Adversarial Sequential Monte Carlo

Kira Kempinska
Department of Security and Crime Science
University College London
London, WC1E 6BT
kira.kowalska.13@ucl.ac.uk

John Shawe-Taylor
Department of Computer Science
University College London
London, WC1E 6BT
j.shawe-taylor@cs.ucl.ac.uk

Abstract

How can we perform efficient inference in directed probabilistic models with intractable posterior distributions? We introduce a new technique for improving finite-sample inference approximations by learning highly flexible proposal distributions for sequential importance samplers, such as particle filters. We represent proposal distributions as implicit generative models, that is models that you can sample from but which do not have an explicit parametric form, and train them using variational inference rephrased as a two-player game, hence establishing a principled connection between Sequential Monte Carlo (SMC) and Generative Adversarial Networks (GANs). Our approach achieves state-of-the-art performance on a range of inference problems.

1 Introduction

Sequential Monte Carlo, or particle filtering, is a popular class of methods for Bayesian inference that approximate an intractable target distribution by drawing samples from a series of simpler intermediate distributions. SMC methods have traditionally been used for filtering in state-space models [6, 14], but have since shown state-of-the-art results in a far broader range of models, including factor graphs, hierarchical Bayesian models [10] and general probabilistic programs [16, 15].

Critical to the performance of SMC is the choice of appropriate proposal distributions for transitioning from one intermediate distribution to the next. Theoretically optimal proposal distributions [3] are in general intractable, thus in practice they are often approximated using a prior distribution, as in the case of bootstrap particle filter [6]. More adaptable proposal distributions have recently been proposed by [13, 7] who use neural networks with parametrised distributions as output layers to design distributions of varying complexity. Their quality is largely dependent on the suitability of their parametrisation to the problem at hand, however, as well as the availability of large volumes of training data. All in all, the ability to automatically learn appropriate proposal distributions for any inference problem is still a real need that we need to address before SMC methods can be automatically applied to new models and problems.

In this paper, we investigate the use of Generative Adversarial Networks (GANs) [5] for constructing highly flexible proposal distributions for Sequential Monte Carlo. GANs provide a tool to learn implicit proposal models from data, i.e. models that define a stochastic procedure that directly generates proposal samples [12, 8], which in contrast to parametric proposal models [13, 7], can learn proposal models of arbitrarily complexity. The proposals are black-box sample generators that are adversarially trained to obtain a close approximation to a target posterior distribution. They offer high flexibility and remove the need to design a suitable parametrisation for a new inference problem [11], hence advancing our work towards automatic SMC inference.

Second workshop on Bayesian Deep Learning (NIPS 2017), Long Beach, CA, USA.

2 Sequential Monte Carlo

Consider a directed graphical model comprising of N latent variables x and N observed variables y, whose joint distribution factorises as

$$p(x_{1:N}, y_{1:N}) = p(x_1)p(y_1|x_1)\prod_{n=2}^{N} p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1}) \tag{1}$$

This form subsumes common state-space models, such as Hidden Markov Models (HMMs), as well as non-Markovian models, such as Gaussian processes.

The goal of sequential importance sampling (SIS) is to approximate the posterior distribution $p(x_{1:N}|y_{1:N})$ with a weighted set of samples drawn from a (presumably simpler) proposal distribution $q(x_{1:N}|y_{1:N})$ [4]. Any form of proposal distribution can be used in principle, but a particularly convenient one takes the same factorisation as the true posterior $q(x_{1:N}|y_{1:N}) = q_1(x_1|y_1)\prod_{n=2}^{N} q_n(x_n|x_{1:n-1}, y_{1:n})$. The samples are weighed according to the *unnormalised weight* function

$$w(x_{1:N}) = \frac{p(x_{1:N}, y_{1:N})}{q(x_{1:N})} = w(x_{1:N-1})\frac{p(x_N|x_{1:N-1})p(y_N|x_{1:N}, y_{1:N-1})}{q(x_N|x_{1:N-1}, y_{1:N})} \tag{2}$$

SIS exploits the factorisation of the proposal and the weight function to approximate the posterior in a sequential fashion using a single forward pass. The sequential approximation is known in the wider literature as the problem of *filtering*.

3 Method

3.1 Objective for proposal learning

We want to find a proposal distribution $q_\phi(x_{1:N}|y_{1:N})$ parametrised by ϕ that closely approximates the target posterior distribution $p(x_{1:N}|y_{1:N})$. We frame the problem in terms of finding proposal parameters ϕ^* that minimise the KL divergence between the proposal and the target densities:

$$\begin{aligned}\min_\phi KL[q_\phi(x_{1:N}|y_{1:N})||p(x_{1:N}|y_{1:N})] &= \min_\phi E_{q_\phi} \log \frac{q_\phi(x_{1:N}|y_{1:N})}{p(x_{1:N}|y_{1:N})} \\ &= \min_\phi E_{q_\phi}[\log \frac{q_\phi(x_{1:N}|y_{1:N})}{p(x_{1:N})} - \log p(y_{1:N}|x_{1:N})] + \log p(y_{1:N})) \end{aligned} \tag{3}$$

We ignore the marginal likelihood $\log p(y_{1:N})$, since it does not depend on ϕ, and use the factorisation of our joint distribution in (1) to arrive at our final optimisation objective for $n = 1, ..., N$:

$$\min_{\phi_n} E_{q_{\phi_n}}\left[\log \frac{q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})}{p(x_n|x_{1:n-1})} - \log p(y_n|x_{1:n}, y_{1:n-1})\right] \tag{4}$$

where q_{ϕ_n} are factors of the full proposal $q_\phi(x_{1:N}|y_{1:N}) = q_1(x_1|y_1)\prod_{n=2}^{N} q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})$.

3.2 Adversarial learning

We represent each factor q_{ϕ_n} of the full proposal as an *implicit probabilistic model*, that is a model that specifies a stochastic procedure that directly generates a latent variable x_n [12]. Implicit proposal models use an input noise variable $g(\epsilon)$ and a deep network architecture x_{ϕ_n} parametrised by ϕ_n to flexibly characterise a wide range of proposal distributions:

$$x_n = x_{\phi_n}(x_{1:n-1}, y_{1:n}, \epsilon'), \qquad \epsilon' \sim g(\epsilon) \tag{5}$$

Implicit formulation requires adversarial training to find ϕ_n that optimise our learning objective in (4). We introduce a dicriminator T with the following objective

$$\max_T E_{q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})} \log \sigma(T(x_{1:n}, y_{1:n})) + E_{p(x_n|x_{1:n-1})} \log(1 - \sigma(T(x_{1:n}, y_{1:n}))) \tag{6}$$

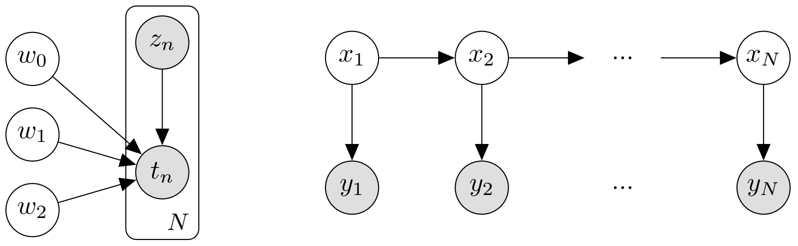

Figure 1: Directed graphical model for (left) non-conjugate regression and (right) first-order Markov chain.

where σ denotes the sigmoid function. Intuitively, T tries to distinguish between pairs $(x_n, y_{1:n})$ with x_n sampled using the prior $p(x_n|x_{1:n-1})$ from those sampled using our proposal $q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})$, where $y_{1:n}$ are real observations. The optimal discriminator T^* is given by

$$T^*(x_{1:n}, y_{1:n}) = \log \frac{q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})}{p(x_n|x_{1:n-1})}. \tag{7}$$

Hence, we can use (5) and (7) to arrive at the objective for adversarial training:

$$\min_{\phi_n} E_\epsilon(T^*(x_{1:n-1}, x_{\phi_n}, y_{1:n}) - \log p(y_n|x_{1:n-1}, x_{\phi_n}, y_{1:n-1})). \tag{8}$$

We find ϕ_n that optimise our objective for proposal learning in (4) by alteratively training the discriminator according to (6) and fixing the discriminator to optimise (8). In all experiments, we train using Adam optimiser [9] and improve the robustness of training using Adaptive Contrast technique recently proposed by [11].

3.3 Adversarial sequential Monte Carlo

We find optimal ϕ_n off-line for $n = 1, ..., N$ and then use the trained proposals q_{ϕ_n} within an SMC algorithm called particle filters [4]. Since the proposal distributions are represented as implicit models, that is, models that we can sample from but whose density we cannot evaluate, we assume a uniform density across all their samples. This simplifies unnormalised particle weights in (2) to:

$$w(x_{1:N}) = p(x_{1:N}, y_{1:N}) = w(x_{1:N-1})p(x_N|x_{1:N-1})p(y_N|x_{1:N}, y_{1:N-1}) \tag{9}$$

4 Experiments

4.1 A linear regression

We begin by illustrating our method with a non-conjugate polynomial regression model proposed by [13], with global-only latent variables. The model is shown in Figure 1 (left). The model defines a Laplace prior on the weights and Student-t likelihoods, giving us

$$\begin{aligned} p(w_d) &= \mathit{Laplace}(0, 10^{1-d}) \\ p(t_n|w_0, w_1, w_2, z_n) &= t_v(w_0 + w_1 z_n + w_2 z_n^2, \epsilon^2) \end{aligned} \tag{10}$$

for $d = 0, 1, 2$, $n = 1, ..., N$, fixed $v = 4$, $\epsilon = 1$ and $z_n \in (-10, 10)$ uniformly. Our goal is to infer latent variables $x \equiv \{w_0, w_1, w_2\}$ given observed variables $y \equiv \{z_n, y_n\}_{n=1}^N$. Since our latent variables are independent of time $n = 1, ..., N$, we train a single proposal distribution $q_\phi(w_{0:2}|z_{1:N}, y_{1:N})$ that can simultaneously sample all latent variables given a sequence of observations. We arrive at samples approximating the posterior $p(w_{0:2}|z_{1:N}, y_{1:N})$ by applying a single step of importance sampling to samples from the proposal distribution.

We represent our proposal by a two-layer network with 128 hidden units in each layer. As in the work of [11], the network accepts the noise ϵ as additional input to *implicitly* model the proposal distribution. We train the proposal against a slightly more powerful discriminator represented by a five-layer network with 256 hidden units in each layer. We show exemplary samples from the

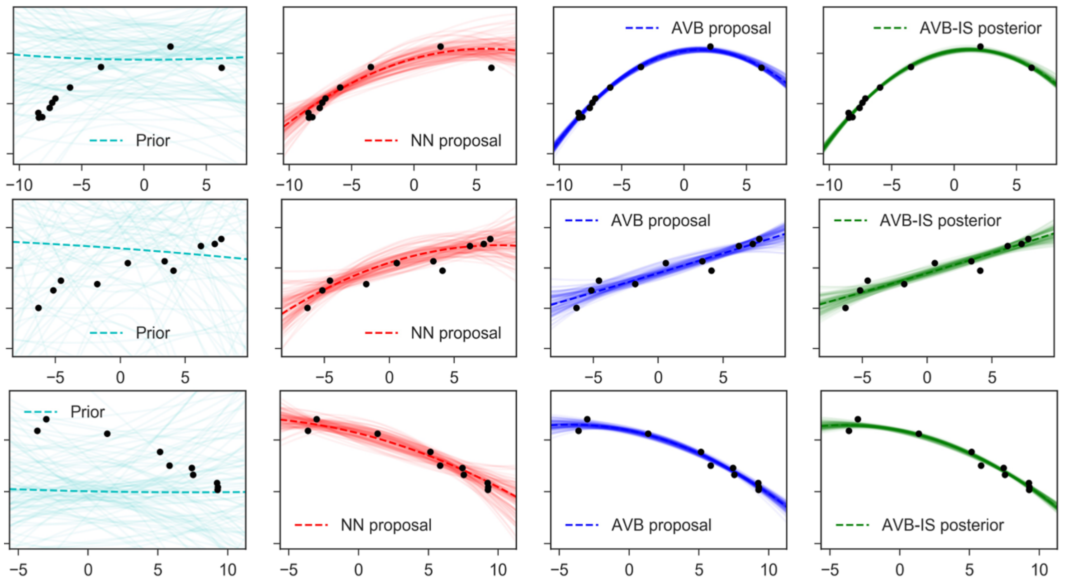

Figure 2: Exemplary output in the polynomial regression case study. Plots show 100 regression curves, each curve estimated from one sample of weights, and the mean curve marked as a dashed line. Our proposal (AVB) yields estimates close to proposal by [13] (NN), but slightly less diffused. Black dots represent true observations corresponding to different true weights in each row.

proposal and the posterior after importance sampling in Figure 2. Alongside, we show samples from a proposal given by the same network, but with a mixture of three Gaussians at each output [13] that *explicitly* parametrise the proposal distribution. In this toy example, both approaches seem to learn proposals that are close to the true posterior.

4.2 A benchmark nonlinear state-space model

We now consider a more complex temporal model given by the following dynamics

$$\begin{aligned} p(x_n|x_{n-1}) &= \mathcal{N}(x_n; f(x_{n-1}), \sigma_v^2), \qquad p(x_1) = \mathcal{N}(x_1; 0, 5) \\ p(y_n|x_n) &= \mathcal{N}(y_n; g(x_n), \sigma_w^2) \\ f(x_{n-1}) &= x_{n-1}/2 + 25x_{n-1}/(1+x_{n-1}^2), \qquad g(x_n) = x_n^2/20 \end{aligned} \tag{11}$$

for fixed $\theta = (\sigma_v, \sigma_w) = (\sqrt{10}, 1)$. The nonlinear model, shown in Figure 1 (right), is often used to assess the performance of SMC methods [2, 1]. The model is *time-invariant*, that is, $p(x_n|x_{n-1})$ and $p(y_n|x_n)$ are independent of n, hence we only train a single proposal distribution $q_\phi(x_n|x_{n-1}, y_n)$ that can be used for sampling x_n for all $n = 1, ..., N$. The posterior density $p_\theta(x_{1:N}|y_{1:N})$ is highly multimodal due uncertainty about the sign of the state x_n which is only observed through its square.

We use a simple two-layer network with 300 hidden units in each layer for the proposal and a five-layer network with 300 hidden units in each layer for the discriminator. We adversarially train them using a dataset of 100 points sampled according to the generative process in (11). We compare the performance of our approach (AVB-PF) against bootstrap particle filters (PF) as well as particle filters with a proposal represented by a neural network of the same dimensions as our proposal, but with density *explicitly* parametrised by a mixture of three Gaussians at each output (NN-PF) [13]. In contrast to NN-PF, we include the noise ϵ as additional input to the proposal model instead of adding it at the very end, thereby allowing the proposal network to learn complex probability distributions. Results show improved performance in terms of sample quality and marginal log likelihood (see Figures 3 and 4).

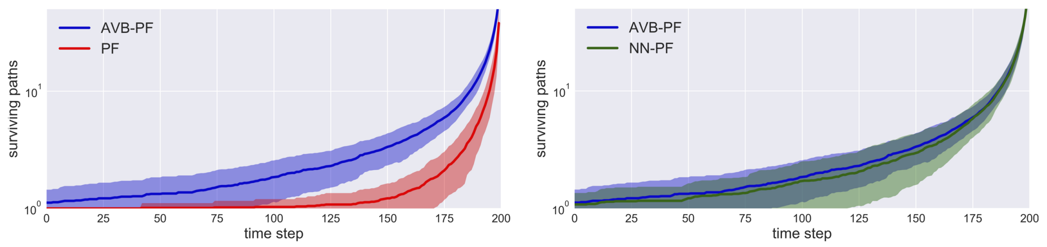

Figure 3: Adversarially learnt proposals reduce particle degeneracy in the nonlinear state-space model. Here we show the number of unique particles which survive over the course of 200 time steps, using 100 particles. Plots show mean and standard deviation over 100 sampled sequences.

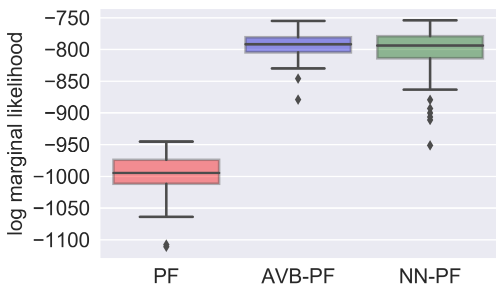

Figure 4: Box-plots for log-marginal likelihood estimates over 100 sampled sequences of length $N = 200$, using 100 particles. Our inference method explains data better, both in terms of median log-likelihood and its variability across different data samples.

References

[1] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[2] Arnaud Doucet, Nando De Freitas, and NJ Gordon. Sequential monte carlo methods in practice. series statistics for engineering and information science, 2001.

[3] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[4] Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET, 1993.

[7] Shixiang Gu, Zoubin Ghahramani, and Richard E Turner. Neural adaptive sequential monte carlo. In *Advances in Neural Information Processing Systems*, pages 2629–2637, 2015.

[8] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[9] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Fredrik Lindsten, Adam M Johansen, Christian A Naesseth, Bonnie Kirkpatrick, Thomas B Schön, JAD Aston, and Alexandre Bouchard-Côté. Divide-and-conquer with sequential monte carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458, 2017.

[11] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

[12] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[13] Brooks Paige and Frank Wood. Inference networks for sequential monte carlo in graphical models. In *International Conference on Machine Learning*, pages 3040–3049, 2016.

[14] Michael K Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

[15] Adrien Todeschini, François Caron, Marc Fuentes, Pierrick Legrand, and Pierre Del Moral. Biips: software for bayesian inference with interacting particle systems. *arXiv preprint arXiv:1412.3779*, 2014.

[16] Frank Wood, Jan Willem Meent, and Vikash Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, pages 1024–1032, 2014.