$. Without additional assumptions, the forward transition kernel can be represented by a $R^n \times R^n$ transition matrix. With a large number of particles such a model is not attractive. To turn this into a more tractable form, we make the simplifying assumption that conditional on x each particle transitions independently. This implies that the forward evolution kernel $\kappa(x, \mathrm{d}y)$ can be represented by $(K_1(x), \dots, K_n(x))$, where $K_i(x)$ is the $R \times R$ transition matrix for the i-th particle. Note that the particles interact because the transition kernel for the ith particle may depend on the state of *all* particles.

Due to interactions, it will computationally be very expensive to use κ for the backward map. Instead, in the backward map we propose to ignore/neglect all interactions between particles. This means that effectively we use a backward map where all particles move independently, and the evolution of the i-th particle depends only on x_i (not x, as in the forward kernel). Put differently, in the backward map, we simplify the graphical model to n line graphs, one for each particle. See Figure 7.

This choice implies that $\widetilde{\kappa}$ can be represented by $(\widetilde{K}_1, \dots, \widetilde{K}_n)$ and the map $x \mapsto h(x) = \prod_{i=1}^n h_i(x)$ can be represented by $(\boldsymbol{h}_1, \dots, \boldsymbol{h}_n)$, where $\boldsymbol{h}_i = [h_{i1}, \dots, h_{iR}]'$. We write

$$h = U(\boldsymbol{h}_1, \dots, \boldsymbol{h}_n).$$

Theorem 7.3. Assume κ and $\widetilde{\kappa}$ are represented by $R \times R$ stochastic matrices $K_1, \dots, K_n$ and $\widetilde{K}_1, \dots, \widetilde{K}_n$ respectively. Assume that $h = U(\boldsymbol{h}_1, \dots, \boldsymbol{h}_n)$.

(i) Pullback for κ: $U(\kappa h)(x) = (K_1(x)\boldsymbol{h}_1, \dots, K_n(x)\boldsymbol{h}_n)$

(ii) Pullback for $\widetilde{\kappa}$: $U(\widetilde{\kappa} h) = (\widetilde{K}_1\boldsymbol{h}_1, \dots, \widetilde{K}_n\boldsymbol{h}_n)$

(iii) Parametrisation of messages: For each $i \in \{1, \dots, n\}$ let M_i be the matrix with elements

$$M_i(x, y) = \frac{U(K_i h)(y)}{U(\widetilde{K}_i h)(x)}.$$

Set $m = (M_1, \dots, M_n)$.

(iv) Forward map: Let $\mu_1, \dots, \mu_n$ be a n (row) probability vectors with μ_i representing the distribution of the i-th particle. If $\mu = \otimes_{i=1}^n \mu_i$ and $\mathcal{F}_\kappa(m, \mu) = \nu$, then $\nu = \otimes_{i=1}^n \nu_i$ where for $k \in E$

$$\nu_i(\{k\}) = \mu(M_i \bigcirc K_i(x))e_k,$$

with e_k the k-th standard basis-vector in $\mathbb{R}^R$.

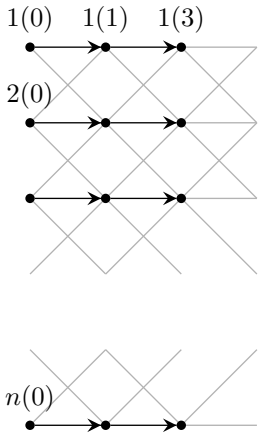


FIGURE 7: DAG for the interacting particle process of section 7.2. Particles move from left to right and the grey grid depicts the dependence structure. Here, parents of a vertex are defined by a local neighbourhood (in this case all particles at distance at most 1). For example particle 1 at time 1 (indexed by 1(1)) has parents 1(0) and 2(0). For backward filtering, the dynamics are reduced

(v) The weight at x is given by

$$w(x) = \frac{(\kappa h)(x)}{(\widetilde{\kappa} h)(x)} = \prod_{i=1}^{n} \frac{\langle K_i(x)\boldsymbol{h}_i\rangle_{x_i}}{\langle \widetilde{K}_i \boldsymbol{h}_i\rangle_{x_i}}.$$

(vi) Initialisation at the leaf: for observation $x_{M+1} \in E$ set $\boldsymbol{h}_i = e_{\langle x_{N+1}\rangle_i}$ for $i \in \{1, \dots, n\}$.

7.3 Backward diagonalisation

The example of the previous section shows a generic way to deal with interacting particles with a pattern as shown in 7. Here, conditional on the state of all particles at a particular "time", all particles transition independently, though with transition probabilities that may depend on the state of *all* particles. The backward filtering is however done on n separate line graphs, thereby fully bypassing the need of a tractable fusion step. We call this backward diagonalisation.

In case of a finite-state space, the fusion step can be done, but the computational cost is exponentially increasing with the number of time-steps. The next section provides an example where the fusion step is not tractable at all, but backward diagonalisation ensures it is not needed.

7.4 Independent Gamma increments

In this example, we consider a process with Gamma distributed increments. We consider a line graph (where $|\mathrm{ch}(s)| = 1$ for $s \in \mathcal{S}_0$) with a single observational leaf v, attached to the end of the single path in $\mathcal{S}_0$. We write $Z \sim \mathrm{Gamma}(\alpha, \beta)$ if Z has density $\psi(x; \alpha, \beta) = \beta^\alpha \Gamma(\alpha)^{-1} x^{\alpha-1} e^{-\beta x} \mathbf{1}_{(0,\infty)}(x)$.

Choose mappings α_t, $\beta_t \colon (0, \infty) \to [1/C, C]$ for some $C > 0$. We define a Markov process with Gamma increments on $\mathcal{G}$ by

$$X_t - X_s \mid X_s = x \sim \mathrm{Gamma}(\alpha_t, \beta_t(x)), \quad \text{if } s = \mathrm{pa}(t), \qquad X_0 = x_0.$$

This implies

$$p_{\to t}(x; y) = \psi(y - x; \alpha_t, \beta_t(x)).$$

Note that X can be thought of as a time discretised version of the SDE $\mathrm{d}X_t = \beta^{-1}(X_t)\,\mathrm{d}L_t$ driven by a Gamma process (L_t) with scale parameter 1, observed at final time T. See Belomestny et al. (2020) for a continuous time perspective on this problem. A statistical application will typically involve multiple observations and henceforth multiple line graphs. Simulation on each line graph corresponds to conditional (bridge) simulation for the process X.

As the h-transform is not tractable, we introduce the process $\widetilde{X}$ which is defined as the Markov process with Gamma increments on $\mathcal{G}$ induced by (α_t, β), with β a user specified nonnegative constant. This process is tractable and is used to define the $\widetilde{h}$-transform.

As before, we state our results using the kernels κ and $\widetilde{\kappa}$. Define for $A, \beta > 0$

$$h(y) = \psi(x_v - y; A, \beta) \tag{7.3}$$

and write $h = U(A, \beta)$.

Define the exponentially-tilted Beta-distribution with parameters $\gamma_1, \gamma_2 > 0$ and $\lambda \in \mathbb{R}$ as the distribution with density

$$q_{\gamma_1,\gamma_2,\lambda}(z) \propto z^{\gamma_1 - 1}(1 - z)^{\gamma_2 - 1} e^{-\lambda z} \mathbf{1}_{(0,1)}(z). \tag{7.4}$$

We denote this distribution by $\mathrm{ExpBeta}(\gamma_1, \gamma_2, \lambda)$. Sampling from this distribution can be accomplished for example using rejection sampling, with importance sampling distribution $\mathrm{Beta}(\gamma_1, \gamma_2)$.

Theorem 7.4. Let $\kappa(x,\, \mathrm{d}y) = \psi(y - x; \alpha, \beta(x))\,\mathrm{d}y$ and $\widetilde{\kappa}(x,\, \mathrm{d}y) = \psi(y - x; \alpha, \beta)\,\mathrm{d}y$. Assume that $h = U(A, \beta)$.

(i) Pullback for κ:

$$(\kappa h)(x) = U(A + \alpha, \beta)(x) \left(\frac{\beta(x)}{\beta}\right)^{\alpha} \mathbb{E}\, e^{-\xi(x)Z},$$

where $Z \sim \mathrm{Beta}(\alpha, A)$ and $\xi(x) = (\beta(x) - \beta)(x_v - x)$.

(ii) Pullback for $\widetilde{\kappa}$: $(\widetilde{\kappa}h)(x) = U(A + \alpha, \beta)(x)$.

(iii) Forward map: if $\mu = \delta_x$, then a draw y from $\mathcal{F}_\kappa(m, \mu)$ can be generated by first drawing $z \sim \mathrm{ExpBeta}(\alpha, A, \xi(x))$ and then setting $y = x + z(x_v - x)$.

(iv) The weight at x is given by

$$w(x) = \frac{(\kappa h)(x)}{(\widetilde{\kappa}h)(x)} = \left(\frac{\beta(x)}{\beta}\right)^{\alpha} \mathbb{E}\, e^{-\xi(x)Z}$$

where $Z \sim \mathrm{Beta}(\alpha, A)$

Proof. See appendix B. □

Note that the expression for the weight immediately suggests a method to estimate $w(x)$ unbiasedly. The setting is restricted to a line graph, as fusion of h_1 and h_2 of the form (7.3) does not lead to a fused function of the same form.

The model can be extended to n Markov processes with Gamma increments, which evolve conditionally independent, where it is assumed that the forward evolution consists of composing kernels

$$\kappa(\boldsymbol{x},\, \mathrm{d}\boldsymbol{y}) = \prod_{i=1}^{n} \psi(\boldsymbol{y}_i - \boldsymbol{x}_i; \alpha, \beta(\boldsymbol{x}))\,\mathrm{d}\boldsymbol{y}.$$

The backward kernel $\widetilde{\kappa}$ is then taken to be the same, with $\beta(\boldsymbol{x})$ replaced with β. This is an instance of backward diagonalisation.

Remark 7.5. The setting of this section can be generalised: suppose for $s = pa(t)$ that

$$X_t - X_s \mid X_s = x \sim \mathcal{D}(\theta_t(x)),$$

with $\mathcal{D}(\theta)$ a distribution with parameter θ supported on $(0, \infty)$, say with density $\psi(\cdot; \theta)$.

Tracing back the proof, the derivation of κh shows that if $\kappa(x, \mathrm{d}y) = \psi(y-x; \theta)\,\mathrm{d}y$, then backward filtering is tractable if

$$(\kappa h)(x) = \int \psi(x_v - y; \theta)\psi(y - x; \theta(x))\,\mathrm{d}y = \int \psi(x_v - x - z; \theta)\psi(z; \theta(x))\,\mathrm{d}z$$

can be written as $c\psi(x_v - x; \widetilde{\theta}(x))$ for some $\widetilde{\theta}(x)$. Hence, the results can be extended to the case where increments are closed under convolution. The Gamma distribution has this property, but so does the Inverse Gaussian $\mathrm{IG}(\mu, 2\mu^2)$-distribution for example.

8 Sampling and statistical inference using guided processes

In this section we discuss a variety of ways in which Theorem 4.6 can be used. We will make the simplifying assumption that h and $\widetilde{h}$ match on edges (s, v) with $s \in \mathrm{pa}(v)$ and $v \in \mathcal{V}$. This means that the product term over $v \in \mathcal{V}$ (appearing in the statement of the theorem) equals 1. We then take as a starting point that on a DAG

$$p^\star(x_\mathcal{S} \mid x_0) = \frac{\Psi(x_\mathcal{T})}{h_0(x_0)} \prod_{s \in \mathcal{S}} p^\circ_{\to s}(x_{\mathrm{pa}(s)}; x_s),$$

where

$$\Psi(x_\mathcal{T}) := \widetilde{h}_0(x_0) \prod_{s \in \mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)}) \tag{8.1}$$

(Theorem 4.6). In this expression $h_0(x_0)$ will virtually always be intractable. Our standing assumption is that we can define $\widetilde{h}_{\to s}(x)$ for all $s \in \mathcal{S}$ such that $\widetilde{h}_s(x)$ can be computed efficiently for all $s \in \mathcal{S}_0$. Depending on the the dynamics of X and $\widetilde{h}$, it may then be possible to easily forward sample X° from root vertices towards leaf vertices (for example, in Gaussian nonlinear smoothing this is the case, cf. section 7.1). Then, again depending on X and $\widetilde{h}$ the weights w appearing in $\Psi(x_\mathcal{T})$ may be tractable or not.

The right-hand-side of (8.1) is obtained by multiplying the evidence obtained from the backward pass and multiplying it with the obtained after running the forward pass.

In the following, we denote the law of the process under X° and $X^\star$ by $\mathbb{P}^\circ$ and $\mathbb{P}^\star$ respectively. There are links with a variety of popular methods from computational statistics:

8.1 Importance sampling

Suppose interest lies in $I := \mathbb{E}\varphi(X^\star)$. We simulate independent copies $\{X^{\circ,i},\, i = 1, \dots B\}$ under $\mathbb{P}^\circ$. Next, define the importance sampling estimator

$$\hat{I} := \sum_{i=1}^{B} w_i \varphi\left(X^{\circ,i}\right), \qquad w_i = \frac{\Psi(X^{\circ,i})}{\sum_{i=1}^{B} \Psi(X^{\circ,i})}.$$

Note that w_i does not contain terms of the form $h_0(x_0)$ and the estimator requires that the weights can be computed efficiently (if this is not the case, this can however be addressed by Pseudo-Marginal Metropolis-Hastings, see below).

The evidence $h_0(x_0)$ can also be approximated by Monte-Carlo sampling since $h_0(x_0) = \mathbb{E}\,\Psi(X^\circ)$.

8.2 Variational inference

If $\widetilde{h}$ is indexed by a parameter $\eta \in D$, then it induces a family of laws $\mathcal{P} = \{\mathbb{P}^{\circ(\eta)}, \eta \in D\}$ which can be used as a variational class in variational inference. That is, we find an element in $\mathcal{P}$ that approximates $\mathbb{P}^\star$ by the information projection. More precisely, we aim to find $\operatorname{argmin}_{\eta\in D} \mathrm{KL}(\mathbb{P}^{\circ(\eta)}, \mathbb{P}^\star)$, KL denoting Kullback-Leibler divergence. The minimisation can be done using stochastic gradient descent, which requires estimating $\nabla_\eta \mathrm{KL}(\mathbb{P}^{\circ(\eta)}, \mathbb{P}^\star)$. Typically, the proposal process can be reparametrised as

$$X^{\circ(\eta)} = g(\eta, Z) \tag{8.2}$$

for a deterministic map and a parameter-free innovation process Z (for example white noise), say with law $\mathbb{Q}$. This enables variational inference with the "reparametrisation trick" (Kingma and Welling, 2014). More precisely, define $\mathbb{Q}^\star$ by

$$\frac{\mathrm{d}\mathbb{Q}^\star}{\mathrm{d}\mathbb{Q}}(Z) = \frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}^\circ}(g(\eta, Z)).$$

By the transformation formula, samples $Z \sim \mathbb{Q}^\star$ give $g(\eta, Z) \sim \mathbb{P}^\star$. Since

$$\nabla_\eta \mathbb{E}^{\circ(\eta)} \log \frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}^{\circ(\eta)}}(X^{\circ(\eta)}) = \nabla_\eta \mathbb{E}^{\mathbb{Q}} \log \frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}^{\circ(\eta)}}(g(\eta, Z)) = \mathbb{E}^{\mathbb{Q}} \nabla_\eta \log \frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}^{\circ(\eta)}}(g(\eta, Z))$$

an estimator for this gradient can be obtained by sampling from $\mathbb{Q}$ (we implicitly assume expectation and differentiation can be interchanged).

Note that the construction of $X^{\circ(\eta)}$ in (8.2) is a normalising flow, see for instance Rezende and Mohamed (2015) and Kobyzev et al. (2020). Alternatively, it can be seen as a noncentred parametrisation (Papaspiliopoulos et al., 2003) or randomness pushback (Fritz, 2020).

8.3 Markov chain Monte Carlo methods

Suppose we can reparametrise $X^\circ = g(Z)$ where Z is standard normal. Then a Metropolis-Hastings algorithm for sampling from $\mathbb{P}^\star$ can be constructed by sampling $W \sim N(0, I)$ (independently from Z), choosing $\alpha \in [0, 1)$ and proposing

$$Z_{\mathrm{new}} = \alpha Z + \sqrt{1 - \alpha^2} W.$$

Next, Z_{new} is accepted with probability

$$1 \wedge \frac{\Psi(g(Z_{\mathrm{new}}))}{\Psi(g(Z))}$$

and upon acceptance X° is updated to $g(Z_{\mathrm{new}})$. The update step for Z_{new} is often referred to as a preconditioned Crank-Nicolson (pCN) step.

In evaluating $\Psi(x_{\mathcal{T}})$, the product of weights, $\prod_{s\in\mathcal{S}} w_{\to s}(x_{\mathrm{pa}(s)})$ may be intractable. The Pseudo-Marginal Metropolis-Hastings (PMMH) algorithm (Andrieu and Roberts, 2009) is based on the

observation that if $\Psi(x_{\mathcal{T}})$ is replaced with an unbiased estimator, the Metropolis-Hastings algorithm still targets the exact target distribution. Here, this algorithm can be used upon realising that $p^\star(x_{\mathcal{S}} \mid x_0)$ is the marginal density of

$$\pi(x_{\mathcal{S}}, y_{\mathcal{S}} \mid x_0) = \frac{\widetilde{h}_0(x_0)}{h_0(x_0)} \prod_{s \in \mathcal{S}} \frac{\widetilde{h}_s(y_s)}{\prod_{u \in \mathrm{pa}(s)} \widetilde{h}_{u \to s}(x_u)} p^\circ_{\to s}(x_{\mathrm{pa}(s)}; x_s) p_{\to s}(x_{\mathrm{pa}(s)}; y_s).$$

Therefore, a Metropolis-Hastings algorithm targeting π is free from intractable weights and the $y_{\mathcal{S}}$ samples can simply be dropped.

8.4 Hamiltonian Monte Carlo

This is also helpful for derivative-based samplers such as Hamiltonian Monte Carlo. Consider the situation where $X^\circ = g(Z)$, for $Z \sim \mathbb{Q} = \mathrm{N}(0, \mathrm{I})$.

The Hamiltonian to be used in a Hamiltonian Monte Carlo sampler targeting $Z^\star$ is

$$\mathrm{H}(\mathsf{z}, \mathsf{p}) = -\log \frac{\mathrm{d}\mathbb{Q}^\star}{\mathrm{d}\mathbb{Q}} \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\lambda}(\mathsf{z}) + \frac{1}{2}\|\mathsf{p}\|,$$

where p is the momentum vector and λ denotes the Lebesgue measure. Thus with

$$\psi_s(\mathsf{z}) = -\frac{\partial}{\partial \mathsf{z}_s} \log \frac{\mathrm{d}\mathbb{Q}^\star}{\mathrm{d}\mathbb{Q}} \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\lambda}(\mathsf{z}) = -\sum_{t \in \mathcal{S}} \frac{\partial}{\partial \mathsf{z}_s} \log w_{\to t}(g(\mathsf{z}_{\mathrm{pa}(t)}) - \mathsf{z}_s.$$

the coordinate functions of the gradient of the negative score (the potential energy) and with λ the Lebesgue measure, the Hamiltonian dynamics are given by

$$\frac{\partial \mathsf{p}_s}{\partial t} = -\frac{\partial \mathrm{H}(\mathsf{z}, \mathsf{p})}{\partial \mathsf{z}_s} = -\psi_s(\mathsf{z}), \quad \frac{\partial \mathsf{z}_s}{\partial t} = \frac{\partial \mathrm{H}(\mathsf{z}, \mathsf{p})}{\partial \mathsf{p}_s} = \mathsf{p}_s.$$

Thus a Hamiltonian Monte Carlo method to sample $Z^\star$ can be used and samples of the target $X^\star$ can then obtained via transformation $g(Z^\star)$.

8.5 Parameter dependence

In a statistical context, the dynamics of X, or more specifically, the transition densities $p^\theta_{\to t}(x_{\mathrm{pa}(t)}; x_t)$ depend on an unknown parameter $\theta \in \Theta$ and $x_{\mathcal{V}}$ is observed and the objective is to infer θ. This setup implies that h depends on θ as well. Writing $h = h^\theta$, we then have that $h^\theta_0(x_0)$ is the likelihood of θ with latent X integrated out,

$$h^\theta_0(x_0) = \widetilde{h}_0(x_0)\, \mathbb{E} \prod_{s \in \mathcal{S}} w_{\to s}(X^\circ_{\mathrm{pa}(s)}),$$

where in the expectation we have disregarded the possible dependency of the proposal process on θ.

Alternatively, this can be resolved by equipping θ with a prior, and augmenting the states to $x^{\mathrm{aug}}_s = (x_s, \theta_s)$ for all $s \in \mathcal{S}$, and setting for $t \in \mathrm{ch}(s)$

$$\mathbb{P}^{\mathrm{aug}}(\mathrm{d}(x_t, \theta_t) \mid (x_s, \theta_s)) = p_{s \to t}(x_s; x_t) \lambda(\mathrm{d}x_t) \delta_{\theta_s}(\mathrm{d}\theta_t)$$

where δ is the Dirac measure. Thus $\theta_t = \theta_s$ with probability 1. This is one reason to allow the conditional distribution to be expressed as Radon-Nikodym derivative with respect to a general reference transition kernel $\lambda(x_s, \mathrm{d}x_t)$. Close inspection of the preceding section shows that this is possible.

9 Guided processes for continuous-time Markov processes

Up to this point, we have assumed edges to represent "discrete time" Markov-transitions. In some applications it is natural to interpret the transition over an edge as the result of evolving a continuous-time Markov process over a fixed time interval. Let (X_t) be a time-homogeneous E-valued Markov process defined on $[a, b]$ with full (extended) generator $\mathcal{L}$ with domain $\mathcal{D}(\mathcal{L})$. Let $\{\mathcal{F}_t\}$ be a right-continuous history filtration and let $\mathbb{P}_t$ denote the restriction of $\mathbb{P}$ to $\mathfrak{F}_t$. X defined on the interval $[S, T]$ has a family of evolution kernels $\kappa_{s,t}$ with transition densities p.

$$\mathbb{P}(X_t \in \mathrm{d}y \mid X_s = x) = \kappa_{s,t}(x, \mathrm{d}y) = p(s, x; t, y)\,\mathrm{d}y$$

To connect to the discrete setting, we could assume that $S', T' \in \mathcal{T}$ (using capital letters for vertices here), such that $S' = \mathrm{pa}(T')$ the transition from parent to child is specified by the transition kernel $\kappa_{S,T}$ of the process X on a time interval $[S, T]$. Slightly abusing notation, we identify begin and end time $[S, T]$ with source and target vertices S', T' of the edge considered and omit the prime symbol.

The discrete pullback of a function h_T defined on $\mathbf{B}(E)$ by $\kappa = \kappa_{S,T}$ extends on the interval $t \in [S, T]$ to a continuous h-transform by setting

$$h(t, x) = (\kappa_{t,T} h_T)(x) = \mathbb{E}[h_T(X_T) \mid X_t = x], \qquad t \in (S, T]. \tag{9.1}$$

Suitable $h(t, x)$ satisfiy the Kolmogorov backward equation

$$\frac{\partial}{\partial t} h(t, x) + \mathcal{L}h(t, x) = 0, \qquad t \in (S, T]. \tag{9.2}$$

Only for few Markov processes it is possible to solve this equation in closed form leading to a tractable expression for h. However, just as in the discrete-time case, we can consider the $\widetilde{h}$-transform and define guided processes. This results in a generalisation of the guided proposals from Schauer et al. (2017), which are restricted to diffusion processes.

9.1 Construction of guided processes for continuous-time Markov processes

Without loss of generality, we assume the process (X_t) is defined on $[0, T]$. We consider a fixed initial value x_0. The space-time process (t, X_t) has full generator $\mathcal{A} = \mathcal{L} + \frac{\partial}{\partial t}$ extending the graph

$$\{(fh, \mathcal{A}(fh)), f \in \mathcal{D}(\mathcal{L}), h \in C[0, T])\},$$

see Theorem 7.1 on page 221 in Ethier and Kurtz (1986).

Define the change of measure

$$\mathrm{d}\mathbb{P}_t^\star = D_t^h \,\mathrm{d}\mathbb{P}_t,$$

with

$$D_t^h = \frac{h(t, X_t)}{h(0, x_0)} \exp\left(-\int_0^t \frac{\mathcal{A}h}{h}(s, X_s)\,\mathrm{d}s\right).$$

Here, we implicitly assume $h \in \mathcal{D}(\mathcal{A})$ is a positive function such that D_t^h is a $\mathfrak{F}_t$-martingale. As in Palmowski and Rolski (2002), we will simply call such a function a *good* function.

Definition 9.1. Let $\widetilde{h}$ be a good function define the change of measure

$$\mathrm{d}\mathbb{P}_t^\circ = D_t^{\widetilde{h}} \,\mathrm{d}\mathbb{P}_t.$$

The process X, with $X_0 = x_0$, under the law $\mathbb{P}_t^\circ$ is denoted by X° and referred to as the *guided process induced by* $\widetilde{h}$.

By formula (1.2) in Palmowski and Rolski (2002) the full generator of X° has the form

$$\mathcal{A}^\circ f = \frac{1}{\widetilde{h}}\left[\mathcal{A}(f\widetilde{h}) - f\mathcal{A}\widetilde{h}\right] \tag{9.3}$$

which characterises the dynamics of X°. This result will be used in multiple examples.

Theorem 9.2. Suppose h and $\widetilde{h}$ good functions and h is space-time harmonic for $\mathcal{A}$, i.e. $\mathcal{A}h = 0$. Then

$$\frac{\mathrm{d}\mathbb{P}^\star_t}{\mathrm{d}\mathbb{P}^\circ_t}(X^\circ) = \frac{h(t, X^\circ_t)\widetilde{h}(0, x_0)}{\widetilde{h}(t, X^\circ_t)h(0, x_0)} \exp\left(\int_0^t \frac{\mathcal{L}\widetilde{h} + \frac{\partial\widetilde{h}}{\partial s}}{\widetilde{h}}(s, X^\circ_s)\,\mathrm{d}s\right).$$

Additionally, suppose $\widetilde{X}_t$ is a Markov process under $\mathbb{P}_t$ with generator $\widetilde{\mathcal{L}}$. If $\widetilde{h}$ is space-time harmonic for $\widetilde{\mathcal{A}} = \widetilde{\mathcal{L}} + \frac{\partial}{\partial t}$, then

$$\frac{\mathrm{d}\mathbb{P}^\star_t}{\mathrm{d}\mathbb{P}^\circ_t}(X^\circ) = \frac{h(t, X^\circ_t)\widetilde{h}(0, x_0)}{\widetilde{h}(t, X^\circ_t)h(0, x_0)} \exp\left(\int_0^t \frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}}{\widetilde{h}}(s, X^\circ_s)\,\mathrm{d}s\right). \tag{9.4}$$

Proof. First note that

$$\frac{\mathrm{d}\mathbb{P}^\star_t}{\mathrm{d}\mathbb{P}^\circ_t}(X^\circ) = \left(\frac{\mathrm{d}\mathbb{P}^\star_t}{\mathrm{d}\mathbb{P}_t} \Big/ \frac{\mathrm{d}\mathbb{P}^\circ_t}{\mathrm{d}\mathbb{P}_t}\right)(X^\circ) = \frac{D^h_t}{D^{\widetilde{h}}_t}.$$

As h is space-time harmonic for $\mathcal{A}$ we have

$$D^h_t = \frac{h(t, X^\circ_t)}{h(0, x_0)}.$$

The proof follows from

$$\begin{aligned} D^{\widetilde{h}}_t &= \frac{\widetilde{h}(t, X^\circ_t)}{\widetilde{h}(0, x_0)} \exp\left(-\int_0^t \frac{\mathcal{L}\widetilde{h} + \frac{\partial\widetilde{h}}{\partial s}}{\widetilde{h}}(s, X^\circ_s)\,\mathrm{d}s\right) \\ &= \frac{\widetilde{h}(t, X^\circ_t)}{\widetilde{h}(0, x_0)} \exp\left(-\int_0^t \frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}}{\widetilde{h}}(s, X^\circ_s)\,\mathrm{d}s\right) \end{aligned}$$

where we used that $\widetilde{h}$ is space-time harmonic for $\widetilde{\mathcal{L}}$, and therefore $\frac{\partial\widetilde{h}}{\partial s} = -\widetilde{\mathcal{L}}\widetilde{h}$. □

The theorem is indeed a continuous-time analogue of Theorem 4.3, with $\widetilde{A}\widetilde{h} = 0$ corresponding to (3.2). Comparing the exponentiated integral in the likelihood ratio to its discrete analogue, the product of w_t in Corollary 4.4 makes the analogy very apparent. We again encounter the situation that in case of tractable observations in the sense of Corollary 4.4, $h(T, \cdot)$ in (9.1) can be chosen as $\widetilde{h}(T, \cdot)$ such that the ratio $h(T, X^\circ_T)/\widetilde{h}(T, X^\circ_T)$ cancels.

A second way to use Theorem 9.1 is in case when $h(t, X^\circ_t)$ and $\widetilde{h}(t, X^\circ_t)$ approach the same Dirac measure for $t \to T$. Situations like this arise when sampling Markov processes conditional on exact observations. This entails to show that the first ratio on the-right hand-side of equation (9.4) converges to 1 upon taking the limit $t \uparrow T$. In the setting where X is defined by a stochastic differential equation sufficient conditions have been derived in Schauer et al. (2017) and Bierkens et al. (2020). Now suppose that the limit is well defined and the first term on the right-hand-side of (9.4) can be "removed" from the expression. The resulting expression then still contains the generally intractable term $h(0, x_0)$. However, when we condition on $X_0 = x_0$ and interest lies in the smoothing distribution this is simply a harmless multiplicative constant.

In parameter estimation problems, where the dynamics of the Markov process depend on an unknown parameter θ, $h(0, x_0)$ will also depend on θ. However, as explained in section 4 of van der Meulen and Schauer (2017), in the acceptance probability of MCMC-update steps for θ, this term will cancel out. The argument is similar to equation 4.3.

Remark 9.3. Note that X and $\widetilde{X}$ are not assumed to have absolutely continuous laws. However, if that is the case, also the proposal process $\widetilde{X}^\star$ with generator

$$\widetilde{\mathcal{A}}^\star f = \frac{1}{\widetilde{h}}\left[\widetilde{\mathcal{A}}(f\widetilde{h}) - f\widetilde{\mathcal{A}}\widetilde{h}\right]$$

and law $\widetilde{\mathbb{P}}_t^\star$ can be used. In this case the change of measure follows from the "abstract Bayes formula"

$$\frac{\mathrm{d}\mathbb{P}_t^\star}{\mathrm{d}\widetilde{\mathbb{P}}_t^\star} = \frac{\mathrm{d}\mathbb{P}_t}{\mathrm{d}\widetilde{\mathbb{P}}_t}\frac{\widetilde{h}(0,x_0)}{h(0,x_0)}.$$

10 Examples of guided processes for continuous-time Markov processes

10.1 Stochastic differential equations

Suppose the Markov process X is defined as a solution to the stochastic differential equation

$$\mathrm{d}X_t = b(t, X_t)\,\mathrm{d}t + \sigma(t, X_t)\,\mathrm{d}W_t.$$

Here σ is possibly non-constant, as is often the case for physically meaningful local noise structure, for example for those obtained through the stochastically constrained variational principle, Holm (2015).

Assume the approximate Doob-h-transform is derived from the simpler process $\mathrm{d}\widetilde{X}_t = \widetilde{b}(t, \widetilde{X}_t)\,\mathrm{d}t + \widetilde{\sigma}(t, \widetilde{X}_t)\,\mathrm{d}W_t$ (for example by taking a linear drift and constant diffusion coefficient). In this case the term appearing in the exponent of the likelihood ratio in Theorem 9.2 satisfies

$$\frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}}{\widetilde{h}} = \sum_i (b_i - \widetilde{b}_i)\frac{\partial_i \widetilde{h}}{\widetilde{h}} + \frac{1}{2}\sum_{i,j}(a_{ij} - \widetilde{a}_{ij})\frac{\partial_{ij}\widetilde{h}}{\widetilde{h}},$$

where we have omitted arguments (s, x) from the functions and ∂_i denotes the partial derivative with respect to x_i. If we set $\widetilde{r}_i = (\partial_i \widetilde{h})/\widetilde{h}$, then $(\partial_{ij}\widetilde{h})/\widetilde{h} = \partial_j \widetilde{r}_i + \widetilde{r}_i \widetilde{r}_j$ and therefore

$$\frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}}{\widetilde{h}} = \sum_i (b_i - \widetilde{b}_i)\widetilde{r}_i + \frac{1}{2}\sum_{i,j}(a_{ij} - \widetilde{a}_{ij})\left(\partial_j \widetilde{r}_i + \widetilde{r}_i \widetilde{r}_j\right).$$

This expression coincides with the expression for the likelihood given in Proposition 1 of Schauer et al. (2017), but the proof given here is much shorter.

In case the process $\widetilde{X}$ defines a linear SDE, then solving the Kolmogorov backwards equation on $[0, T]$ reduces to solving a system of ordinary differential equations. Theorem 3.3 in Mider et al. (2020) gives these ODEs in their "information filter" form (in that paper c is defined with the opposite sign). The output of these equations is fully compatible with the form of h in subsection 7.1. In particular, fusion is tractable and our approach enables to simulate guided processes for conditioned branching diffusions.

The corresponding rules for `Mitosis.jl` are provided by `MitosisStochasticDiffEq.jl` using solversfrom Julia's SciML[7] ecosystem for Scientific Machine Learning (Rackauckas and Nie, 2017).

[7] sciml.ai.

10.2 First hitting time of a diffusion

Doob's h-transforms are not restricted to conditioning on the value of a process at a fixed time. Rather, they can generally be applied for different types of conditioning such as for example conditioning on the first hitting time of a set. Accordingly, guided processes can be used for general conditionings.

As a concrete example, suppose that over an edge a one-dimensional diffusion process instead of evolving for a fix time, evolves until it first hits level v, v being considered fixed and the hitting time is observed.

Suppose the dynamics of the diffusion are governed by the SDE $\mathrm{d}X_t = b(X_t)\,\mathrm{d}t + \mathrm{d}W_t$. Let τ_v denote the first hitting time of v. Now $\mathbb{P}_{s,x}(\tau_v \in \mathrm{d}T)$ itself is intractable. However, if $b \equiv 0$ we may use

$$\widetilde{h}(t,x) = \frac{|v-x|}{2\pi(T-t)^3} e^{-(v-x)^2/(2(T-t))}\,\mathrm{d}T$$

as approximate h-transform and derive a guided process and likelihood.

It is easily verified that $\widetilde{h}$ is space-time harmonic,

$$\frac{\partial}{\partial t}\widetilde{h}(t,x) - \frac{1}{2}\frac{\partial^2}{\partial x^2}\widetilde{h}(t,x) = 0.$$

Furthermore

$$\nabla \log \widetilde{h}(t,x) = \frac{v-x}{T-t} + \frac{1}{x-v},$$

which implies

$$\mathcal{L}^\circ f(x) = \left(b(x) + \frac{v-x}{T-t} + \frac{1}{x-v}\right) f'(x) + \frac{1}{2} f''(x).$$

Note that in case $b \equiv 0$, this is the generator of a (shifted and scaled) Bessel bridge.

In the larger picture of continuous processes embedded into graphical models, the observation of such a hitting time may correspond to an "observation operation" (a leaf in the DAG) with transition density $p(t \mid s,x) = \frac{|v-x|}{2\pi(T-s)^3} e^{-(v-x)^2/(2(T-s))}\,\mathrm{d}T$ (using Bayesian notion).

10.3 Continuous time Markov chains on a discrete lattice

Consider a continuous time Markov chain taking values in a subset E of a discrete lattice in $\mathbb{R}^d$ such that

$$\mathbb{P}(X_{t+\Delta} - X_t = \xi_\ell \mid \mathcal{F}_t) = \lambda_\ell(X_t)\Delta + o(\Delta), \quad \Delta \downarrow 0,$$

with $\mathcal{F}_t$ the natural filtration generated by the process X and λ_ℓ specifying the intensity of jumps of size ξ_ℓ. As detailed in chapter 1 of Anderson and Kurtz (2015), the process X can be constructed by starting off with independent unit intensity Poisson processes Y_ℓ and subsequently defining it by the relation

$$X_t = X_0 + \sum_\ell \xi_\ell Y_\ell \left(\int_0^t \lambda_\ell(X_s)\,\mathrm{d}s \right). \tag{10.1}$$

Define the generator

$$\mathcal{L}f(x) = \sum_\ell \lambda_\ell(x)\,(f(x+\xi_\ell) - f(x))$$

Theorem 1.22 in Anderson and Kurtz (2015) gives conditions under which the solution to (10.1) is the unique minimal solution of the martingale problem for $\mathcal{L}$ (one condition on the intensities

is that for all $x \in E$ $\sum_\ell \lambda_\ell(x) < \infty$). This encompasses that there is a filtration $\mathcal{F}_t$ such that for each f with finite support $f(X_t) - f(X_0) - \int_0^t (\mathcal{L}f)(X_s)\,\mathrm{d}s$ is a $\mathcal{F}_t$-martingale.

Applying (9.3) gives

$$\mathcal{L}_t^\circ f(x) = \sum_\ell \lambda_\ell(x) \frac{\widetilde{h}(t, x+\xi_\ell)}{\widetilde{h}(t,x)} \left[f(x+\xi_\ell) - f(x)\right].$$

This shows that the guided process is time-inhomogeneous and given by $X_t^\circ = X_0^\circ + \sum_\ell \xi_\ell Y_\ell \left(\int_0^t \lambda_\ell^\circ(s, X_s^\circ)\,\mathrm{d}s\right)$ with

$$\lambda_\ell^\circ(s,x) = \lambda_\ell(x) \frac{\widetilde{h}(s, x+\xi_\ell)}{\widetilde{h}(s,x)}.$$

Exact simulation of of continuous time Markov chains with time dependent intensity functions can for example be done using the next reaction method (section 5.3.2.1 in Anderson and Kurtz (2015)).

One tractable candidate for $\widetilde{h}$ is to used the Linear Noise Approximation to the Markov chain (cf. Fearnhead et al. (2014)). The following example considers the much simpler setting of a non homogeneous Poisson process.

Example 10.1. A particularly simple case is obtained if $\xi_1 = 1$ and $\lambda_\ell = 0$ for all $\ell \geq 2$. This corresponds to an inhomogeneous Poisson process. In that case, a simple choice for the auxiliary process $\widetilde{X}$ is a homogenous Poisson process of rate $\widetilde{\lambda}$.

For a Poisson process $\widetilde{X}$ with constant intensity $\widetilde{\lambda}$ we have

$$\widetilde{p}(t,x;T,x_T) = e^{-\widetilde{\lambda}(T-t)} \frac{(\widetilde{\lambda}(T-t))^{x_T - x}}{(x_T - x)!}, \qquad x_T \geq x.$$

Define $\widetilde{h}(t,x) = \widetilde{p}(t,x;T,x_T)$, then $\widetilde{h}$ is space-time harmonic for $(t, \widetilde{X}_t)$. This shows that the guided process is a Poisson process with stochastic intensity

$$\lambda^\circ(t,x) = \frac{\lambda(x)}{\widetilde{\lambda}} \frac{x_T - x}{(T-t)},$$

a result already derived in chapter 4 of Marchand (2012), albeit derived differently. The integrand appearing in (9.4) is given by

$$\frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}(s,x)}{\widetilde{h}(s,x)} = \frac{\lambda(x) - \widetilde{\lambda}}{\widetilde{h}(s,x)} (\widetilde{h}(s,x+1) - \widetilde{h}(s,x)) = (\lambda(x) - \widetilde{\lambda}) \left(\widetilde{\lambda}^{-1} \frac{x_T - x}{T-s} - 1\right)$$

Preliminary results jointly with Marc Corstanje indicate that taking $\widetilde{\lambda}$ such that $\widetilde{\lambda} \leq \inf_x \lambda(x)$ suffices to ensure that law of the conditioned process is absolutely continuous to the law of the guided process.

In more complicated settings, finding a tractable $\widetilde{h}$ may be more difficult. In case $E = \mathbb{Z}^d$ one may try to use a diffusion approximation to find a such a mapping, for example $\widetilde{h}$ derived from $\mathrm{d}\widetilde{X}_t = \widetilde{\mu}\,\mathrm{d}t + \widetilde{\sigma}\,\mathrm{d}W_t$.

10.4 Continuous time Markov chains on a countable set

Let X denote a continuous time Markov process taking values in the countable set E. Let $\mathcal{Q} = (q(x,y),\, x,y \in E)$ be a Q-matrix, i.e. its elements $q(x,y)$ satisfy $q(x,y) \geq 0$ for all $x \neq y$

and $\sum_y q(x,y) = 0$. Define $c(x) = -q(x,x)$. If $\sup_x c(x) < \infty$, then $\mathcal{Q}$ unique defines a continuous time Markov chain with values in E. Its transition probabilities $p_t(x,y)$ are continuously differentiable in t for all x and y and satisfy Kolmogorov's backward equations:

$$\frac{\mathrm{d}}{\mathrm{d}t} p_t(x,y) = \sum_x q(x,z) p_t(z,y)$$

(cf. Liggett (2010), chapter 2, in particular Corollary 2.34). The infinitesimal generator acts upon functions $g\colon E \to \mathbb{R}$ by

$$\mathcal{L}g(x) = \sum_{y \in E} q(x,y)\,(g(y) - g(x)) = \sum_{y \in E} q(x,y) g(y), \qquad t \in [0,\infty), \quad x \in E. \tag{10.2}$$

In the specific setting where E is finite with states labeled by $1, 2, \ldots, R$, $\mathcal{Q}$ is an $R \times R$-matrix. Then, by evaluating $\mathcal{L}g(x)$ for each $x \in E$ we can compactly write $\mathcal{L}\boldsymbol{g} = \mathcal{Q}\boldsymbol{g}$, where $\boldsymbol{g}$ is the column vector with i-th element $\boldsymbol{g}_i = g(i)$.

Suppose an approximate h transform $\widetilde{h}\colon [0,\infty) \times E \to \mathbb{R}$ is specified. It induces a guided process X° for which the generator of the space-time process can be derived using equation (9.3) and the preceding display. Some simple calculations reveal that for $f\colon [0,\infty) \times E \to \mathbb{R}$

$$\widetilde{h}(t,x)(\mathcal{A}^\circ f)(t,x) = \sum_{y \in E} q(x,y)\,(f(t,y) - f(t,x))\,\widetilde{h}(t,y) + \widetilde{h}(t,x)\frac{\partial f}{\partial t}(t,x)$$

This implies that the generator of X° acts upon functions $g\colon [0,\infty) \times E \to \mathbb{R}$ as

$$(\mathcal{L}_t^\circ g)(x) = \sum_{y \in E} q(x,y)(g(y) - g(x))\frac{\widetilde{h}(t,y)}{\widetilde{h}(t,x)}. \tag{10.3}$$

Therefore, comparing with (10.2), we see that X° is a time-inhomogenuous Markov process X° with time dependent generator matrix $\mathcal{Q}_t^\circ = (q_t^\circ(x,y),\, x,y \in E)$, where

$$q_t^\circ(x,y) = q(x,y)\frac{\widetilde{h}(t,y)}{\widetilde{h}(t,x)} \quad \text{if} \quad y \neq x$$

and $q_t^\circ(x,x) = 1 - \sum_{y \neq x} q_t^\circ(x,y)$.

Now assume that $\widetilde{X}$ is a continuous time Markov process on E with tractable $\widetilde{h}$ and $\widetilde{\mathcal{A}}\widetilde{h} = 0$, where $\widetilde{\mathcal{A}}$ is the generator of the space-time process $(t, \widetilde{X}_t)$. If $\widetilde{\mathcal{Q}} = (\widetilde{q}(x,y),\, x,y \in E)$ denotes the Q-matrix of $\widetilde{X}$, then the key term in the likelihood ratio in equation (9.4) from Theorem 9.2 can be expressed as

$$\int_0^T \frac{(\mathcal{L} - \widetilde{\mathcal{L}})\widetilde{h}}{\widetilde{h}}(t, X_t^\circ)\,\mathrm{d}t = \int_0^T \frac{\sum_{y \in E}^R (q(X_t^\circ, y) - \widetilde{q}(X_t^\circ, y))\widetilde{h}(t,y)}{\widetilde{h}(t, X_t^\circ)}\,\mathrm{d}t.$$

For finite-state Markov chains transition densities can be computed via $(p_t(x,y),\, x,y \in E) = e^{q\mathcal{Q}}$ and this would immediately give h, implying no need for constructing a guided process via an approximation to Doob's h-transform. While strictly speaking this is true, in case $|E|$ is large, numerically approximating the matrix exponential can be computationally demanding. In that case, by simplifying the dynamics of the Markov process, one may choose an approximating continuous time Markov process $\widetilde{X}$ (on E) where $\widetilde{\mathcal{Q}}$ is block diagonal, as this simplifies evaluation of matrix exponentials.

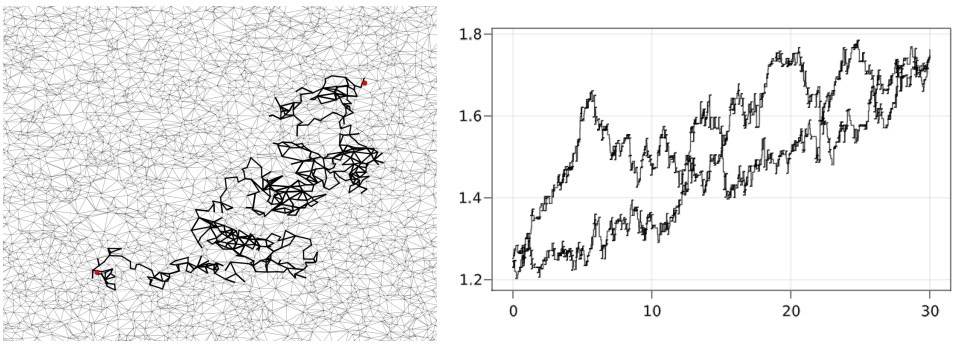


FIGURE 8: Poisson-Delaunay random walk bridge between $u \approx (1.25, 1.25)$ to $v \approx (1.75, 1.75)$ and time span $T = 30$ simulated using the $\widetilde{h}$-transform with Brownian motion scaling limit with estimated variance 0.13 in a Poisson-Delaunay grid with intensity 5000. Right: Piecewise constant coordinate processes versus time.

Remark 10.2. Note that the process in (10.1) corresponds to a Q-matrix with $q(x, x + \xi_\ell) = \lambda_\ell(x)$.

Example 10.3. Consider a Poisson process ξ on the plane $\mathbb{R}^2$ with unit intensity and the corresponding Delaunay triangulation $D(\xi)$. We consider the simple nearest neighbour random walk X^ξ on $D(\xi)$ with exponential jump times, defined by the generator.

$$\mathcal{L}^\xi f(x) := \sum_{y \in N_\xi(x)} (f(y) - f(x)), \quad x \in \xi$$

with nearest neighbours of $x \in \xi$ in $D(\xi)$ denoted by $N_\xi(x)$.

By Rousselle (2015), Theorem 1.1, if we define the time-changed and scaled process

$$(X^\varepsilon_t)_{t \geq 0} := (\varepsilon X_{\varepsilon^{-2} t})_{t \geq 0}$$

then this process has as limit as ϵ tends to 0 a scaled Brownian motion, where the scaling of the Brownian motion is independent of the realisation ξ (but depends for example on the intensity of the Poisson process).

We consider the task of sampling the process X, conditioned to start at a point $x \in \xi$ and to end in $y \in \xi$ after a fixed number of n steps. We call the resulting processing a Poisson-Delaunay bridge.

Note that the scaling limit implies that the conditional distribution of X_T given $X_t = x$ for large $T - t$ is well approximated by a Normal distribution $N(x, \sigma^2(T-t)I)$. We estimate σ^2 as scaled variance of $X_T/\sqrt{T}$ for large T and obtain an importance sampler for the Poisson-Delaunay bridge using the transition density of the Brownian $\sigma^2 W_t$ as guiding function

$$\widetilde{h}(t, x) = \frac{1}{2\pi\sigma^2(T-t)} \mathrm{e}^{-\frac{1}{2}\frac{\|v-x\|^2}{\sigma^2(T-t)}}, \quad v \in \xi$$

in (10.3).

The likelihood ratio in equation (9.4) from Theorem 9.2 can be expressed as

$$\mathbf{1}(X^\circ_T = v) 2\pi T \sigma^2 \frac{\widetilde{h}(0, x_0)}{h(0, x_0)} \exp\left(\int_0^T \left[\sum_{y \in N_\xi(X^\circ_s)} \left(\frac{\widetilde{h}(t, y)}{\widetilde{h}(s, X^\circ_s)} - 1 \right) - \frac{1}{2} \frac{\|v - X^\circ_s\|^2}{\sigma^2 (T-s)^2} \right] \mathrm{d}s \right)$$

and is bounded on trajectories with $X^\circ_t = v \in \xi$ for $t > T - \varepsilon$, $\varepsilon > 0$.

11 Probabilistic representation of Doob's h-transform using the adjoint process

In the discrete time setting, Doob's h-transform satisfies equation (3.3) and (3.2). In the continuous time setting, it satisfies Kolmogorov's backward equation (9.2). Computing h from the leaves back towards the root node is a backward filtering step. In this section we show, using a Feynman-Kac approach, that backward filtering has a probabilistic interpretation using the adjoint process of the Markov process X on $\mathcal{G}$.

Our work is related to that of Milstein et al. (2007), who consider the specific setting of Markov chains and derive Monte-Carlo estimators for the transition densities of the process using forward and reverse representations of the Markov chain. In appendix E we shed some more light on the relations between their work and ours.

One application of the adjoint process consists of approximating the h-transform using particle filtering, where h is obtained as the expectation of a weighted set of particles. Pitt et al. (2012)

11.1 Discrete time case

In this section we define the adjoint process of a Markov process on a directed tree.

Definition 11.1. Define the adjoint Markov process to $(X_t,\, t \in \mathcal{T})$ on $\mathcal{G}$ as the process $(X_t^\dagger,\, t \in \mathcal{T})$ that starts from the leaves with $X_{\mathcal{V}}^\dagger = x_{\mathcal{V}}$ and has transition densities

$$p_{t\leftarrow}^\dagger(x_{\mathrm{ch}(t)}; x_t) = \frac{\prod_{u\in\mathrm{ch}(t)} p_{t\to u}(x_t; x_u)}{\delta_t(x_{\mathrm{ch}(t)})} \tag{11.1}$$

where the normalising constant is given by

$$\delta_t(x_{\mathrm{ch}(t)}) = \int \prod_{u\in\mathrm{ch}(t)} p_{t\to u}(x_t; x_u)\lambda(\mathrm{d}x_t).$$

We obtain the following Feyman-Kac type result (the relation to continuous-time Feynman-Kac formula is detailed in subsection 11.2, but see also Del Moral et al. (2009) for discrete Feynman-Kac processes on trees.)

Theorem 11.2. For $s \in \mathcal{S}_0$ we have

$$h_s(x_s) = \int \varpi_s\, p_{s\leftarrow}^\dagger(x_{\mathrm{ch}(s)}; x_s) \prod_{\{t\in\mathcal{S}_0\setminus\{s\}\colon\, s\rightsquigarrow t\}} p_{t\leftarrow}^\dagger(x_{\mathrm{ch}(t)}; x_t)\lambda(\mathrm{d}x_t).$$

where

$$\varpi_s^\dagger = \prod_{\{t\in\mathcal{S}_0\colon\, s\rightsquigarrow t\}} \delta_t(X_{\mathrm{ch}(t)}^\dagger).$$

In particular, if $\varpi_s^\dagger$ does not depend on $X^\dagger$, then the backward filtering density $p_s^\dagger$ (i.e. the law of $X_s^\dagger$) satisfies $h_s(x) = \varpi_s^\dagger \cdot p_s^\dagger(x)$.

Here $X_t^\dagger$ can be understood as particle with weight $\varpi_t^\dagger$. Before giving the proof, note that ϖ_s in fact depends on $\{X_t^\dagger,\, t \neq s\colon s \rightsquigarrow t\}$, but this is not explicitly included in the notation. That is, at vertex s it depends on $X_t^\dagger$, where t ranges over all its children, grand-children..., i.e. its descendants.

Proof. First note for $s \in \mathcal{S}_0$ the recursion

$$\begin{aligned} h_s(x_s) &= \left(\prod_{v \in \mathrm{ch}(s) \cap \mathcal{V}} p_{s \to v}(x_s; x_v) \right) \prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} \int h_t(x_t) p_{s \to t}(x_s; x_t) \lambda(\mathrm{d}x_t) \\ &= \int \left(\prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} h_t(x_t) \right) \left(\prod_{t \in \mathrm{ch}(s)} p_{s \to t}(x_s; x_t) \right) \prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} \lambda(\mathrm{d}x_t) \end{aligned}$$

which implies with definition 11.1

$$\begin{aligned} h_s(x_s) &= \int p^\dagger_{s \leftarrow}(x_{\mathrm{ch}(s)}; x_s) \delta_s(x_{\mathrm{ch}(s)}) \prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} \omega^\dagger_t \, p^\dagger_{t \leftarrow}(x_{\mathrm{ch}(t)}; x_t) \lambda(\mathrm{d}x_t) \\ &= \int \varpi^\dagger_s \, p^\dagger_{s \leftarrow}(x_{\mathrm{ch}(s)}; x_s) \prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} p^\dagger_{t \leftarrow}(x_{\mathrm{ch}(t)}; x_t) \lambda(\mathrm{d}x_t) \end{aligned}$$

where we used that

$$\varpi^\dagger_s = \delta_s(x_{\mathrm{ch}(s)}) \prod_{t \in \mathrm{ch}(s) \cap \mathcal{S}} \varpi^\dagger_t$$

at the second equality. □

Note that $\varpi^\dagger_t$, $\varpi^\dagger_s$ are independent for s, t with $s \not\rightsquigarrow t$ and $t \not\rightsquigarrow s$.

Corollary 11.3 (Independent increments)**.** Assume $\mathcal{T}$ is a line graph and $p_{\to t}(x; y) = \psi_t(y - x)$. Then $\varpi_t = 1$.

Proof. On a line graph we have

$$p^\dagger_{t \leftarrow}(x; y) = \frac{p_{\to t}(x; y)}{\int p_{\to t}(x; y) \, \mathrm{d}x} = \frac{\psi_t(y - x)}{\varpi^\dagger_t},$$

with $\varpi^\dagger_t = \int \psi_t(y - x) \, \mathrm{d}x$. Here we use the independent increments property at the second equality. Now $\int \psi_t(y - x) \, \mathrm{d}x = 1 \ \forall y$ since $\int \psi_t(\xi - \eta) \, \mathrm{d}\xi = 1 \ \forall \eta$ by substitution. □

Example 11.4 (Independent Gamma increments)**.** Here, we get back to the example of section 7.4 and derive the $\widetilde{h}$-transform via identifying the adjoint process. As $\widetilde{X}$ has independent increments, the adjoint process $\widetilde{X}^\dagger$ for the $\widetilde{h}$-transform is the monotone decreasing process with $\widetilde{X}^\dagger_v = x_v$, $v \in \mathcal{V}$ and Gamma distributed decrements, for $(s, t) \in \mathcal{E}$,

$$\widetilde{X}^\dagger_s - \widetilde{X}^\dagger_t \mid \widetilde{X}^\dagger_t \sim \mathrm{Gamma}(\alpha_t, \beta).$$

The backward filtering distribution, the marginal distribution of $\widetilde{X}^\dagger$ has a closed form, given recursively

$$\widetilde{X}^\dagger_{\mathrm{pa}(v)} \sim x_v - \mathrm{Gamma}(\alpha_v, \beta), \quad v \in \mathcal{V}.$$

$$\widetilde{X}^\dagger_t - \widetilde{X}^\dagger_s \sim \mathrm{Gamma}(\alpha_t, \beta), \quad t \in \mathcal{T}.$$

As $\widetilde{X}$ has independent increments, by Corollary 11.3, $\widetilde{\varpi}_v \equiv 1$.

11.2 Continuous time case

The Kolmogorov backward equation (9.2) is an *analytical* description of the evolution of h as defined in (9.1) backwards in time. In this section we derive a probabilistic counterpart such that h is the (weighted) forward marginal distribution of a backward running process $X^\dagger$. In general, the forward evolution of the marginal distribution of a Markov process is described by Fokker-Planck equation. The natural object of the backward filtering pass – the adjoint infinitesimal operator – becomes Markov only after a normalisation.

Let $\mathcal{L}^*$ be the Hermitian adjoint of the operator $\mathcal{L}$ with respect to the L_2 inner product, which defines an operator on smooth Lebesgue densities. Given that X_S has initial density p_S, the marginal density $p(t,x)$ is differentiable with respect to t and in the domain of $\mathcal{L}^*$ for fix t and solves the Fokker-Planck type equation $\frac{\partial}{\partial t}p(t,x) = \mathcal{L}^* p(t,x)$.

What we are ideally looking for is a process (Y_t) with generator $\mathcal{L}_Y$ for which $\mathcal{L} = \mathcal{L}_Y^*$. Thus $\mathcal{L}_Y^*$ simultaneously describes the forward evolution of marginal densities of Y and the backward evolution of h. This is possible to limited extend: The construction of subsection 11.1 suggests to decompose

$$\mathcal{L}^* f = \mathcal{L}_Y f + cf$$

such that $\mathcal{L}_Y$ is the infinitesimal operator of a *Markov* process (Y_t), $t \in [0, T-S]$. This requires $\mathcal{L}_Y \mathbf{1} = 0$. $X^\dagger_{T-t} = Y_t$ with generator $\mathcal{L}^\dagger = \mathcal{L}_Y$, started in $X^\dagger_T \sim p^\dagger_T$ is then the continuous counterpart to the adjoint process from subsection 11.1 and the backward marginal distribution of $X^\dagger$ determines h together the corresponding weight process

$$\varpi(s,t) = \exp\left(\int_s^t c(X^\dagger_\tau)\,\mathrm{d}\tau\right).$$

Together they allow to obtain the h-transform via backward filtering densities using the following duality result.

Assumption 11.5. Assume that for bounded and continuous $f\colon [S,T]\times E \to \mathbb{R}$

$$\lim_{t\to T}\int f(t,z)p(t,z;T,v)\mathrm{d}z = f(T,v). \tag{11.2}$$

Theorem 11.6. Assume that h is given with $\int h(x)\,\mathrm{d}x =: \varpi_T$. Define the h-transform $h(s,x) = \mathbb{E}_{s,x}h(X_T)$. Let (Y_t) be an independent time-homogeneous E-valued Markov process defined on $[S,T]$ with full (extended) generator $\mathcal{L}^\dagger$ with domain $\mathcal{D}(\mathcal{L}^\dagger)$.

Assume that Y solves the martingale problem for $\mathcal{L}^\dagger$ and starting distribution h/ϖ_T and that

$$\mathcal{A}^\dagger_{t,y}p(s,x;t,y) = -c(y)p(s,x;t;y)$$

for $\mathcal{A}^\dagger = \mathcal{L}^\dagger - \frac{\partial}{\partial t}$.

Let $X^\dagger_{T-t} = Y_t$ with marginal density $p^\dagger_{T-t}$ (by definition of Y, $p^\dagger_T = h/\varpi_T$) Define

$$\varpi_s(X^\dagger_{[s,T]}) = \exp\left(\int_s^T c(X^\dagger_u)\,\mathrm{d}u\right)\varpi_T.$$

If the measure

$$\mu_s(A) = \mathbb{E}[\mathbf{1}_A(X^\dagger_s)\varpi_s(X^\dagger_{[s,T]}))]$$

has a density f_s, and $1/h(x)$ is continuous, integrable and bounded away from zero in a neighbourhood of x, and provided that $\int c(X^\dagger)\,\mathrm{d}s < \infty$ almost surely, then

$$h(s,x) = f_s(x).$$

Proof. As $\mathcal{A}_{r,x}p(s+r,x;t;y) = 0 = (\mathcal{L}^\dagger_{r,y} + \frac{\partial}{\partial r})p(t,x;T-r,y) + c(y)p(t,x;T-r;y)$ (operators act on subscripted variables)

$$p(s+r, X_{s+r}, t, y) \quad \text{and} \quad p(t,x;T-r,Y_r) + \int_0^r c(Y_u)p(t,x;T-u,Y_u)\,\mathrm{d}u$$

are martingales. Thus, from Corollary 4.13., p. 195, in Ethier and Kurtz (1986), substituting $t = s+r$, $X^\dagger_{T-r} = Y_r$,

$$\mathbb{E}_{s,x}\left[p(t,X_t;T,X^\dagger_T)\right] = \mathbb{E}\left[p(s,x;{}_{T-(t-s)},X^\dagger_{T-(t-s)})\exp\left(\int_{T-(t-s)}^T c(X^\dagger(u))\,\mathrm{d}u\right)\right].$$

We obtain using (11.2)

$$\lim_{t\to T}\mathbb{E}_{s,x}\left[p(t,X_t;T,X^\dagger_T)\right] = \lim_{t\to T}\mathbb{E}_{s,x}\int[p(t,X_t;T,x)h(x)/\varpi_T\,\mathrm{d}x] = \mathbb{E}h(X_T)/\varpi_T$$

By time-homogeneity, (11.2) also implies $\lim_{t\to T}\int f({}_{T-(t-s)},z)p(s,u;{}_{T-(t-s)},z)\mathrm{d}z = f(s,u)$ and therefore

$$\lim_{t\to T}\mathbb{E}\left[p(s,x;{}_{T-(t-s)},X^\dagger_{T-(t-s)})\varpi_{T-(t-s)}(X^\dagger_{[T-(t-s),T]})/\varpi_T\right] = f(x)/\varpi_T.$$

For conditions for property (11.2), see Lemma 7 of Schauer et al. (2017) and assumptions.

□

Remark 11.7 (Backward particle filter)**.** A numerical application of the preceding corollary is the use of a particle filter to backward filter the process X_t. Simulate weighted particles $(X^{\dagger,(i)}_t, \varpi_t)$, $i \in \{1,\dots,n\}$ backwards from time T with boundary condition $X^{\dagger,(i)}_T \sim p^\dagger_T = h/\varpi_T$ and to use a kernel estimate

$$h(s,x) \approx \frac{1}{n}\sum_{i=1}^n \kappa_{x,\delta}(X^{\dagger,(i)}_s)\varpi_s.$$

using a compactly supported kernel function with $\int\kappa_{x,\delta}(y)\,\mathrm{d}y = 1$ concentrated at x, say with $\mathbf{1}_{[x-\delta/3,x+\delta/3]} \le \delta\kappa_{x,\delta} \le \mathbf{1}_{[x-\delta,x+\delta]}$.

Example 11.8. Consider a diffusion process with transition densities $p(s,x;t;y)$ and generator extending

$$\mathcal{L}f(x) = \sum_{i=1}^d b(x)\frac{\partial}{\partial x_i}f(x) + \frac{1}{2}\sum_{i,j=1}^d a_{ij}(x)\frac{\partial^2}{\partial x_i\,\partial x_j}f(x).$$

Then the Hermitian adjoint is

$$\mathcal{L}^*f(x) = -\sum_{i=1}^d\frac{\partial}{\partial x_i}\left(b_i(x)f(x)\right) + \frac{1}{2}\sum_{i,j=1}^d\frac{\partial^2}{\partial x_i\,\partial x_j}\left(a_{ij}(x)f(x)\right).$$

Let

$$c(x) = \mathcal{L}^*\mathbf{1}(x) = -\sum_{i=1}^d\frac{\partial b_i(x)}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^d\frac{\partial^2 a_{ij}(x)}{\partial x_i\partial x_j}.$$

and define

$$\mathcal{L}^\dagger f = \mathcal{L}^*f - cf,$$

implying

$$\mathcal{L}^\dagger f(x) = \sum_{i=1}^{d} b_i^\dagger(x) \frac{\partial}{\partial x_i} f(x) + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij}(x) \frac{\partial^2}{\partial x_i \, \partial x_j} f(x),$$

where

$$b_i^\dagger(x) = -b_i(x) + \sum_j \partial_j a_{ij}(x).$$

Compare also Milstein et al. (2004). Under regularity conditions on $b^\dagger$, there is a diffusion process Y which solves the associated martingale problem for initial law h/ϖ_T. Finally, by Kolmogorov's forward equation,

$$\mathcal{A}_{t,y}^\dagger p(s,x;t,y) = -c(y)p(s,x;t;y).$$

Note that for a linear process with $b(t,x) = B_t x + \beta_t$ and $\sigma(t,x) = \sigma_t$,

$$c(t,x) = \operatorname{trace}(B_t)$$

and ϖ is a deterministic function of s and t. In fact, by Liouville's formula,

$$\varpi(s,t) = |\Phi(s,t)|.$$

Remark 11.9. Compare with Milstein et al. (2004), who derive for diffusion processes the representation

$$\int g(x)p(s,x;t,y)\,\mathrm{d}x = \mathbb{E}\left[g(X_s^\dagger)\varpi(s,t) \mid X_t^\dagger = y\right]$$

of the functionals of transition densities using the backward process $X^\dagger$.

Remark 11.10. An alternative approach to the adjoint process is via time reversion of a stochastic differential equations. Let X be a diffusion process with

$$\mathrm{d}X_t = b(t,X_t)\,\mathrm{d}t + \sigma(t,X_t)\,\mathrm{d}W_t. \tag{11.3}$$

This corresponds to the Stratonovich SDE

$$\mathrm{d}X_t = c(t,X_t)\,\mathrm{d}t + \sigma(t,X_t)\partial W_t$$

with

$$c^i = b^i - \frac{1}{2}\sum_q U_q(\sigma_q^i), \qquad U_q = \sigma_q^i(x)\mathrm{D}_i.$$

Let $Y_t \equiv X_{T-t}$ denote the time reversed process; it satisfies the Stratonovich SDE

$$\mathrm{d}Y_t = -c(Y_t)\,\mathrm{d}t - \sigma_q^i(Y_t)\partial\bar{W}_t.$$

This in turn corresponds to the Ito-SDE

$$\mathrm{d}Y_t = \left(-c(t,Y_t) + \frac{1}{2}\sum_q U_q(\sigma_q^i)\right)(t,Y_t)\,\mathrm{d}t - \sigma_q^i(t,Y_t)\,\mathrm{d}\bar{W}_t.$$

and hence

$$\mathrm{d}Y_t = \left(-b(t,Y_t) + \sum_q U_q(\sigma_q^i)\right)(t,Y_t)\,\mathrm{d}t - \sigma_q^i(t,Y_t)\,\mathrm{d}\bar{W}_t.$$

Example 11.11 (Two-sided bridge)**.** Take independent X and $X^\dagger$ on the same probability space. We are interested in the h-transform $X^\star$ for a given h. Consider the joint process

$$\mathbb{X}_t = (X_t, X^\dagger_{T-t}), \quad X_0 = u, \quad X^\dagger_T \sim h/\varpi_T.$$

Then the h-transform

$$h(s,(x,y)) = \frac{p(s,x;T-s,y)h(T-s,y)}{h(0,u)p^\dagger(T-s,y \mid T,v)h(v)/\varpi_T}$$

defined on $[0,T] \times E^2$ changes $\mathbb{X}$ into $\mathbb{X}^\star$ and reflect$(\mathbb{X})$,

$$\text{reflect}(\mathbb{X})_t = \begin{cases} X_t, & t \le T/2 \\ X^\dagger_t, & t > T/2 \end{cases}$$

into $X^\star = \text{reflect}(\mathbb{X}^\star)$ on $[0, T/2-\varepsilon) \cup (T/2+\varepsilon, T]$, as it is the density of the joint law of $X^\star_t$, $X^\star_{T-t}$ with respect to the independent product of the laws of X_t and $X^\dagger_{T-t}$ as reference measure. If ϖ_{T-s} is deterministic, this simplifies to

$$h(s,(x,y)) = \frac{p(s,x;T-s,y)}{h(0,u)\varpi_{T-s}}.$$

For example, let $\mathbb{X}^\star$ be the two-dimensional solution to

$$\mathrm{d}\mathbb{X}^\star_t = \frac{1}{T-2t}\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \mathbb{X}^\star_t \,\mathrm{d}t + \mathrm{d}W_t, \quad \mathbb{X}^\star_0 = (u,v)', \quad t \in [0, T/2),$$

where W is a two-dimensional Brownian motion. Then $(B_t)_{t\in[0,T]}$ with $B = \text{reflect}(\mathbb{X}^\star)$ is a Brownian bridge from u to v.

Typically, h as defined above is intractable. Then one chooses a guided process derived from a process $(\widetilde{X}, \widetilde{X}^\dagger)$ for which the corresponding transform $\widetilde{h}$ is a tractable space-time harmonic function, as in the example of the Brownian motion. See also Schauer (2015), chapter 6.5 and accompanying propositions.

12 A numerical example

Here we give an example that fits within the setup of section 7.2. The code is available online[8]. We consider a population of n individuals, where each individual is either **S**usceptible, **I**nfected or **R**ecovered. We assume that each individual has a fixed (non-time varying) set of neighbours and that each individual transitions from state **S** to **I** proportional to the number infected neighbours. Furthermore, each individual transitions from state **I** to **R** and **R** back to **S** at constant (not necessarily equal) rates. Hence, we can view this as a chemical reaction network with three "reactions": Let x be the vector of states at time t and write x_i for its i-th component.

- If $x_i = \mathbf{S}$, then it transitions to **I** with intensity $\lambda N_i(x)$, where $N_i(x)$ is the number of infected neighbours of individual i at time t.
- If $x_i = \mathbf{I}$, then it transitions to **R** with intensity μ.
- If $x_i = \mathbf{R}$, it transitions to **S** with intensity ν.

[8] https://github.com/fmeulen/GuidedDAGExamples.jl

In particular, contrary to the traditional SIR-model, individuals can become susceptible again after being recovered.

We consider a time-discretised version of the problem, where time steps are multiples of $\tau > 0$. In each time interval of length τ, conditional on the the present state of all individuals, each individual independently remains in its present state or transitions. Hence, individuals are like particles in the setup of section 7.2. The transition matrix for individual i is given by

$$K_i(x) = \begin{bmatrix} \psi(\lambda N_i(x)) & 1-\psi(\lambda N_i(x)) & 0 \\ 0 & \psi(\mu) & 1-\psi(\mu) \\ 1-\psi(\nu) & 0 & \psi(\nu) \end{bmatrix},$$

where

$$N_i(x) = \{\text{number of infected neighbours of individual } i \text{ in state } x\}$$

and $\psi(u) = \exp(-\tau u)$ (as τ is fixed we drop it in the notation). Note that at each time there is a transition matrix $K_i(x)$, but that this has been suppressed in the notation.

Assume an integer $T > 2$ is given and the state of each individual is recorded at times jT, where $j = 0, 1, \ldots, N$. We aim to reconstruct the state of individuals at non-observation times and estimate the parameter vector $\theta := (\lambda, \mu, \nu)$.

A simple choice that induces $\widetilde{h}$ is to take

$$\widetilde{K}_i = \begin{bmatrix} \psi(\widetilde{\lambda}) & 1-\psi(\widetilde{\lambda}) & 0 \\ 0 & \psi(\mu) & 1-\psi(\mu) \\ 1-\psi(\nu) & 0 & \psi(\nu) \end{bmatrix}. \tag{12.1}$$

Note however that a problem can occur when an individual is in state **S** while having no infected neighbours and being conditioned to be in another state at time T. In that case the term $p_{i(T-1)\to i(T)}(x_{i(T-1)}; x_{i(T)})$ appearing in the likelihood will be zero. This does not invalidate the approach but it may induce simulation of guided processes that will be rejected for sure. To deal with this problem, we propose to modify the dynamics of the model slightly by redefining $\psi(\lambda N_i(x))$ by $(1-\delta)\psi(\lambda N_i(x))$ and $\psi(\widetilde{\lambda})$ by $(1-\delta)\psi(\widetilde{\lambda})$ with δ a small positive number. This implies that there is a small probability for an individual to get infected, even if all its neighbours are not infected.

In Figure 9 we show on the bottom-right panel a forward simulation from the model with 100 individuals using time-discretisation step $\tau = 0.1$. At each location, the neighbourhood of a person is defined by its two neighbours both on the left and right (at the boundaries persons have less neighbours as there are no neighbours on either the left or right). The individuals are located horizontally on a grid and the vertical axis denotes time. At time zero, the seven leftmost individuals are infected. From the forward simulation we extracted 20 observations, where in between discrete time observations there are 48 latent states. The following parameters were taken: $\delta = 0.001$, $\lambda = 2.5$, $\mu = 0.6$, $\nu = 0.1$. We ran an MCMC-algorithm that iteratively updates the latent states and the parameter $\theta = (\lambda, \mu, \nu)$, assuming δ to be known. For updating θ we used a random-walk Metropolis-Hastings step on $\log\lambda$, $\log\mu$ and $\log\nu$. For updating paths we used a preconditioned Crank-Nicolsen scheme by turning path-simulation into a normalising flow.

The computational efficiency of the resulting algorithm strongly depends on the choice of the approximate h-transform. With the simple choice of (12.1) this translates to choosing $\widetilde{\lambda}$ which is non-trivial. Moreover, taking the matrix in (12.1) independent of i and the same for all backward kernels is too simple to yield satisfactory results in case the number of missing data is relatively large (as for our simulated data). For that reason, we propose to take $\widetilde{K}_i$ as in equation (12.1),

but with elements $[1,1]$ and $[1,2]$ replaced with c_i and $1-c_i$ respectively, where c_i is initialised by

$$(1-\delta)\psi(n_i).$$

with n_i denoting the number of infected neighbours of individual i. Next, at each iteration c_i is recomputed and updated to a convex combination of the newly computed values with the values from the previous iteration. After a pre specified number of iterations, the matrices $\widetilde{K}_i$ are kept fixed. Again, while dependence on time is suppressed in the notation, note that there is a kernel $\widetilde{K}_i$ at each time, and henceforth also number c_i and n_i for each time.

We ran the MCMC algorithm for $10,000$ iterations; adapting the approximate h-transform for the first third of those. In Figure 9 we show visualisations of the observed data, the initially simulated guided process X°, the reconstructed state at an iteration halfway the number of iterations, the final iterate, as also the true forward simulated paths from which the data were extracted. Whereas the observed data do not clearly reveal a pattern of infections, the final iterate does so. Also note the visual improvements obtained over the initial iterate.

In Figure 10 we show corresponding trace plots for parameters. 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. If the latter is underestimated, a satisfactory reconstruction of the smoothing distribution can be obtained by overestimation of λ.

Acknowledgement: Frank Schäfer implemented Mitosis rules for stochastic differential equation based on earlier code by Marcin Mider. We thank Richard Kraaij for discussions on the topic of h-transforms in the early stage of this work. We thank for valuable input Marc Corstanje, Keno Fischer, Philipp Gabler, Evan Patterson and Chad Scherrer.

References

Ambrogioni, L., Lin, K., Fertig, E., Vikram, S., Hinne, M., Moore, D. and van Gerven, M. (2021). Automatic structured variational inference.

Anderson, D. and Kurtz, T. (2015). *Stochastic Analysis of Biochemical Systems*. Mathematical Biosciences Institute Lecture Series. Springer International Publishing.

Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *Ann. Statist.* **37**(2), 697–725.

Bardel, N. and Desbouvries, F. (2012). Exact Bayesian prediction in a class of Markov-switching models. *Methodol. Comput. Appl. Probab.* **14**(1), 125–134.

Belomestny, D., Gugushvili, S., Schauer, M. and Spreij, P. (2020). Nonparametric Bayesian volatility estimation for Gamma-driven stochastic differential equations.

Bierkens, J., van der Meulen, F. and Schauer, M. (2020). Simulation of elliptic and hypo-elliptic conditional diffusions. *Advances in Applied Probability* **52**(1), 173–212.

Briers, M., Doucet, A. and Maskell, S. (2010). Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics* **62**(1), 61.

Carlson, F. B. (2020). MonteCarloMeasurements.jl: Nonlinear propagation of arbitrary multivariate distributions by means of method overloading. arXiv: 2001.07625.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika* **81**(3), 541–553.

Chen, R. T., Rubanova, Y., Bettencourt, J. and Duvenaud, D. K. (2018). Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583.

Chou, K. C., Willsky, A. S. and Benveniste, A. (1994). Multiscale recursive estimation, data fusion, and regularization. *IEEE Trans. Automat. Control* **39**(3), 464–478.

Del Moral, P., Patras, F. and Rubenthaler, S. (2009). Tree based functional expansions for Feynman-Kac particle models. *Ann. Appl. Probab.* **19**(2), 778–825.

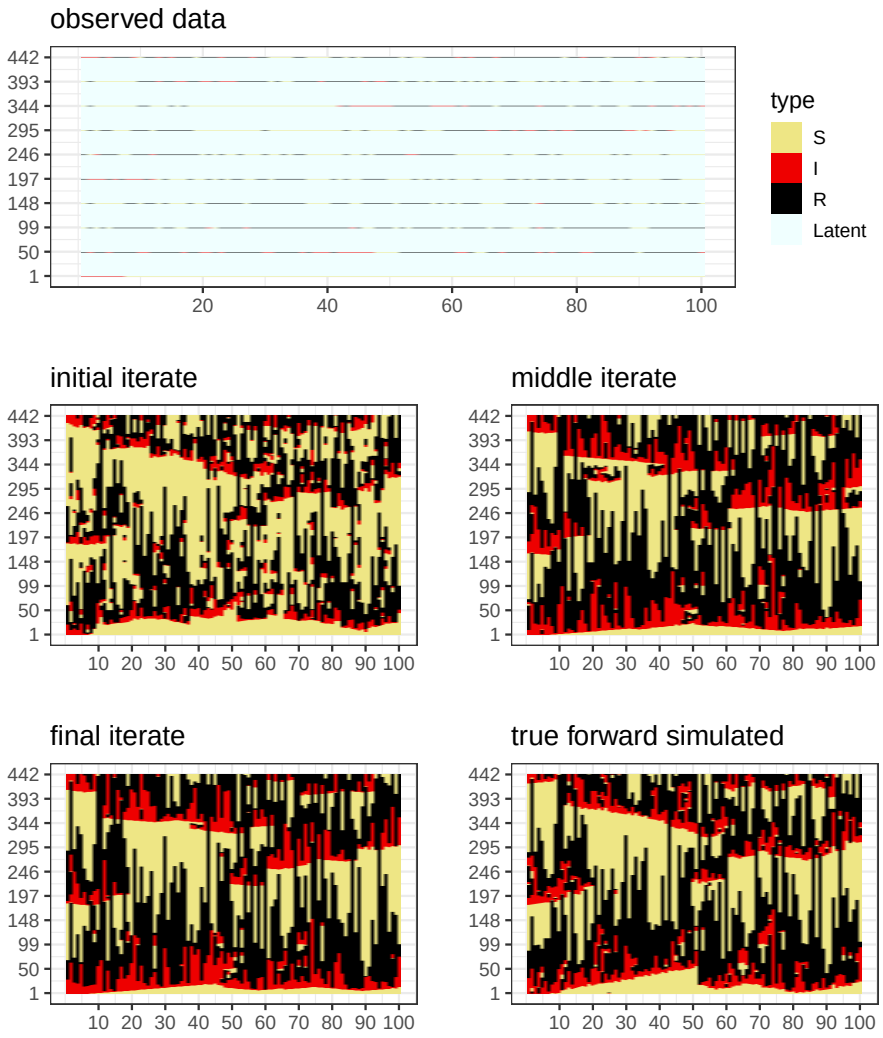


FIGURE 9: Top row: visualisation of the observed data. Middle row and bottom-left figures: 3 iterates of the reconstructed configuration. Bottom-right; true forward simulated configuration. In each panel: the horizontal axis shows the 100 individuals, while the vertical axis displays time.

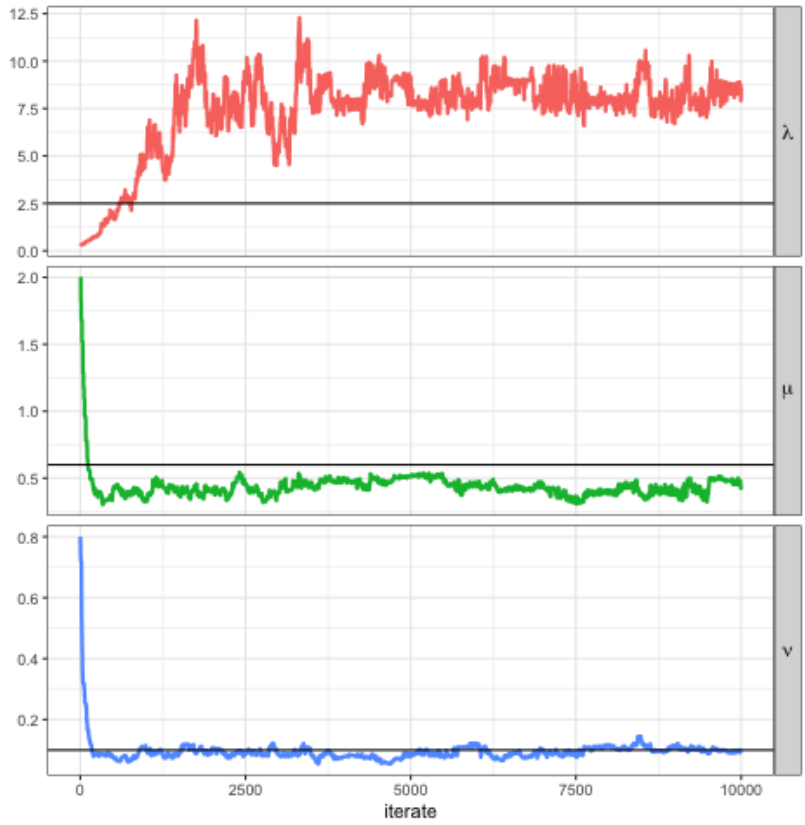


FIGURE 10: Trace plots for parameters. The values used in the forward simulated model (from which the observations were extracted) are denoted by black horizontal lines.

Delyon, B. and Hu, Y. (2006). Simulation of conditioned diffusion and application to parameter estimation. *Stochastic Processes and their Applications* **116**(11), 1660 – 1675.

Desbouvries, F., Lecomte, J. and Pieczynski, W. (2006). Kalman filtering in pairwise Markov trees. *Signal Processing* **86**, 1049–1054.

Doucet, A. and Lee, A. (2018). Sequential Monte Carlo methods. *Handbook of Graphical Models* pp. 165–189.

Ethier, S. N. and Kurtz, T. G., eds. (1986). *Markov Processes*. John Wiley & Sons, Inc.

Fearnhead, P., Giagos, V. and Sherlock, C. (2014). Inference for reaction networks using the linear noise approximation. *Biometrics* **70**(2), 457–466.

Fischer, K. (2020). Compiler3 project. https://github.com/keno/Compiler3.jl.

Fritz, T. (2020). A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics* **370**, 107239.

Glynn, P. W. and Iglehart, D. L. (1989). Importance sampling for stochastic simulations. *Management Science* **35**(11), 1367–1392.

Holm, D. D. (2015). Variational principles for stochastic fluid dynamics. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **471**(2176), 20140963.

Innes, M. (2018). Don't unroll adjoint: Differentiating SSA-form programs. arXiv:1810.07951.

Jacobs, B. (2019). *Structured Probabilistic Reasoning*. Forthcoming, available at http://www.cs.ru.nl/B.Jacobs/PAPERS/ProbabilisticReasoning.pdf.

Jacobs, B. and Zanasi, F. (2020). *The logical essentials of Bayesian reasoning*, p. 295–332. Cambridge University Press.

Jordan, M. I. (2004). Graphical models. *Statist. Sci.* **19**(1), 140–155.

Ju, N., Heng, J. and Jacob, P. E. (2021). Sequential Monte Carlo algorithms for agent-based models of disease transmission.

Kallenberg, O. (2002). *Foundations of Modern Probability*. Probability and Its Applications. Springer New York.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. *Proceedings of the 2nd International Conference on Learning Representations (ICLR)* .

Kobyzev, I., Prince, S. and Brubaker, M. (2020). Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* pp. 1–1.

Lahoz, W., Khattatov, B. and Menard, R. (2010). *Data Assimilation: Making Sense of Observations*. Springer Berlin Heidelberg.

Law, J. and Wilkinson, D. (2019). Functional probabilistic programming for scalable Bayesian modelling. arXiv: 1908.02062.

Liggett, T. (2010). *Continuous Time Markov Processes: An Introduction*. Graduate studies in mathematics. American Mathematical Society.

Liggett, T. M. (2005). *Interacting particle systems*. Classics in Mathematics. Springer-Verlag, Berlin. Reprint of the 1985 original.

Marchand, J.-L. (2012). *Conditionnement de processes markoviens*. IRMAR, Ph.d. Thesis Université de Rennes 1.

Mider, M., Schauer, M. and van der Meulen, F. (2020). Continuous-discrete smoothing of diffusions. arXiv: 1712.03807.

Milstein, G., Schoenmakers, J. and Spokoiny, V. (2007). Forward and reverse representations for Markov chains. *Stochastic Processes and their Applications* **117**(8), 1052 – 1075.

Milstein, G. N., Schoenmakers, J. G. and Spokoiny, V. (2004). Transition density estimation for stochastic differential equations via forward-reverse representations. *Bernoulli* **10**(2), 281–312.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective (Adaptive computation and machine learning)*. Massachusetts Institute of Technology.

nLab authors (2021). HomePage. http://ncatlab.org/nlab/show/HomePage. Revision 281.

Palmowski, Z. and Rolski, T. (2002). A technique for exponential change of measure for Markov processes. *Bernoulli* **8**(6), 767–785.

Panangaden, P. (1999). The category of Markov kernels. *Electronic Notes in Theoretical Computer Science* **22**, 171 – 187. PROBMIV'98, First International Workshop on Probabilistic Methods in Verification.

Papaspiliopoulos, O., Roberts, G. O. and Sköld, M. (2003). Non-centered parameterizations for hierarchical models and data augmentation. In *Bayesian statistics, 7 (Tenerife, 2002)*, pp. 307–326. Oxford Univ. Press, New York. With a discussion by Alan E. Gelfand, Ole F. Christensen and Darren J. Wilkinson, and a reply by the authors.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann series in representation and reasoning. Elsevier Science.

Pitt, M. K., dos Santos Silva, R., Giordani, P. and Kohn, R. (2012). On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal of Econometrics* **171**(2), 134–151.

Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., Skinner, D. and Ramadhan, A. (2020). Universal differential equations for scientific machine learning. arXiv: 2001.04385.

Rackauckas, C. and Nie, Q. (2017). Adaptive methods for stochastic differential equations via natural embeddings and rejection sampling with memory. *Discrete and continuous dynamical systems. Series B* **22**(7), 2731.

Revels, J., White, L. and contributors (2018-2020). Chainrules.jl. https://github.com/JuliaDiff/ChainRules.jl.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In F. Bach and D. Blei, eds., *Proceedings of Machine Learning Research*, volume 37, pp. 1530–1538. PMLR, Lille, France.

Riley, M. (2018). Categories of optics. arXiv:1809.00738.

Rousselle, A. (2015). Quenched invariance principle for random walks on Delaunay triangulations. *Electron. J. Probab.* **20**, 32 pp.

Schauer, M. (2015). *Bayesian inference for discretely observed diffusion processes*. Technical University Delft. Verlag Blaues Schloss Marburg.

Schauer, M., van der Meulen, F. and van Zanten, H. (2017). Guided proposals for simulating multi-dimensional diffusion bridges. *Bernoulli* **23**(4A), 2917–2950.

Selinger, P. (2011). A survey of graphical languages for monoidal categories. In B. Coecke, ed., *New Structures for Physics*, pp. 289–355. Springer Berlin Heidelberg, Berlin, Heidelberg.

Suda, D. (2020). Stability of sampling proposals for reducible diffusions over large time intervals. *Communications in Statistics - Theory and Methods* **0**(0), 1–19.

van der Meulen, F. and Schauer, M. (2017). Bayesian estimation of discretely observed multi-dimensional diffusion processes using guided proposals. *Electron. J. Statist.* **11**(1), 2358–2396.

Wang, B., Yuan, B., Shi, Z. and Osher, S. J. (2018). ResNets Ensemble via the Feynman-Kac formalism to improve natural and robust accuracies. arXiv: 1811.10745.

Wilkinson, D. (2017). An introduction to category theory and functional programming for scalable statistical modelling and computation.

Yonekura, S. and Beskos, A. (2020). Online smoothing for diffusion processes observed with noise. arXiv: 2003.12247.

A Notation

Borel measurable spaces $(E, \mathfrak{B})$, $(F, \mathfrak{C})$ are often written as S and T respectively. A Markov kernel between S and T is denoted by $\kappa\colon S \twoheadrightarrow T$. The set of bounded measures on S is denoted by $\mathcal{M}(S)$; the space of bounded measurable function on S is denoted by $\mathbf{B}(S)$.

Let $\mathcal{G} = (\mathcal{T}, \mathcal{E})$ be a directed, finite graph without parallel edges and self-loops, with vertices $\mathcal{T}$ and edges $\mathcal{E}$, where $e = (s, t)$, $s, t \in \mathcal{T}$ denotes a directed edge from s to t.

- The parent set of a vertex is the set of all vertices that feed into it: $\mathrm{pa}(s) = \{t\colon e(t, s) \in \mathcal{E}\}$.
- The children set of a vertex is the set of all vertices that feed out of it: $\mathrm{ch}(s) = \{t\colon e(s, t) \in \mathcal{E}\}$.
- Define $s \rightsquigarrow t$ recursively by $s = t$ or $s \rightsquigarrow a$ with $(s, a) \in \mathcal{E}$ for some a. That is, $s = t$ or there is a directed path from s to t.
- Ancestors of a vertex are its parent, grand-parents, etc.: $\mathrm{anc}(t) = \{s \neq t\colon s \rightsquigarrow t\}$. Thus, for fixed t, $\{s\colon s \rightsquigarrow t\} = \mathrm{anc}(t) \cup \{t\}$.

- A leaf (or sink) is a vertex with no children. The set of leaves is denoted by $\mathcal{V}$.
- For each vertex t, we denote by $\mathcal{V}_t = \{v \in \mathcal{V}, t \rightsquigarrow v\}$ the leaves which are descendants of t (including t, if t is a leaf).
- A source is a vertex with no parents.

B Remaining proofs

Proof of Proposition 5.8. We first rewrite the right-hand-side. We have $\mu M = \mu_1 \otimes \mu_2$, where $\mu_1 = c_\mu^{-1/2} \int \mu(\cdot\, \mathrm{d}x')$, with $c_\mu = \int \mathrm{d}\mu$ and μ_2 defined likewise. Hence

$$\mu M(\kappa \otimes \kappa') = (\mu_1 \otimes \mu_2)(\kappa \otimes \kappa') = (\mu_1 \kappa) \otimes (\mu_2 \kappa').$$

To rewrite the left-hand-side, first define $\bar{\mu} = \mu(\kappa \otimes \kappa')$, then

$$\mu(\kappa \otimes \kappa')M = \bar{\mu}M = \bar{\mu}_1 \otimes \bar{\mu}_2$$

where $\bar{\mu}_1 = c_{\bar{\mu}}^{-1/2} \int \bar{\mu}(\cdot, \mathrm{d}x')$ (and similarly $\bar{\mu}_2$). By comparing the right-hand-sides of the previous two displayed equations we see it suffices to show that $\bar{\mu}_1 = \mu_1 \kappa$. To this end, note that

$$\mu(\kappa \otimes \kappa')(B, \mathrm{d}x') = \int_{x \in B} \int_z \int_{z'} \kappa(z, \mathrm{d}x) \kappa'(z', \mathrm{d}x') \mu(\mathrm{d}z\, \mathrm{d}z')$$

and therefore

$$\begin{aligned}
\bar{\mu}_1(B) &= c_{\bar{\mu}}^{-1/2} \int \mu(\kappa \otimes \kappa')(B, \mathrm{d}x') \\
&= c_{\bar{\mu}}^{-1/2} \int_{x \in B} \int_z \int_{z'} \kappa(z, \mathrm{d}x) \mu(\mathrm{d}z\, \mathrm{d}z') \\
&= c_{\bar{\mu}}^{-1/2} \int_{x \in B} \int_z \kappa(z, \mathrm{d}x) \mu_1(\mathrm{d}z) c_\mu^{1/2} \\
&= (c_\mu / c_{\bar{\mu}})^{1/2} (\mu_1 \kappa)(B).
\end{aligned}$$

The result now follows since $c_\mu = c_{\bar{\mu}}$. □

Proof of Proposition 5.15.

$$\begin{aligned}
&\left(M^*_{\lambda_1 \otimes \lambda_1'}(\kappa \otimes \kappa')h\right)(x, x') \\
&\quad = \frac{\int h(y, y') \kappa(x, \mathrm{d}y) \kappa'(x', \mathrm{d}y') \lambda_1'(\mathrm{d}x') \int h(y, y') \kappa(x, \mathrm{d}y) \lambda_1(\mathrm{d}x) \kappa'(x', \mathrm{d}y')}{\int h(y, y') \kappa(x, \mathrm{d}y) \kappa'(x', \mathrm{d}y') \lambda_1(\mathrm{d}x) \lambda_1'(\mathrm{d}x')} \\
&\quad = \frac{1}{\int h\, \mathrm{d}(q \otimes q')} \int h(y, y') \kappa(x, \mathrm{d}y) q'(\mathrm{d}y') \int h(y, y') q(\mathrm{d}y) \kappa'(x', \mathrm{d}y') \\
&\quad = \int \frac{\int h(y, y') \lambda_2'(\mathrm{d}y')}{\sqrt{\int h\, \mathrm{d}(\lambda_2 \otimes \lambda_2')}} \kappa(x, \mathrm{d}y) \int \frac{\int h(y, y') \lambda_2(\mathrm{d}y)}{\sqrt{\int h\, \mathrm{d}(\lambda_2 \otimes \lambda_2')}} \kappa(x', \mathrm{d}y') \\
&\quad = \left((\kappa \otimes \kappa')M^*_{\lambda_2 \otimes \lambda_2'} h\right)(x, x')
\end{aligned}$$

where $q = \int \kappa(x, \cdot) \lambda_1(\mathrm{d}x) = \lambda_2$ and $q' = \int \kappa'(x', \cdot) \lambda_1'(\mathrm{d}x') = \lambda_2'$. □

Proof of Proposition 6.1. One verifies that for each $\kappa \colon S \nrightarrow T$ and $\kappa' \colon S' \nrightarrow T'$ the following diagrams commute,

$$\begin{array}{ccc}
G(S) \times G(S') & \xrightarrow{\otimes_{S,S'}} & G(S \otimes S') \\
{\scriptstyle G(\kappa) \times G(\kappa)'}\downarrow & & \downarrow{\scriptstyle G(\kappa \otimes \kappa)'} \\
G(T) \times G(T') & \xrightarrow{\otimes_{T,T'}} & G(T \otimes T')
\end{array}$$

and

$$\begin{array}{ccc} H(T)\times H(T) & \xrightarrow{\odot_{T,T'}} & H(T\otimes T') \\ {\scriptstyle H(\kappa)\times H(\kappa')}\downarrow & & \downarrow{\scriptstyle H(\kappa\otimes\kappa')} \\ H(S)\times H(S') & \xrightarrow{\odot_{S,S'}} & H(S\otimes S') \end{array}$$

and that the morphisms $\varphi_F\colon I \nrightarrow H(\mathcal{I})$, $\varphi_G\colon I \nrightarrow G(\mathcal{I})$ between monoidal identity elements satisfies associativity and certain unitality axioms. □

Proof of Theorem 7.1. First note that

$$\begin{aligned}(\kappa h)(x) &= \int h(y)\kappa(x,\,\mathrm{d}y) = \int \varpi(c,F,H)\varphi(y;H^{-1}F,H^{-1})\varphi(y;\mu(x),Q(x))\,\mathrm{d}y \\ &= \varpi(c,F,H)\varphi(H^{-1}F;\mu(x),Q(x)+H^{-1}).\end{aligned}$$

By using the specific form of $\mu(x)$ and $Q(x)$ for the kernel $\widetilde{\kappa}$ the expression for the weight follows.

To find the parametrisation of $\widetilde{\kappa}h$, let $C = Q + H^{-1}$. We have

$$\begin{aligned}(\widetilde{\kappa}h)(x) &= \varpi(c,F,H)\varphi(H^{-1}F;\Phi x+\beta,Q+H^{-1}) \\ &= \varpi(c,F,H)(2\pi)^{-\mathrm{d}/2}|C|^{-1/2} \\ &\quad\times\exp\left(-\frac{1}{2}x'\Phi'C^{-1}\Phi x + (H^{-1}F-\beta)'C^{-1}\Phi x - \frac{1}{2}(H^{-1}F-\beta)'C^{-1}(H^{-1}F-\beta)\right) \\ &= U(c-\log\varphi^{\mathrm{can}}(0,F,H)+\log\varphi(\beta;H^{-1}F,C),\Phi'C^{-1}(H^{-1}F-\beta),\Phi'C^{-1}\Phi)(x).\end{aligned}$$

By equating this to $U(\bar{c},\bar{F},\bar{H})$ the parametrisation can be inferred directly, If Φ is invertible, then

$$\bar{F}'\bar{H}\bar{F} = (H^{-1}F-\beta)'C^{-1}(H^{-1}F-\beta).$$

A bit of algebra then gives the stated result.

The derivation of the parametrisation of the fusion step is trivial.

To derive the forward map, we write $\propto$ to denote proportionality with respect to y

$$\begin{aligned}\frac{h(y)\kappa(x,\,\mathrm{d}y)}{\mathrm{d}y} &\propto \exp\left(-\frac{1}{2}y'Hy+y'F\right)\exp\left(-\frac{1}{2}(y-\mu(x))'Q(x)^{-1}(y-\mu(x))\right) \\ &\propto \exp\left(-\frac{1}{2}y'(H+Q(x)^{-1})y + y'(F+Q(x)^{-1}\mu(x))\right) \\ &\propto \varphi^{\mathrm{can}}(y;F+Q(x)^{-1}\mu(x),H+Q(x)^{-1})\end{aligned}$$

which suffices to be shown. □

Proof of Thm. 7.4. For deriving the pullback of κ, we compute

$$\begin{aligned}(\kappa h)(x) &= \int_x^{x_v}\psi(x_v-y;A,\beta)\psi(y-x;\alpha,\beta(x))\,\mathrm{d}y \\ &= \frac{\beta^A\beta(x)^\alpha}{\Gamma(A)\Gamma(\alpha)}e^{-\beta x_v+\beta(x)x}\int_x^{x_v}(x_v-y)^{A-1}(y-x)^{\alpha-1}e^{(\beta-\beta(x))y}\,\mathrm{d}y \\ &= \frac{\beta^A\beta(x)^\alpha}{\Gamma(A)\Gamma(\alpha)}e^{-\beta(x_v-x)}(x_v-x)^{A+\alpha-1}\int_0^1 z^{\alpha-1}(1-z)^{A-1}e^{-z\xi(x)}\,\mathrm{d}z \\ &= h_{A+\alpha,\beta}(x)\frac{1}{B(\alpha,\beta)}\int_0^1 z^{\alpha-1}(1-z)^{A-1}e^{-z\xi(x)}\,\mathrm{d}z\end{aligned}$$

where we made the substitution $y = x + z(x_v - x)$ at the third equality and $\mathrm{B}(\alpha,\beta)$-denotes the Beta-function, evaluated at (α,β).

From this expression the pullback for $\widetilde{\kappa}$ follows directly, since $\xi(x) = \xi$ implies $\xi(x) = 0$. This also directly gives the expression for the weight.

If $\mu = \delta_x$, then drawing from the forward measure entails drawing from a density which is proportional to $h(y)\kappa(x, \mathrm{d}y)$ (proportionality with respect to y). The claim now follows upon inspecting the derivation of $(\kappa h)(x)$ and keeping all terms under the integral proportional to z.

□

C Implementation details

The backward transformation (5.9) is

```
function backward(tkappa, U)
    Φ, β, Q = params(tkappa)
    c, F, H = params(U)
    Σ = inv(H)
    K = Φ'\(Σ + Q)
    v = Σ*F - β
    Fp = K*v
    Hp = K*Φ
    cp = c - logphican(0; F, H) + logphi(β; H\F, Σ + Q)
    message = (U(c, F, H), U(cp, Fp, Hp))
    message, U(cp, Fp, Hp)
end
```

and the forward transformation (5.8) for $\mu = \mathrm{e}^{\ell}\delta_x$ (a weighted Dirac measure how it is needed for sampling a new value with importance weight)

```
function forward(kappa, message, (l, x))
    μ, Q = params(kappa)
    U(c, F, H), U(cp, Fp, Hp) = message
    lp = l + logphi(H\F, μ(x), Q(x) + inv(H)) +
            c - logphican(0,F,H) - logU(x, cp, Fp, Hp)
    return (lp, NCan(F + Q(x)\μ(x), H + inv(Q(x))))
end
```

D Continuation-passing style h-transform

By Remark 6.10, a right-to-left optic can be transformed into a left-to-right optic. This allows in principle to adapt rule-based code for automatic differentiation to task of computing the h-transform. A continuation style optic for the h-transform can be written in Julia inspired pseudo code as

```
function htransform(g, a)
    m, b = left(g, a)
    b, b′ -> right(g, m, b′)
end
function left(g, a)
    m = h -> (a′ -> left′(g, a′, h))
    m, g(a)
end
function right(g, m, (fs, h))
    fs ∘ m(h), right′(g, h)
end
```

E Connection to Milstein et al. (2007)

Here, we shed light on some connections between the approach in Milstein et al. (2007) and the present work. Suppose $\{X_n\}$ is a Markov chain with $X_0 = x$ and transition densities p_n (with respect to the dominating measure λ). Denote the state space of the Markov chain by S and assume $\varphi_k \colon S \times S \to \mathbb{R}$ and $u_N : S \to \mathbb{R}$ are measurable and bounded functions. For given u_N, define u_0 by recursively solving

$$u_k(x) = \int u_{k+1}(z)\varphi_k(x,z)p_{k+1}(z \mid x)\lambda(\mathrm{d}z), \tag{E.1}$$

starting from $k = N-1$ back to $k = 0$. By Theorem 1 in Milstein et al. (2007) (with $g_n \equiv 0$ and $n = 0$ in its statement) u_0 has the following probabilistic representation

$$u_0(x) = \mathbb{E}_{0,x}\left[u_N(X_N)\prod_{k=0}^{N-1}\varphi_k(X_k, X_{k+1})\right].$$

To link this with the present work, if $\varphi_k \equiv 1$, then (E.1) becomes the discrete integral Cauchy problem $u_n(x) = \int u_{n+1}(z)p^\circ_{n+1}(z \mid x)\lambda(\mathrm{d}z)$ so that $u_0(x) = \mathbb{E}_{0,x}\left[u_N(X_N)\right]$. However, equation (E.1) can be rewritten to

$$u_k(x) = \int u_{k+1}(z)w(k+1,x)\frac{\widetilde{h}_{k+1}(k,x)}{\widetilde{h}_{k+1}(k+1,z)}p^\circ_{k+1}(z \mid x)\lambda(\mathrm{d}z)$$

(cf. equation (4.2)), and hence the following representation holds true

$$u_0(x) = \mathbb{E}_{0,x}\left[u_N(X^\circ_N)\prod_{k=0}^{N-1} w(k+1, X^\circ_k)\frac{\widetilde{h}_{k+1}(k, X^\circ_k)}{\widetilde{h}_{k+1}(k+1, X^\circ_{k+1})}\right].$$