Learning to Reject Sequential Importance Steps for Continuous-Time Bayesian Networks

Jeremy C. Weiss
University of Wisconsin-Madison
Madison, WI, US
jcweiss@cs.wisc.edu

Sriraam Natarajan
Indiana University
Bloomington, IN, US
natarasr@indiana.edu

C. David Page
University of Wisconsin-Madison
Madison, WI, US
page@biostat.wisc.edu

Abstract

Applications of graphical models often require the use of approximate inference, such as sequential importance sampling (SIS), for estimation of the model distribution given partial evidence, *i.e.*, the target distribution. However, when SIS proposal and target distributions are dissimilar, such procedures lead to biased estimates or require a prohibitive number of samples. We introduce ReBaSIS, a method that better approximates the target distribution by sampling variable by variable from existing importance samplers and accepting or rejecting each proposed assignment in the sequence: a choice made based on anticipating upcoming evidence. We relate the per-variable proposal and model distributions by expected weight ratios of sequence completions and show that we can learn accurate models of optimal acceptance probabilities from local samples. In a continuous-time domain, our method improves upon previous importance samplers by transforming an SIS problem into a machine learning one.

1 Introduction

Sequential importance sampling (SIS) is a method for approximating an intractable target distribution by sampling from a proposal distribution and weighting the sample by the ratio of target to proposal distribution probabilities at each step of the sequence. It provides the basis for many distribution approximations with applications including robotic environment mapping and speech recognition (Montemerlo et al. 2003; Wolfel and Faubel 2007). The characteristic shortcoming of importance sampling stems from the potentially high weight variance that results from large differences in the target and proposal densities. SIS compounds this problem by iteratively sampling over the steps of the sequence, resulting in sequence weights equal to the product of step weights. The sequence weight distribution is exponential, so only the high-weight tail of the sample distribution contributes substantially to the approximation. Two approaches to mitigate this problem are filtering, *e.g.*, (Doucet, Godsill, and Andrieu 2000; Fan, Xu, and Shelton 2010) and adaptive importance sampling, *e.g.*, (Cornebise, Moulines, and Olsson 2008; Yuan and Druzdzel 2003; 2007a).

One drawback of filtering is that it does not efficiently account for future evidence, and in cases of severe proposal-evidence mismatch, many resampling steps are required, leading to sample impoverishment. Akin to adaptive importance sampling, our method addresses proposal-evidence mismatch by developing "foresight", *i.e.* adaptation to approaching evidence, to guide its proposals. It develops "foresight" by learning a binary classifier dependent on approaching evidence to accept or reject each proposal. Our procedure can be viewed as the construction of a second proposal distribution, learned to account for evidence and to better approximate the target distribution, and is a new form of adaptive importance sampling.

In greater detail, our task is to recover a target distribution f^*, which can be factored variable by variable into component conditional distributions f_i^* for $i \in 1 \dots k$. The SIS framework provides a suboptimal surrogate distribution g, which likewise can be factored into a set of conditional distributions g_i. We propose a second surrogate distribution h closer to f^* based on learning conditional acceptance probabilities a_i of rejection samplers relating f_i^* and g_i. That is, to sample from h, we iteratively (re-)sample from proposals g_i and accept with probability a_i.

Our key idea is to relate the proposal g_i and target f_i^* distributions by the ratio of expected weights of sequence completions, *i.e.*, a setting for each variable from i to k, given acceptance and rejection of the sample from g_i. Given the expected weight ratio, we can recover the optimal acceptance probability a_i^* and thus f_i^*.

Unfortunately, the calculation of the expected weights, and thus the ratio, is typically intractable because of the exponential number of sequence completions. Instead, we can approximate it using machine learning. First, we show that the expected weight ratio equals the odds of accepting a proposal from g_i under the f_i^* distribution. Then, transforming the odds to a probability, we can learn a binary classifier for the probability of acceptance under f_i^* given the sample proposal from g_i. Finally, we show how to generate examples to train a classifier to make the optimal accept/reject decision. We specifically examine the application of our rejection-based SIS (ReBaSIS) algorithm to continuous-time Bayesian networks (CTBNs) (Nodelman, Shelton, and Koller 2002), where inference (even approximate) is NP-hard (Sturlaugson and Sheppard 2014).

We proceed as follows. First we present an example and related work, and we outline the problem of SIS and the identification of good proposal distributions. Then, we define rejection sampling within SIS and show how to approximate the target distribution via binary classification. Finally, we extend our analysis to CTBNs and describe experiments that show the empirical advantages of our method over previous CTBN importance samplers.

1.1 An Illustrative Example

Figure 1 describes our method in the simplest relevant example: a binary-state Markov chain. For our example, let $k = 3$: then we have evidence that $z_3 = 1$. One possible sample procedure could be: S, accept z_1^2, reject z_2^2, accept z_2^1, reject z_3^2, accept z_3^1, T, giving us the path: S, z_1^2, z_2^1, z_3^1, T. Note that if the proposal g_3: $z_2^1 \rightarrow z_3^1$ were very improbable under g but not f (*i.e.*, proposal-evidence mismatch), all samples running through z_2^1 would have very large weight. By introducing the possibility of rejection at each step, our procedure can learn to reject samples to z_3^2, reducing the importance sampling weight, and learn to enter states z_2^1 and z_2^2 proportionally to $f(\cdot|e)$, *i.e.*, develop "foresight".

1.2 Related Work

As mentioned above, batch resampling techniques based on rejection control (Liu, Chen, and Wong 1998; Yuan and Druzdzel 2007b) or sequential Monte Carlo (SMC) (Doucet, Godsill, and Andrieu 2000; Fan, Xu, and Shelton 2010), *i.e.* particle filtering, can mitigate the SIS weight variance problem, but they can lead to reduced particle diversity, especially when many resampling iterations are required. Particle smoothing (Fan, Xu, and Shelton 2010) combats particle impoverishment, but the exponentially-large state spaces used in CTBNs limit its ability to find alternative, probable sample histories. Previous adaptive importance sampling methods rely on structural knowledge and other inference methods, *e.g.*, (Cheng and Druzdzel 2000; Yuan and Druzdzel 2003), to develop improved proposals, whereas our method learns a classifier to help guide samples through regions of proposal-evidence mismatch. One interesting idea combining work in filtering and adaptive importance sampling is the SMC^2 algorithm (Chopin et al. 2011), which maintains a sample distribution over both particles and parameters determining the proposal distribution, resampling along either dimension as necessary. The method does not anticipate future evidence, so it may complement our work, which can similarly be used in the SMC framework. Other MCMC (Rao and Teh 2011) or particle MCMC (Andrieu, Doucet, and Holenstein 2010) methods may have trouble in large state spaces (*e.g.*, CTBNs) with multiple modes and low density regions in between, especially if there is proposal-evidence mismatch.

2 Background

We begin with a review of importance sampling and then introduce our surrogate distribution. Let f be a p.d.f. defined on an ordered set of random variables $Z = \{Z_1, \ldots, Z_k\}$

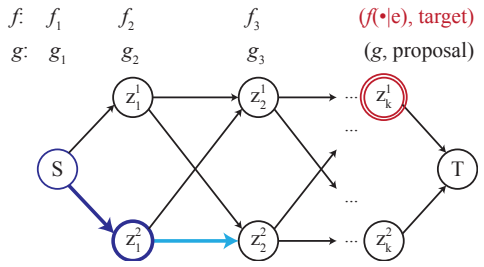

Figure 1: A source-to-sink representation of a binary-state Markov chain with evidence at z_k (red). Distributions f and g are defined over paths from source S to sink T and are composed of element-wise distributions f_i and g_i. For a sample at state z_1^2 (dark blue), an assignment to z_2^2 is proposed (light blue) according to g_2. To mimic sampling from $f_2^* = f_2(\cdot|e)$, the proposed assignment is accepted with probability proportional to the ratio of expected weights of path completions from z_2^2 and z_1^2 to T.

over an event space Ω. We are interested in the conditional distribution $f(z|e)$, where evidence e is a set of observations about a subset κ of values $\{Z_i = z_i\}_{i\in\kappa}$. For fixed e, we define our target p.d.f. $f^*(z) = f(z|e)$. Let $g(z)$ be a surrogate distribution from which we can sample such that if $f^*(z) > 0$ then $g(z) > 0$. Then for any subset $\mathcal{Z} \subseteq \Omega$, we can approximate f^* with n weighted samples from g:

$$\int_{z\in\mathcal{Z}} f^*(z)dz = \int_{\mathcal{Z}} \frac{f^*(z)}{g(z)} g(z)dz \approx \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[z^i\in\mathcal{Z}]\frac{f^*(z^i)}{g(z^i)} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[z^i\in\mathcal{Z}]w^i$$

where $\mathbb{1}[z^i\in\mathcal{Z}]$ is the indicator function with value 1 if $z^i \in \mathcal{Z}$ and 0 otherwise, and w^i is the importance sample weight.

We design a second surrogate $h(z)$ with the density corresponding to accepting a sample from g:

$$h(z) = g(z)a(z)\left(\int_{\Omega} g(\zeta)a(\zeta)d\zeta\right)^{-1} \quad (1)$$

where $a(z)$ is the sample acceptance probability from g. The last term in Equation 1 is a normalizing functional (of g and a) to ensure that $h(z)$ is a density. Procedurally, we sample from h by (re-)sampling from g and accepting with probability a. The approximation of f^* with h is given by:

$$\int_{\mathcal{Z}} f^*(z)dz = \int_{\mathcal{Z}} \frac{f^*(z)}{g(z)}\frac{g(z)}{h(z)}h(z)dz \approx \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[z^i\in\mathcal{Z}]w^i_{f^*g}w^i_{gh}$$

with weights $w^i_{f^*h} = w^i_{f^*g}w^i_{gh}$. To ensure that h has the support of f^*, we require that both a and g are non-zero everywhere f^* is non-zero.

Now we can define our optimal resampling density $h^*(z)$ using the optimal choice of acceptance probability $a^*(z) =$

$\min(1, f^*(z)/\alpha g(z))$, where $\alpha \geq 1$ is a constant determining the familiar rejection sampler "envelope": $\alpha g(z)$. The density $h^*(z)$ is optimal in the sense that, for appropriate choice of α such that $f^*(z) < \alpha g(z)$ for all z, $h^*(z) = f^*(z)$. When $h^*(z) = f^*(z)$, the importance weights are exactly 1, and the effective sample size is n.

In many applications the direct calculation of $f^*(z)$ is intractable or impossible and thus we cannot directly recover $a^*(z)$ or $h^*(z)$. However, we can still use these ideas to find $h^*(z)$ through sequential importance sampling (Liu, Chen, and Wong 1998), which we describe next.

2.1 Sequential Importance Sampling

Sequential importance sampling (SIS) is used when estimating the distribution f^* over the factorization of Z. In time-series models, Z_i is a random variable over the joint state corresponding to a time step; in continuous-time models, Z_i is the random variable corresponding to an interval. Defining $z_{j\leftarrow i} = \{z_j, z_{j-1}, \ldots, z_i\}$ for $i, j \in \{1, \ldots, k\}$ and $j \geq i$, we have the decomposition:

$$\begin{aligned} f^*(z) &= p(z_1, \ldots, z_k|e) \\ &= g_1(z_1|e)w_1(z_1|e) \\ &\quad \prod_{i=2}^{k} g_i(z_i|z_{(i-1)\leftarrow 1}, e)w_i(z_i|z_{(i-1)\leftarrow 1}, e) \quad (2) \end{aligned}$$

where $p(\cdot)$ is the probability distribution under f. Equation 2 substitutes p with p.d.f. g_i by defining functions g_i and $w_i(\cdot) = p(\cdot)/g_i(\cdot)$ and requiring g_i to have the same support as p. Then we define g by the composition of g_i: $g(z|e) = g_1(z_1|e)\prod_{i=2}^{k} g_i(z_i|z_{(i-1)\leftarrow 1}, e)$, and likewise for w. To generate a sample z^j from proposal distribution $g(z|e)$, SIS samples each z_i in order from 1 to k.

3 Methods

In this section we relate the target and proposal decompositions. Recall that we are interested in sampling directly from the conditional distribution $f^*(z) = p(z|e)$ for fixed e. We define interval distributions $f_1^*(z_1)$ and $f_i^*(z_i|z_{(i-1)\leftarrow 1})$ such that $f^*(z)$ can be factored into the interval distributions: $f^*(z) = f_1^*(z_1)\prod_{i=2}^{k} f_i^*(z_i|z_{(i-1)\leftarrow 1})$. We define the interval distributions:

$$f_i^*(z_i|z_{(i-1)\leftarrow 1}) = \frac{p(e|z_{i\leftarrow 1})}{p(e|z_{(i-1)\leftarrow 1})} p(z_i|z_{(i-1)\leftarrow 1})$$

for $i > 1$ and $p(e|z_{(i-1)\leftarrow 1}) > 0$, and $f^*(z_1) = p(e|z_1)p(z_1)/p(e)$ for $i = 1$ and $p(e) > 0$. Then, by the law of probability, we have:

$$f_i^*(z_i|z_{(i-1)\leftarrow 1}) = \frac{\mathbb{E}_f[\mathbb{1}[e, z]|z_{i\leftarrow 1}]}{\mathbb{E}_f[\mathbb{1}[e, z]|z_{(i-1)\leftarrow 1}]} p(z_i|z_{(i-1)\leftarrow 1}). \quad (3)$$

The indicator function $\mathbb{1}[e, z]$ is shorthand for $\mathbb{1}[\bigcap_{l\in\kappa}\{z_l = e_l\}|z]$ and takes value 1 if the evidence matches z and 0 otherwise. Note that in the sampling framework, Equation 3 corresponds to sampling from the unconditioned proposal distribution $p(z_i|z_{(i-1)\leftarrow 1})$ and calculating the expected weight of sample completions $z_{k\leftarrow(i+1)}$ and $z_{k\leftarrow i}$, given by the indicator functions. This procedure describes the expected outcome obtained by forward sampling with rejection.

However, when f^* and f are highly dissimilar, the vast majority of samples from f will be rejected, *i.e.*, $\mathbb{1}[e, z] = 0$ for most z. It may be better to sample from a proposal g with weight function $w = f/g$ so that sampling leads to fewer rejections. Substituting g in Equation 3, we get:

$$f_i^*(z_i|z_{(i-1)\leftarrow 1}) = \frac{\mathbb{E}_g[w^a_{k\leftarrow i}]}{\mathbb{E}_g[w^r_{k\leftarrow i}]} g_i(z_i|z_{(i-1)\leftarrow 1}). \quad (4)$$

The terms $\mathbb{E}_g[w^a_{k\leftarrow i}]$ and $\mathbb{E}_g[w^r_{k\leftarrow i}]$ are the expected forward importance sampling weights of $z_{k\rightarrow i}$ given acceptance (a) or rejection (r) of proposed assignment z_i. The derivation of Equation 4 is provided in the Appendix.

Equation 4 provides the relationship: f_i^* versus g_i, given by the ratio of expected weights of completion of sample z under acceptance or rejection of z_i. This allows us to further improve g and gives us the sampling distribution h.

3.1 Rejection Sampling to Recover f_i^*

Because Equation 4 relates the two distributions, we can generate samples from f_i^* by conducting rejection sampling from g_i. Selecting constant α such that $f_i^* \leq \alpha g_i$, *i.e.*, αg_i is the rejection envelope, we define the optimal interval acceptance probability a_i^* by:

$$a_i^*(z_i|z_{(i-1)\leftarrow 1}) = \frac{f_i^*(z_i|z_{(i-1)\leftarrow 1})}{\alpha g_i(z_i|z_{(i-1)\leftarrow 1})} = \frac{\mathbb{E}_g[w^a_{k\leftarrow i}]}{\alpha \mathbb{E}_g[w^r_{k\leftarrow i}]}. \quad (5)$$

By defining a_i^* for all i, we can generate an unweighted sample from $f^*(z)$ in $O(k \max_i(f_i^*(\cdot)/g_i(\cdot)))$ steps given the weight ratio expectations and appropriate choice of α. Thus, if we can recover a_i^* for all i, we get $h = f^*$ as desired and our procedure generates unweighted samples.

3.2 Estimating the Weight Ratio

The procedure of sampling intervals to completion depends on the expected weight ratio in Equation 4. Unfortunately, exact calculation of the ratio is impractical because the expectations involved require summing over an exponential number of terms. We could resort to estimating it from weighted importance samples: completions of z given $z_{i\leftarrow 1}$ and z given $z_{(i-1)\leftarrow 1}$. While possible, this is inefficient because (1) it would require weight estimations for every z_i given $z_{i\leftarrow 1}$, and (2) the estimation of the expected weights itself relies on importance sampling.

However, we can cast the estimation of the weight ratio as a machine learning problem of binary classification. We recognize that similar situations, in terms of state $z_{i\leftarrow 1}$, evidence e, model (providing f) and proposal g, result in similar values of a_i^*. Thus, we can learn a binary classifier $\Phi_i(z_{i\leftarrow 1}, e, f, g)$ to represent the probability of $\{$acceptance, rejection$\}$ = $\{\phi_i(\cdot), 1 - \phi_i(\cdot)\}$ as a function of the situation.

In particular, the expected weight ratio in Equation 4 is proportional to the odds under f^* of accepting the z_i sampled from g_i. The binary classifier provides an estimate of the probability of acceptance $\phi_i(z_{i\leftarrow 1}, e, f, g)$, from which

we can derive the odds of acceptance. Substituting into Equation 5, we have:

$$a_i^*(z_i|z_{(i-1)\leftarrow 1}) \approx \frac{1}{\alpha}\left(\frac{\phi_i(z_{i\leftarrow 1}, e, f, g)}{1-\phi_i(z_{i\leftarrow 1}, e, f, g)}\right) = a_i(z_i|z_{(i-1)\leftarrow 1}),$$

denoting the approximations as a_i for all i. Then our empirical proposal density h is:

$$\begin{aligned} h(z) &= h_1(z_1)\prod_{i=2}^{k} h_i(z_i) \\ &= g_1(z_1)a_1(z_1)c_1[g_1, a_1] \\ &\quad \prod_{i=2}^{k} g_i(z_i|z_{(i-1)\leftarrow 1})a_i(z_i|z_{(i-1)\leftarrow 1})c_i[g_i, a_i], \end{aligned}$$

where the c_i are the normalizing functionals as in Equation 1. We provide pseudocode for the rejection-based SIS (ReBaSIS) procedure in Algorithm 1.

3.3 Training the Classifier Φ_i

Conceptually, generating examples for the classifier Φ_i is straightforward. Given some $z_{(i-1)\leftarrow 1}$, we sample z_i from g_i, accept with probability $\rho = 1/2$, and sample to completion using g to get the importance weight. Then, an example is $(y, \mathbf{x}, w)$: $y = \{\text{accept}, \text{reject}\}$, $\mathbf{x}$ is a set of features encoding the "situation", and w is the importance weight.

The training procedure works because the mean weight of the positive examples estimates $\mathbb{E}_g[w^a_{k\leftarrow i}]$, and likewise the mean weight of the negative examples estimates $\mathbb{E}_g[w^r_{k\leftarrow i}]$. By sampling with training acceptance probability ρ, a calibrated classifier Φ_i^ρ (*i.e.*, one that minimizes the L_2 loss) estimates the probability: $\rho\mathbb{E}_g[w^a_{k\leftarrow i}]/(\rho\mathbb{E}_g[w^a_{k\leftarrow i}] + (1-\rho)\mathbb{E}_g[w^r_{k\leftarrow i}])$. The estimated probability can then be used to recover the expected weight ratio:

$$\frac{\mathbb{E}_g[w^a_{k\leftarrow i}]}{\mathbb{E}_g[w^r_{k\leftarrow i}]} \approx \left(\frac{1-\rho}{\rho}\right)\left(\frac{\phi_i^\rho(z_{i\leftarrow 1}, e, f, g)}{1-\phi_i^\rho(z_{i\leftarrow 1}, e, f, g)}\right),$$

and thus, the optimal classifier can be used to recover the rejection-based acceptance probability a_i^*. We get our particular estimator $\phi_i/(1-\phi_i)$ by setting $\rho = 1/2$, though in principle we could use other values of ρ.

In practice, we sample trajectories alternating acceptance and rejection of samples z_i to get a proportion $\rho = 1/2$. Then, we continue sampling the same trajectory to produce $2k$ training examples for a sequence of length k. We adopt this procedure for efficiency at the cost of generating related training examples. Pseudocode for the generation of examples to train the classifier is provided in the Supplement.

Inevitably, there is some cost to pay to construct h, including time for feature construction, classifier training, and classifier use. However, learning only needs to be done once, while inference may be performed many times. Also, in many challenging problems g will not produce an ESS of any appreciable size, in our case because of a mismatch with f^*, and in MCMC because of mode hopping difficulties. Our method adopts an approach complementary to particle methods to help tackle such problems, and we show its utility in the CTBN application.

Algorithm 1 Rejection-based SIS (ReBaSIS)

Input: conditional distributions $\{f_j\}$ and $\{g_j\}$ $\forall j$, evidence e; constants α, k; $i = 1$, $z = \{\}$, $w = 1$; classifiers $\{\phi_j\}$
Output: sample z with weight w

```
while i ≤ k do
    accept = false
    while not accept do
        Sample z_i ~ g_i, r ~ U[0, 1]
        a = (1/α) (φ_i(z_i,z,e,f,g) / (1−φ_i(z_i,z,e,f,g)))
        if r < a then
            accept = true
        end if
    end while
    z = {z_i, z}, w = w f_i(z_i) c[g_i(z_i), a(z_i)] / (g_i(z_i) a)
    i = i + 1
end while
return (z, w)
```

4 Continuous-Time Bayesian Networks

We extend our analysis to a continuous-time model: the continuous-time Bayesian network (CTBN) (Nodelman, Shelton, and Koller 2002), which have applications for example in anomaly detection (Xu and Shelton 2010) and medicine (Weiss, Natarajan, and Page 2012). We review the CTBN model and sampling methods and extend our analysis.

CTBNs are a model of discrete random variables $X_1, X_2, \ldots, X_d = \mathcal{X}$ over time. The model specifies a directed graph over $\mathcal{X}$ that determines the parents of each variable. The parents setting u_X is the joint state of the parents of X. Then, CTBNs assume that the probability of transition for variable X out of state x at time t is given by the exponential distribution $q_{x|u}e^{-q_{x|u}t}$ with rate parameter (intensity) $q_{x|u}$ for parents setting u. The variable X transitions from state x to x' with probability $\Theta_{xx'|u}$, where $\Theta_{xx'|u}$ is an entry in a state transition matrix Θ_u. A complete CTBN model is described by two components: a distribution $\mathcal{B}$ over the initial joint state $\mathcal{X}$, typically represented by a Bayesian network, and a directed graph over nodes representing $\mathcal{X}$ with corresponding conditional intensity matrices (CIMs). The CIMs hold the intensities $q_{x|u}$ and state transition probability matrices Θ_u.

The likelihood of a CTBN model given data is computed as follows. A trajectory is a sequence of intervals of fixed state. For each interval $[t_0, t_1)$, the duration $t = t_1 - t_0$ passes, and a variable X transitions at t_1 from state x to x'. During the interval all other variables $X_i \neq X$ remain in their current states x_i. The interval likelihood is given by:

$$\underbrace{q_{x|u}e^{-q_{x|u}t}}_{X \text{ transitions}}\ \underbrace{\Theta_{xx'|u}}_{\text{to state } x'}\ \underbrace{\prod_{x_i : X_i \neq X} e^{-q_{x_i|u}t}}_{\text{while } X_i\text{'s rest}}. \tag{6}$$

Then, the sequence likelihood is given by the product of interval likelihoods:

$$\prod_{X\in\mathcal{X}}\prod_{x\in X}\prod_{u\in U_X} q_{x|u}^{M_{x|u}} e^{-q_{x|u}T_{x|u}} \prod_{x'\neq x}\Theta_{xx'|u}^{M_{xx'|u}}$$

where the $M_{x|u}$ (and $M_{xx'|u}$) are the numbers of transitions out of state x (to state x'), and where the $T_{x|u}$ are the amounts of time spent in x given parents settings u.

4.1 Sampling in CTBNs

Evidence provided in data is typically incomplete, *i.e.*, the joint state is partially or fully unobserved over time. Thus, inference is performed to probabilistically complete the unobserved regions. CTBNs are generative models and provide a sampling framework to complete such regions. Let a trajectory z be a sequence of (state,time) pairs $(z_i = \{x_{1i}, x_{2i}, \ldots, x_{di}\}, t_i)$ for $i = \{0, \ldots, k\}$, where x_{ji} is the jth CTBN variable at the ith time, such that the sequence of t_i are in $[t_{\text{start}}, t_{\text{end}})$. Given an initial state $z_0 = \{x_{10}, x_{20}, \ldots, x_{d0}\}$, transition times are sampled for each variable x according to $q_{x|u}e^{-q_{x|u}t}$ where x is the active state of X. The variable X_i that transitions in the interval is selected based on the shortest sampled transition time. The state to which X_i transitions is sampled from $\Theta_{x_i x'_i|u}$. Then the transition times are resampled as necessary according to intensities $q_{x|u}$, noting that these intensities may be different because of potential changes in the parents setting u. The trajectory terminates when all sampled transition times exceed a specified ending time.

Previous work by Fan et al. describes a framework for importance sampling, particle filtering, and particle smoothing in CTBNs (Fan, Xu, and Shelton 2010). By sampling from truncated exponentials, they incur importance sample downweights on their samples. Recall that Equation 6 is broken into three components. The weights f_i^*/g_i are decomposed likewise: (1) a downweight for the variable x that transitions, according to the ratio of the exponential to the truncated exponential: $(1 - e^{-q_{x|u}\tau})$, (2) a downweight corresponding to a lookahead point-estimate of Θ_u assuming no other variables change until the evidence (we leave this unmodified in our implementation), and (3) a downweight for each resting variable x_i given by the ratio of not sampling a transition in time t from an exponential and a truncated exponential: $(1 - e^{-q_{x_i|u}\tau_i})/(1 - e^{-q_{x_i|u}(\tau_i - t_1)})$. Finally, the product of all interval downweights provides the trajectory importance sample weight.

4.2 Rejection-Based Importance Sampling in CTBNs

There is an equivalence between a fixed continuous-time trajectory and the discrete sequences described above. In particular, the rejection-based importance sampling method requires that the number of intervals k must be fixed, while CTBNs produce trajectories with varying numbers of intervals. Nevertheless, for any set of trajectories, we can define ϵ-width intervals small enough that at most one transition occurs per interval and that such transitions occur at the end of the interval. Then for any set of trajectories over the duration $[t_{\text{start}}, t_{\text{end}})$, we set $k = (t_{\text{end}} - t_{\text{start}})/\epsilon$. Using the memoryless property of exponential distributions, the density of a single, one-transition interval is equal to the density of the product of a sequence of ϵ-width, zero-transition intervals and one ϵ-width, one-transition interval. In practice, it is simpler to use the CTBN sampling framework so that each interval is of appreciable size. We denote the evidence e as a sequence of tuples of type (state, start time, duration): $(e_i, t_{i,0}, \tau_i)$ allowing for point evidence with zero duration, $\tau_i = 0$, and $\mathbb{1}[e, z]$ checks to see if z agrees with each e_i over the duration.

Unlike discrete time where we can enumerate the states z_i, in continuous time the calculation of w_{gh} can be time-consuming because of the normalizing integrals $c_i[g_i, a_i]$. From Equation 1, we have, omitting the conditioning:

$$\frac{g_i(z_i)}{h_i(z_i)} = \frac{\int_{\Omega_i} a_i(\zeta) g_i(\zeta) d\zeta}{a_i(z_i)}.$$

which can either be approximated by $(1 - \phi_i(\cdot))/\phi_i(\cdot)$, or calculated piecewise. The approximation method assumes the learned acceptance probability produces a proper conditional probability distribution for each situation. In our experiments, we adopt the approximation method and compare the results to show it does not introduce significant bias. We also compare using a restricted feature set where an exact, piecewise computation is possible.

Several properties of CTBNs make our approach appealing. First, CTBNs possess the Markov property; namely, the next state is independent of previous states given the current one. Second, CTBNs are homogeneous processes, so the model rate parameters are shared across intervals. We leverage these facts when learning each acceptance probability a_i. The Markov property simplifies the learned probability of acceptance $\phi_i(z_i | z_{(i-1)\leftarrow 1}, e, f, g)$ to $\phi_i(z_i | z_{(i-1)}, e, f, g)$. Homogeneity simplifies the learning process because, if $z_{(i-1)} = z_{(j-1)}$ and $t_{i,0} = t_{j,0}$ for $j \neq i$, then $\phi_i(z_i | z_{(i-1)}, e, f, g) = \phi_i(z_j | z_{(j-1)}, e, f, g)$. The degeneracy of these two cases indicates that the probability of acceptance is situation-dependent and interval index-independent, so a single classifier can be learned in place of k classifiers.

5 Experiments

We compare our learning-based rejection method (setting $\alpha = 2$) with the Fan et al. sampler (2010). We learn a logistic regression (LR) model using online, stochastic gradient descent for each CTBN variable state. An LR data example is $(y, \mathbf{x}, w)$, where y is one of $\{\text{accept}, \text{reject}\}$, $\mathbf{x}$ is a set of features, and w is the sequence completion weight. For our experiments we use per-variable features encoding the state (as indicator variables), the time from the current time to next evidence, the time from the proposed sample to next evidence, and the time from the proposed sample to next matching evidence. Except in the restricted feature set, the times are mapped to intervals $[0, 1]$ by using a $e^{-t/\lambda}$ transformation, with $\lambda = \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ to capture effects at different time scales. The restricted feature set uses linear times truncated at a maximum time of 10. This allows fort piecewise calculation of the normalizing functional Because of the high variance in weights for samples of full sequence completions, we instead choose a local weight: the weight of the sequence through the next $m = 10$ evidence times. This biases the learner to have low variance weights within the local window, but it does not bias the proposal h.

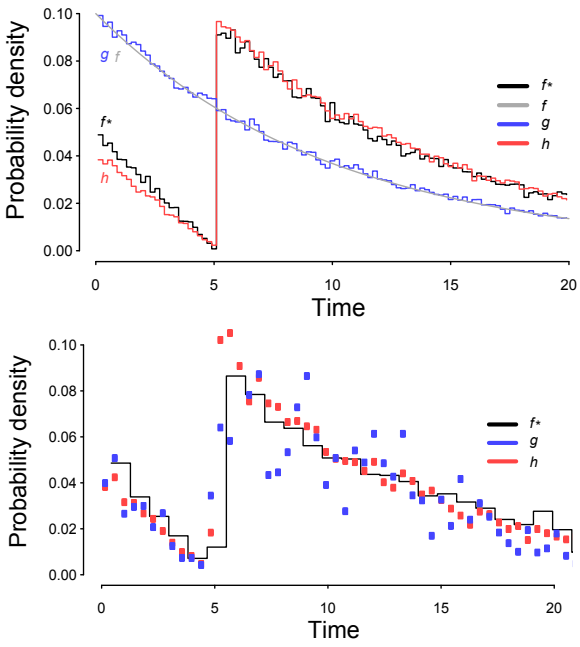

Figure 2: Approximate transition densities (top) of f^* (target), f (target without evidence), g (surrogate), and h (learned rejection surrogate) in a one-variable, binary-state CTBN with uniform transition rates of 0.1 and matching evidence at t=5. The learned distribution h closely mimics f^*, the target distribution with evidence, while g was constructed to mimic f (exactly, in this situation). All methods recover the weighted transition densities (bottom); for 20 evidence points with one at t=5 and 19 after t=20, h recovers the target distribution more precisely than g per 10^6 samples.

We analyze the performance of the rejection-based sampler by inspection of learned transition probability densities and the effective sample size (ESS) (Kong, Liu, and Wong 1994). ESS is an indicator of the quality of the samples, and a larger value is better: ESS $= 1/(\sum_{i=1}^{n}(W^i)^2)$, where $W^i = w^i/\sum_{j=1}^{n} w^j$. We test our method in several models: one-, two- and three- variable, strong-cycle binary-state CTBNs, and the part-binary, 8-variable drug model presented in the original CTBN paper (Nodelman, Shelton, and Koller 2002). The "strong-cycle" models encode an intensity path for particular joint states by shifting bit registers and adding and filling in an empty register with a 0 or 1 as in the following example. For the 3-variable strong cycle, intensities involved in the path $000 \rightarrow 001 \rightarrow 011 \rightarrow 111 \rightarrow 110 \rightarrow 100 \rightarrow 000$ are 1, and intensities are 0.1 for all other transitions. We generate sequences from ground truth models and censor each to retain only 100 point evidences with times t_i drawn randomly, uniformly over the duration [0,20). The code is provided at `http://cs.wisc.edu/~jcweiss/aaai15`.

Figure 2 (top) illustrates the ability of h to mimic f^*, the target distribution, in a one-node binary-state CTBN with matching evidence at $t = 5$. The Fan et al. proposal g matches the target density in the absence of evidence, f. However, when approaching evidence (at $t = 5$), the transition probability given evidence goes to 0 as the next transition must also occur before $t = 5$ to be a viable sequence. Only f^* and h exhibit this behavior. Figure 2 (bottom) shows the density approximations after weighting the samples, given a trajectory with evidence at $t = 5$ and 19 evidence points after $t = 20$. Each method recovers the target distribution, but h does so more precisely than g, given a fixed number of samples (one million). The proposal acceptance rate for h was measured to be 45 percent. The proposal h based on the restricted feature set for unbiased inference is provided in the Supplement.

Table 1: Geometric mean of effective sample size (ESS) over 100 sequences, each with 100 observations; ESS is per 10^5 samples. The proposal h was learned with 1000 sequences.

Model	Fan et al. (g)	Rejection SIS (h)
Strong cycle, n=1	690	6400
Strong cycle, n=2	19000	35000
Strong cycle, n=3	960	5800
Drug	29	170

Table 1 shows that the learned, rejection-based proposal h outperforms the other CTBN importance sampler g across all 4 models, resulting in an ESS 2 to 10 times larger. Generally as the number of variables increases, the ESS decreases because of the increasing mismatch between f^* and g. With an average ESS of only 29 in an 8-variable model, as we increase model sizes, we expect that g would fail to produce a meaningful sample distribution more quickly than h would.

Figure 3 shows that the weight distribution from h is narrower than that from g on the log scale, using the drug model and 10 evidence points. The interpretation is that a larger fraction of examples from h contribute substantially to the total weight, resulting in a lower variance sample distribution. For example, any sample with weight below e^{-10} has negligible relative weight and does not substantially affect the sample distribution. There are many fewer such samples generated from h than from g.

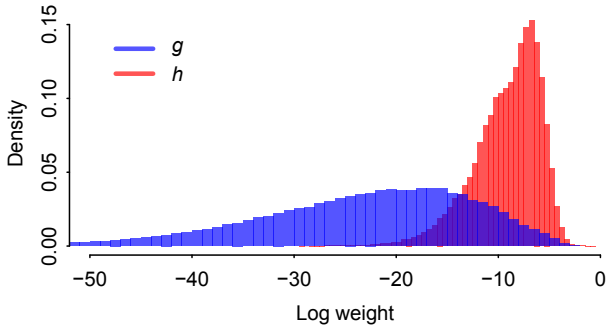

Figure 3: Distribution of log weights. For sample completions of a trajectory with 10 evidence points in the drug model, the distribution of log weights using h is much narrower than the distribution of log weights using g (bottom).

6 Conclusion

Our work has demonstrated that machine learning can be used to improve sequential importance sampling via a rejection sampling framework. We showed that the proposal and target distributions are related by an expected weight ratio, and that the weight ratio can be estimated by the probabilistic output of a binary classifier learned from weighted, local importance samples. We extended the algorithm to CTBNs, where we found experimentally that using our learning algorithm produces a sampling distribution closer to the target and generates more effective samples. Continued investigations are warranted, including the use of non-parametric learning algorithms, a procedure for expanding local approximations, and extensions to other graphical models, where conjugacy of the acceptance probability with the proposal distribution could lead to improved performance.

7 Acknowledgments

We gratefully acknowledge the CIBM Training Program grant 5T15LM007359, the NIGMS grant R01GM097618, the NLM grant R01LM011028, and the NSF grant IIS-1343940.

A Appendix

To derive Equation 4, we relate the target densities $f_i^*(z_i|z_{(i-1)\leftarrow 1})$ with the proposal densities $g_i(z_i|z_{(i-1)\leftarrow 1})$ via the (standard) derivation of Equation 3 using Bayes' theorem, the law of total probability, and substitution:

$$
\begin{aligned}
&f_i^*(z_i|z_{(i-1)\leftarrow 1}) \\
&= \frac{p(e|z_{i\leftarrow 1})}{p(e|z_{(i-1)\leftarrow 1})} p(z_i|z_{(i-1)\leftarrow 1}) \\
&= \frac{\sum_{z_{k\leftarrow i+1}} p(e|z_{k\leftarrow i+1}, z_{i\leftarrow 1}) p(z_{k\leftarrow i+1}|z_{i\leftarrow 1})}{\sum_{z_{k\leftarrow i}} p(e|z_{k\leftarrow i}, z_{i-1\leftarrow 1}) p(z_{k\leftarrow i}|z_{i-1\leftarrow 1})} p(z_i|z_{i-1\leftarrow 1}) \\
&= \frac{\sum_{z_{k\leftarrow i+1}} \mathbb{1}[\bigcap_{l\in\kappa}\{z_l{=}e_l\}|z] p(z_{k\leftarrow i+1}|z_{i\leftarrow 1})}{\sum_{z_{k\leftarrow i}} \mathbb{1}[\bigcap_{l\in\kappa}\{z_l{=}e_l\}|z] p(z_{k\leftarrow i}|z_{i-1\leftarrow 1})} p(z_i|z_{i-1\leftarrow 1}) \\
&= \frac{\mathbb{E}_g[\mathbb{1}[e,z]\prod_{j=i+1}^{k} w_j(z_j|z_{j-1\leftarrow 1})|z_{i\leftarrow 1}]}{\mathbb{E}_g[\mathbb{1}[e,z]\prod_{j=i}^{k} w_j(z_j|z_{j-1\leftarrow 1})|z_{i-1\leftarrow 1}]} p(z_i|z_{i-1\leftarrow 1}) \\
&= \frac{\mathbb{E}_g[\mathbb{1}[e,z]\prod_{j=i}^{k} w_j(z_j|z_{j-1\leftarrow 1})|z_{i\leftarrow 1}]}{\mathbb{E}_g[\mathbb{1}[e,z]\prod_{j=i}^{k} w_j(z_j|z_{j-1\leftarrow 1})|z_{i-1\leftarrow 1}]} g_i(z_i|z_{i-1\leftarrow 1}) \\
&= \frac{\mathbb{E}_g[w^a_{k\leftarrow i}]}{\mathbb{E}_g[w^r_{k\leftarrow i}]} g_i(z_i|z_{(i-1)\leftarrow 1}). \qquad \square
\end{aligned}
$$

References

Andrieu, C.; Doucet, A.; and Holenstein, R. 2010. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72(3):269–342.

Cheng, J., and Druzdzel, M. J. 2000. AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks. *Journal of Artificial Intelligence Research* 13(1):155–188.

Chopin, N.; Jacob, P.; Papaspiliopoulos, O.; et al. 2011. Smc2: A sequential Monte Carlo algorithm with particle Markov chain Monte Carlo updates. *JR Stat. Soc. B (2012, to appear)*.

Cornebise, J.; Moulines, E.; and Olsson, J. 2008. Adaptive methods for sequential importance sampling with application to state space models. *Statistics and Computing* 18(4):461–480.

Doucet, A.; Godsill, S.; and Andrieu, C. 2000. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing* 10(3):197–208.

Fan, Y.; Xu, J.; and Shelton, C. R. 2010. Importance sampling for continuous time Bayesian networks. *The Journal of Machine Learning Research* 11:2115–2140.

Kong, A.; Liu, J. S.; and Wong, W. H. 1994. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association* 89(425):278–288.

Liu, J. S.; Chen, R.; and Wong, W. H. 1998. Rejection control and sequential importance sampling. *Journal of the American Statistical Association* 93(443):1022–1031.

Montemerlo, M.; Thrun, S.; Koller, D.; and Wegbreit, B. 2003. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *International Joint Conference on Artificial Intelligence*, volume 18, 1151–1156. Lawrence Erlbaum Associates LTD.

Nodelman, U.; Shelton, C.; and Koller, D. 2002. Continuous time Bayesian networks. In *Uncertainty in artificial intelligence*, 378–387. Morgan Kaufmann Publishers Inc.

Rao, V., and Teh, Y. W. 2011. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Uncertainty in Artificial Intelligence*.

Sturlaugson, L., and Sheppard, J. W. 2014. Inference complexity in continuous time bayesian networks. In *Uncertainty in Artificial Intelligence*.

Weiss, J.; Natarajan, S.; and Page, D. 2012. Multiplicative forests for continuous-time processes. In *Advances in Neural Information Processing Systems 25*, 467–475.

Wolfel, M., and Faubel, F. 2007. Considering uncertainty by particle filter enhanced speech features in large vocabulary continuous speech recognition. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, IV–1049. IEEE.

Xu, J., and Shelton, C. R. 2010. Intrusion detection using continuous time Bayesian networks. *Journal of Artificial Intelligence Research* 39(1):745–774.

Yuan, C., and Druzdzel, M. J. 2003. An importance sampling algorithm based on evidence pre-propagation. In *Uncertainty in Artificial Intelligence*, 624–631. Morgan Kaufmann Publishers Inc.

Yuan, C., and Druzdzel, M. J. 2007a. Importance sampling for general hybrid Bayesian networks. In *Artificial intelligence and statistics*.

Yuan, C., and Druzdzel, M. J. 2007b. Improving importance sampling by adaptive split-rejection control in Bayesian networks. In *Advances in Artificial Intelligence*. Springer. 332–343.