We develop **two new inference strategies** for **performing statistically correct** and **computational efficient inference** in complicated real-world simulators.

Efficient Bayesian Inference for Nested Simulators

Bradley J. Gram-Hansen[1,2,*] Christian Schroeder de Witt[1*]
Phil H.S. Torr[1] Yee Whye Teh[2] Atılım Güneş Baydin[1]
Tom Rainforth[2]

[1]Department of Engineering Science, [2]Department of Statistics, University of Oxford

* Equal Contribution

Background

- Simulators arise in a number of industrial and scientific domains, **encoding sophisticated generative models**[3][4].
- Probabilistic programming provides a way to perform statistical inference over simulations of events in a programmatic way.
- Simulators contain many nested sub-procedures that generate *priors* that the outer simulator program is dependent upon.

Problem

- In order leverage scalable variational inference (VI) and Markov Chain Monte Carlo (MCMC) inference methods, **we need to construct the program density for the entire simulator, nested sub-procedures inhibit this.**

Solution

- Introduce a way of replacing sub-procedures by a variational surrogate which turns an intractable program density, into a tractable density (Method 1)[2]
- Our second method learns an unbiased regressor to predict the marliginal of the sub-procedure. Once learnt, we combine that with the output of evaluating the sub-procedure to learn a approximation that can be used to construct the full program density (Method 2)[2]

Acknowledgements

BGH is supported by the EPSRC Autonomous Intelligent Machines and Systems grant (EP/L015897/1). CSW is supported by the project Free the Drones (FreeD) under the Innovation Fund Denmark and Microsoft. YWT's and TR's research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 617071. AGB and PHT are supported by EPSRC/MURI grant EP/N019474/1 and AGB is also supported by Lawrence Berkeley National Lab.

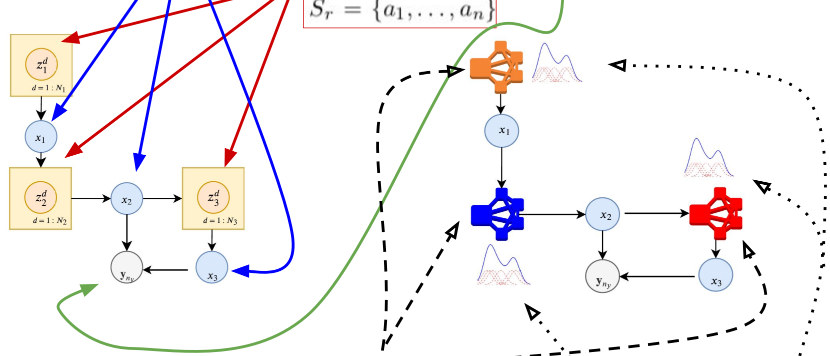
How does it work?

Can write the unnormalized simulator program density as:

$$\gamma(x_{1:n_x}|\lambda) = \prod_{j=1,a_j\notin S_r}^{n_x} f_{a_j}(x_j|\phi_j) \prod_{j=1,a_j\in S_r}^{n_x} P^{in}_{a_j}(x_j|\phi_j) \prod_{k=1}^{n_y} g_{b_k}(y_k|\psi_k)$$

subprogram densities

$$S_r = \{a_1,\dots,a_n\}$$



Method 1: Learn the Surrogate via a Variational Objective

Replace a sampling sub-procedures by using input-output pairs in order to learn an approximate surrogate:

$$P_{pr}(x_{1:n_x}|\lambda) := \prod_{j=1}^{n_x} f_{a_j}(x_j|\phi_j) \simeq q(x_{1:n_x}|\lambda;\kappa)$$

$$= \prod_{\substack{j=1\\ a_j\notin S_r}}^{n_x} f_{a_j}(x_j|\phi_j) \prod_{\substack{j=1\\ a_j\in S_r}}^{n_x} q^{in}_{a_j}(x_j|\phi_j;\eta_{a_j})$$

Here, the variational parameters can be learned by optimising a variational objective:

$$\kappa^* = \operatorname*{argmin}_{\kappa}\ J(\eta) = \operatorname*{argmin}_{\kappa} \mathbb{E}_{X\sim P_{pr}}\left[-\log q(X = x_{1:n_x}|\phi;\kappa)\right]$$

where the variational parameters for each sub-procedure are in the set:

$$\kappa = \{\eta_{a_j}; a_j \in S_r\}$$

with corresponding gradient update:

$$\nabla_{\eta_r} J(\kappa) \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{j=1}^{n_x} \mathbb{I}(r=a_j)\nabla_{\eta_r}\log(q^{in}_r(x^n_j|\phi_j;\eta_r))$$

where: $x^n_{1:n_x} \overset{i.i.d.}{\sim} P_{pr}(x_{1:n_x}|\lambda)$

Method 2: Learn the normalisation constant of the nested sub-procedure.

Alternatively, we can rewrite the unnormalized simulator program density as:

$$\gamma(x_{1:n_x}|\lambda) = \prod_{i=1,a_i\notin S_R}^{n_x} f_{a_i}(x_i;\phi_i) \prod_{j=1,a_j\in S_R}^{n_x} \frac{\gamma^{in}_{a_j}(x_j|\phi_j)}{I^{in}_{a_j}(\phi_j)} \prod_{k=1}^{n_y} g_{b_k}(y_k;\phi_k)$$

with the normalisation constant (marginal)

$$I^{in}_{a_j}(\phi_j) = \int \gamma^{in}_{a_j}(x_j = z^{a_j}_{1:n_z}|\phi_j)dz^{a_j}_{1:n_z}$$

The marginal is not a normalized density, hence approximate it using a regressor:

$$P^{in}_{a_j}(x_j|\phi_j) \simeq \frac{\gamma^{in}_{a_j}(x_j|\phi_j)}{R_{a_j}(\phi_j;\tau)}$$

We learn an unbiased mean using the L2-Norm[1]:

$$\mathcal{L} = \mathbb{E}_{\phi_j}\left\{\mathbb{E}_{\hat{I}_{a_i}}\left[\left\|R_{a_j}(\phi_j;\tau) - \hat{I}_{a_i}(\phi_j)\right\|_2^2 \,\middle|\, \phi_j\right]\right\}$$

$$\nabla_\kappa\mathcal{L} = \mathbb{E}_{\phi_j}\left\{\mathbb{E}_{\hat{I}_{a_i}}\left[\nabla_\kappa\left\|R_{a_j}(\phi_j;\tau) - \hat{I}_{a_i}(\phi_j)\right\|_2^2 \,\middle|\, \phi_j\right]\right\}$$

Once trained, we can run inference on the approximate, unnormalised, target program density:

$$\gamma(1:x_{n_x}|\lambda) = \prod_{i=1,a_i\notin S_R}^{n_x} f_{a_i}(x_i;\phi_i) \prod_{k=1,b_k}^{n_y} g_{b_k}(y_k;\phi_k) \prod_{j=1,a_j\in S_R}^{n_x} \frac{\gamma^{in}_{a_j}(x_j|\phi_j)}{R_{a_j}(\phi_j;\tau)}$$

References

[1] Casella, George, and Roger L. Berger. *Statistical inference*. Vol. 2. Pacific Grove, CA: Duxbury, 2002.
[2] Gram-Hansen, Bradley, Christian Schröder de Witt, Tom Rainforth, Philip HS Torr, Yee Whye Teh, and Atılım Güneş Baydin. "*Efficient Bayesian Inference for Nested Simulators*" AABI (2019).
[3] Baydin, Atilim Gunes, Lei Shao, Wahid Bhimji, Lukas Heinrich, Bradley Gram-Hansen, et al. "*Efficient probabilistic inference in the quest for physics beyond the standard model.*" In Advances in Neural Information Processing Systems, pp. 5460-5473. 2019.
[4] Smith, T., N. Maire, A. Ross, M. Penny, N. Chitnis, A. Schapira, A. Studer et al. "*Towards a comprehensive simulation model of malaria epidemiology and control.*" *Parasitology* 135, no. 13 (2008): 1507-1516.

EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines & Systems

Download the full paper and code with this QR code