Nesting Probabilistic Programs – Supplementary Material

Tom Rainforth
Department of Statistics
University of Oxford
rainforth@stats.ox.ac.uk

A PROOFS

Theorem 1. *Let $g(x, y, z)$ be an integrable function, let $\gamma_0 = \mathbb{E}_{p_o(x,y,z)}[g(x, y, z)]$, and let I_0 be a self-normalized MC estimate for γ_0 calculated using $\hat{p}(\cdot)$ as per* (9)*. Assuming that $q(x, y, z)$ forms a valid importance sampling proposal distribution for $p_o(x, y, z)$, then*

$$\mathbb{E}\left[(I_0 - \gamma_0)^2\right] = \frac{\sigma^2}{N_0} + \frac{\delta^2}{N_1^2} + O(\epsilon) \tag{10}$$

where σ and δ are constants derived in the proof and, as before, $O(\epsilon)$ represents asymptotically dominated terms.

Proof. Though informally the high-level result follows directly from (Rainforth et al., 2018, Theorem 3), there are three subtleties that require further attention. Firstly, unlike (Rainforth et al., 2018, Theorem 3), this result is an asymptotic equality rather than a bound – in the limit of large N_0, N_1 it holds exactly. This more powerful result is made possible by knowing the exact form of the nonlinearity. Secondly, our overall estimator uses the ratio of two NMC estimators. Though Slutsky's Theorem means this does not create complications in the general demonstration of convergence, additional care is required when calculating the exact rate. Finally, samples are reused in both the inner and outer estimators. This could easily be avoided by sampling an additional z for the outer estimator, thereby giving an estimator trivially of the form considered by (Rainforth et al., 2018, Theorem 3). However, doing so would be less efficient and is expected to have a larger variance than the estimator used.

We start by considering the the partition function estimate, noting that true value is $Z = \iiint \pi_o(x, y, z)dxdydz$,

$$\hat{Z} = \frac{1}{N_0}\sum_{n=0}^{N_0} \frac{\frac{1}{N_1}\sum_{m=1}^{N_1} \frac{\psi(x_n, y_n, z_{n,m})\pi_i(y_n, z_{n,m})}{q(x_n, y_n, z_{n,m})}}{\frac{1}{N_1}\sum_{m=1}^{N_1} \frac{\pi_i(y_n, z_{n,m})}{q(z_{n,m}|y_n)}} \tag{21}$$

$$= \frac{1}{N_0}\sum_{n=0}^{N_0} \frac{\frac{1}{N_1}\sum_{m=1}^{N_1} v_{n,m}}{\frac{1}{N_1}\sum_{m=1}^{N_1} u_{n,m}} \tag{22}$$

where

$$u_{n,m} = \frac{\psi(x_n, y_n, z_{n,m})\pi_i(y_n, z_{n,m})}{q(x_n, y_n, z_{n,m})} \quad \text{and} \tag{23}$$

$$v_{n,m} = \frac{\pi_i(y_n, z_{n,m})}{q(z_{n,m}|y_n)} \tag{24}$$

will be used as shorthands. Further defining

$$\pi_i(y_n) = \int \pi_i(y_n, z)dz, \tag{25}$$

$$V_n = \frac{1}{N_1}\sum_{m=1}^{N_1} v_{n,m}, \text{ and } U_n = \frac{1}{N_1}\sum_{m=1}^{N_1} u_{n,m}, \tag{26}$$

and using Taylor's Theorem on $1/U_n$ about $\pi_i(y_n)$ gives

$$\hat{Z} = O(\epsilon) + \frac{1}{N_0}\sum_{n=0}^{N_0} \frac{V_n}{\pi_i(y_n)} \times \left(1 + \frac{\pi_i(y_n) - U_n}{\pi_i(y_n)} + \left(\frac{\pi_i(y_n) - U_n}{\pi_i(y_n)}\right)^2\right) \tag{27}$$

provided each $U_n, \pi(y_n) \neq 0$ to avoid singularity issues. We have by assumption that $\pi(y_n) \neq 0$ for all possible y_n as otherwise the problem becomes ill-defined. On the other hand, if $U_n = 0$, it must also be the case that $V_n = 0$. Here by taking the convention $V_n/U_n = 0$ when $U_n = V_n = 0$, we can avoid all further possible singularity issues, such that (27) always holds.

Meanwhile, the standard breakdown of the mean squared error to the variance plus the bias squared gives

$$\mathbb{E}\left[\left(\hat{Z} - Z\right)^2\right] = \mathrm{Var}\left[\hat{Z}\right] + \left(\mathbb{E}\left[\hat{Z} - Z\right]\right)^2.$$

Using (27), we see that the first term in the expansion dominates for the variance (as $\pi_i(y_n) - U_n$ decreases with N_1), such that the weak law of large numbers gives

$$\mathrm{Var}\left[\hat{Z}\right] = \frac{1}{N_0}\mathrm{Var}\left[\frac{V_1}{\pi_i(y_1)}\right] + O(\epsilon).$$

Now we have

$$V_1 = \mathbb{E}[v_{1,1}|x_1, y_1] + \frac{1}{N_1}\sum_{m=1}^{N_1}\left(v_{1,m} - \mathbb{E}[v_{1,m}|x_1, y_1]\right)$$

and we further see from the weak law of large numbers that the second term tends to 0 as N_1 increases, but the first term remains fixed. Thus the first term is dominant and we have

$$\mathrm{Var}\left[\hat{Z}\right] = \frac{1}{N_0}\mathrm{Var}\left[\frac{\mathbb{E}[v_{1,1}|x_1, y_1]}{\pi_i(y_1)}\right] + O(\epsilon) \tag{28}$$

$$= \frac{1}{N_0}\mathrm{Var}\left[\frac{\int \psi(x_1, y_1, z)\pi_i(y_1, z)dz}{q(x_1, y_1)\pi_i(y_1)}\right] + O(\epsilon) \tag{29}$$

$$= \frac{\sigma_z^2}{N_0} + O(\epsilon) \tag{30}$$

where

$$\sigma_z^2 = \mathrm{Var}\left[\frac{\int \pi_o(x_1, y_1, z)dz}{q(x_1, y_1)}\right]. \tag{31}$$

Switching focus to the bias we have

$$\begin{aligned}\mathbb{E}\left[\hat{Z} - Z\right] &= O(\epsilon) + \mathbb{E}\Bigg[\left(\frac{V_1}{\pi_i(y_1)}\right) \\ &\quad \times \left(\frac{\pi_i(y_1) - U_1}{\pi_i(y_1)} + \left(\frac{\pi_i(y_1) - U_1}{\pi_i(y_1)}\right)^2\right)\Bigg] \\ &= O(\epsilon) + \mathbb{E}\Bigg[\mathbb{E}\Bigg[\left(\frac{v_{1,1}}{\pi_i(y_1)}\right) \\ &\quad \times \left(\frac{\pi_i(y_1) - U_1}{\pi_i(y_1)} + \left(\frac{\pi_i(y_1) - U_1}{\pi_i(y_1)}\right)^2\right)\Bigg|y_1\Bigg]\Bigg].\end{aligned}$$

For the first order term in the expansion, only the component with respect to $u_{1,1}$ is non-zero as, for $m \neq 1$,

$$\begin{aligned}&\mathbb{E}[v_{1,1}\left(\pi_i(y_1) - u_{1,m}\right)|y_1] = \\ &\quad \mathbb{E}[v_{1,1}|y_1]\mathbb{E}[\left(\pi_i(y_1) - u_{1,m}\right)|y_1] = 0.\end{aligned} \tag{32}$$

Denoting the first order term as T_1, we thus have

$$\begin{aligned}T_1 &= \mathbb{E}\left[\frac{v_{1,1}\left(\frac{1}{N_1}\sum_{m=1}^{N_1}\pi_i(y_1) - u_{1,m}\right)}{\left(\pi_i(y_1)\right)^2}\right] \\ &= \frac{1}{N_1}\left(\mathbb{E}\left[\frac{v_{1,1}}{\pi_i(y_1)}\right] - \mathbb{E}\left[\frac{v_{1,1}u_{1,1}}{\left(\pi_i(y_1)\right)^2}\right]\right) \\ &= \frac{1}{N_1}\left(Z - \iiint \frac{\psi(x, y, z)\left(\pi_i(y, z)\right)^2}{q(z|y)\left(\int \pi_i(y, z')dz'\right)^2}dxdydz\right) \\ &= \frac{1}{N_1}\left(Z - \iiint \frac{\pi_o(x, y, z)p_i(z|y)}{q(z|y)}dxdydz\right).\end{aligned}$$

For the second order term, T_2, components of $u_{1,m}$ for $m \neq 1$ are no longer zero as follows

$$T_2 = \mathbb{E}\left[\mathbb{E}\left[\frac{v_{1,1}}{\pi_i(y_1)}\left(\frac{1}{N_1}\sum_{m=1}^{N_1}\frac{\pi_i(y_1) - u_{1,m}}{\pi_i(y_1)}\right)^2\middle|y_1\right]\right]$$

$$\begin{aligned}&= \frac{1}{N_1^2}\mathbb{E}\left[\frac{v_{1,1}\left(\pi_i(y_1) - u_{1,1}\right)^2}{\left(\pi_i(y_1)\right)^3}\right] + \frac{1}{N_1^2}\mathbb{E}\Bigg[\frac{1}{\left(\pi_i(y_1)\right)^3}\times \\ &\mathbb{E}\left[v_{1,1}\sum_{m=2}^{N_1}\sum_{\ell=1}^{N_1}\left(\pi_i(y_1) - u_{1,m}\right)\left(\pi_i(y_1) - u_{1,m}\right)\middle|y_1\right]\Bigg],\end{aligned}$$

now using an argument akin to (32) shows that terms for which $m \neq \ell$ are all zero. Further noticing that the first term is asympotitically dominated gives

$$\begin{aligned}&= O(\epsilon) + \\ &\frac{1}{N_1^2}\mathbb{E}\left[\frac{1}{\left(\pi_i(y_1)\right)^3}\mathbb{E}\left[v_{1,1}\sum_{m=2}^{N_1}\left(\pi_i(y_1) - u_{1,m}\right)^2\middle|y_1\right]\right], \\ &= O(\epsilon) + \left(\frac{N_1 - 1}{N_1^2}\right)\times \\ &\mathbb{E}\left[\mathbb{E}\left[\frac{v_{1,1}}{\pi_i(y_1)}\middle|y_1\right]\mathbb{E}\left[\left(\frac{\pi_i(y_1) - u_{1,1}}{\pi_i(y_1)}\right)^2\middle|y_1\right]\right], \\ &= O(\epsilon) + \\ &\frac{1}{N_1}\mathbb{E}\left[\frac{\int \pi_o(x, y_1, z)dxdz}{q(y_1)}\mathrm{Var}\left[\frac{u_{1,1}}{\pi_i(y_1)}\middle|y_1\right]\right], \\ &= O(\epsilon) + \\ &\frac{1}{N_1}\mathbb{E}\left[\frac{\iint \pi_o(x, y_1, z)dxdz}{q(y_1)}\mathrm{Var}\left[\frac{p_i(z_{1,1}|y_1)}{q(z_{1,1}|y_1)}\middle|y_1\right]\right],\end{aligned}$$

Putting the bias terms together now gives

$$\mathbb{E}\left[\hat{Z} - Z\right] = \frac{\delta_z}{N_1} + O(\epsilon) \tag{33}$$

where

$$\begin{aligned}\delta_z = &\iiint \pi_o(x, y, z)\mathrm{Var}\left[\frac{p_i(z_{1,1}|y_1)}{q(z_{1,1}|y_1)}\middle|y_1 = y\right]dxdydz \\ &+ Z - \iiint \frac{\pi_o(x, y, z)p_i(z|y)}{q(z|y)}dxdydz.\end{aligned} \tag{34}$$

We thus have that the mean squared error is

$$\mathbb{E}\left[\left(\hat{Z} - Z\right)^2\right] = \frac{\sigma_z^2}{N_0} + \frac{\delta_z^2}{N_1^2} + O(\epsilon) \tag{35}$$

where σ_z^2 and δ_z a respectively defined in (31) and (34).

If we now consider the estimator for the unnormalized target expectation (i.e. the numerator in the self-normalized estimator), we see that we can use the same arguments with $\psi(x, y, z)$ replaced by $\psi(x, y, z)g(x, y, z)$. Thus denoting this estimator as $\hat{G}$ and its true value as $G = \gamma_0 Z$, we have

$$\mathbb{E}\left[\left(\hat{G} - G\right)^2\right] = \frac{\sigma_g^2}{N_0} + \frac{\delta_g^2}{N_1^2} + O(\epsilon) \tag{36}$$

where

$$\sigma_g = \mathrm{Var}\left[\frac{\int g(x_1, y_1, z)\pi_o(x_1, y_1, z)dz}{q(x_1, y_1)}\right] \tag{37}$$

$$\delta_g = G - \iiint \frac{g(x,y,z)\pi_o(x,y,z)p_i(z|y)}{q(z|y)}dxdydz + \iiint \pi_o(x,y,z)\text{Var}\left[\frac{p_i(z_{1,1}|y_1)}{q(z_{1,1}|y_1)}\middle| y_1 = y\right]dxdydz. \tag{38}$$

Now the self-normalized estimator we actually use is $I_0 = \hat{G}/\hat{Z}$. To assess this, we represent

$$\hat{G} = G + \frac{\delta_g}{N_1} + \frac{\sigma_g \xi_1}{\sqrt{N_0}} + O(\epsilon) \tag{39}$$

$$\hat{Z} = Z + \frac{\delta_z}{N_1} + \frac{\sigma_z \xi_2}{\sqrt{N_0}} + O(\epsilon) \tag{40}$$

where ξ_1 and ξ_2 are correlated random variables, each with mean zero and variance 1 under their marginal distributions. Now again using Taylor's theorem

$$\begin{aligned}\frac{1}{\hat{Z}} &= \frac{1}{Z}\left(1 + \frac{Z - \hat{Z}}{Z}\right) + O(\epsilon) \\ &= \frac{1}{Z}\left(1 - \frac{1}{Z}\left(\frac{\delta_z}{N_1} + \frac{\sigma_z \xi_2}{\sqrt{N_1}}\right)\right) + O(\epsilon)\end{aligned} \tag{41}$$

where singularity issues are again dealt with because $Z \neq 0$ by our assumptions, noting $\hat{G} = 0$ if $\hat{Z} = 0$, and taking the convention $\hat{G}/\hat{Z} = 0$ whenever $\hat{G} = 0$. Thus

$$\begin{aligned}I_0 &= \frac{1}{Z^2}\left(G + \frac{\delta_g}{N_1} + \frac{\sigma_g \xi_1}{\sqrt{N_0}}\right)\left(Z - \frac{\delta_z}{N_1} - \frac{\sigma_z \xi_2}{\sqrt{N_0}}\right) + O(\epsilon) \\ &= \gamma_0 + \frac{\delta_g - \gamma_0 \delta_z}{Z N_1} + \frac{\sigma_g \xi_1 - \gamma_0 \sigma_z \xi_2}{Z\sqrt{N_0}} - \frac{\sigma_g \xi_1 \sigma_z \xi_2}{Z^2 N_0} + O(\epsilon)\end{aligned}$$

Therefore,

$$\text{Var}\left[I_0\right] = \frac{\sigma_g^2 + \gamma_0^2 \sigma_z^2 - 2\sigma_g \gamma_0 \sigma_z \text{Cov}(\xi_1, \xi_2)}{Z^2 N_0} + O(\epsilon) \tag{42}$$

and

$$\mathbb{E}\left[I_0 - \gamma_0\right] = \frac{\delta_g - \gamma_0 \delta_z}{Z N_1} - \frac{\sigma_g \sigma_z}{Z^2 N_0}\mathbb{E}\left[\xi_1 \xi_2\right] + O(\epsilon) \tag{43}$$

and therefore

$$\mathbb{E}\left[\left(I_0 - \gamma_0\right)^2\right] = \frac{\sigma^2}{N_0} + \frac{\delta^2}{N_1^2} + O(\epsilon) \tag{44}$$

where

$$\sigma^2 = \frac{\sigma_g^2 + \gamma_0^2 \sigma_z^2 - 2\sigma_g \gamma_0 \sigma_z \text{Cov}(\xi_1, \xi_2)}{Z^2} \tag{45}$$

$$\text{and} \quad \delta = \frac{\delta_g - \gamma_0 \delta_z}{Z}. \tag{46}$$

A full characterization of $\text{Cov}(\xi_1, \xi_2)$ can further be calculated by considering the full expansions for $\hat{G}$ and $\hat{Z}$. Though we do not trawl through the necessary algebra here, we note that $\text{Corr}(\xi_1, \xi_2) = 1$ if $g(x, y, z)$ is constant, in which case we also have $\delta_g = \gamma_0 \delta$ and $\sigma_g^2 = \gamma_0^2 \sigma_z^2$ and so $\delta = \sigma^2 = 0$. This is to be expected as, in this scenario, we have the trivial estimator $I_0 = \gamma_0 = g(x, y, z)\, \forall x, y, z$. □

Corollary 2. *The un-Rao-Blackwellized form of the estimator given in* (11)*, whereby only a single sample is returned from the inner query sampled in proportion to its weight, also converges. Specifically, it has the same rate of convergence for the bias, but has a constant factor increase in the variance.*

Proof. The un-Rao-Blackwellized estimator for the partition function can be represented as

$$\hat{Z}' = \frac{1}{N_0}\sum_{n=0}^{N_0} \frac{\psi(x_n, y_n, z_{n,m^*(n)})}{q(x_n, y_n)} \tag{47}$$

where

$$m^*(n) \sim \text{DISCRETE}\left(\frac{\pi_i(y_n, z_{n,m})/q(z_{n,m}|y_n)}{\sum_{\ell=1}^{N_1}\pi_i(y_n, z_{n,\ell})/q(z_{n,\ell}|y_n)}\right)$$

We first show $\hat{Z}$ from (21) is a true Rao-Blackwellization of $\hat{Z}'$ by noting that

$$\begin{aligned}&\mathbb{E}\left[\hat{Z}'\middle| x_{1:N_0}, y_{1:N_0}, z_{1:N_0,1:N_1}\right] \\ &= \frac{1}{N_0}\sum_{n=0}^{N_0}\sum_{m=1}^{N_1} \frac{\frac{\psi(x_n, y_n, z_{n,m})}{q(x_n, y_n)}\frac{\pi_i(y_n, z_{n,m})}{q(z_{n,m}|y_n)}}{\sum_{m=1}^{N_1}\frac{\pi_i(y_n, z_{n,m})}{q(z_{n,m}|y_n)}} = \hat{Z}.\end{aligned}$$

We thus see that $\hat{Z}'$ and $\hat{Z}$ have the same expectation as required. Equivalent arguments can further be applied to show the unnormalized target estimate has the same expectation as before.

For the variance, we can consider that

$$\begin{aligned}\text{Var}\left[\hat{Z}'\right] &= \frac{1}{N_0}\text{Var}\left[\frac{\psi(x_1, y_1, z_{1,m^*(n)})}{q(x_1, y_1)}\right] \\ &= \frac{1}{N_0}\text{Var}\left[\frac{\psi(x_1, y_1, z^*)}{q(x_1, y_1)} - \frac{\psi(x_1, y_1, z^*) - \psi(x_1, y_1, z_{1,m^*(n)})}{q(x_1, y_1)}\right]\end{aligned}$$

where $z^* \sim p_i(z|y)$. Now as N_1 increases, the second of these terms will diminish while the first does not, meaning the first is dominant.

By following the same steps as Theorem 1, we thus achieve the same result for the convergence rate except for substituting in for the following definitions

$$\sigma_z^2 = \text{Var}\left[\frac{\psi(x_1, y_1, z^*)}{q(x_1, y_1)}\right] \tag{48}$$

$$\sigma_g^2 = \text{Var}\left[\frac{g(x_1, y_1, z^*)\psi(x_1, y_1, z^*)}{q(x_1, y_1)}\right]. \tag{49}$$

Note that these variances are always larger than those from Theorem 1. □

Theorem 2. *If each* $\tau_k(n_0) \geq A\left(\log(n_0)\right)^\alpha, \forall n_0 > B$

for some constants $A, B, \alpha > 0$ and each f_k is continuously differentiable, then the mean squared error of J_0 as an estimator for γ_0 converges to zero as $N_0 \to \infty$.

Proof. Let $\hat{f}_{n_o} := f_0\left(y_{n_0}^{(0)}, I_1\left(y_{n_0}^{(0)}, \tau_{1:D}(n_0)\right)\right)$ and let $I_0(n_0)$ be a NMC estimator that uses $\tau_k(n_0)$ samples at each layer. We have

$$\mathbb{E}\left[(J_0-\gamma_0)^2\right] = \text{Var}\left[J_0\right] + \left(\mathbb{E}\left[J_0-\gamma_0\right]\right)^2$$

$$= \frac{1}{N_0^2}\sum_{n_0=1}^{N_0}\text{Var}\left[\hat{f}_{n_o}\right] + \left(\frac{1}{N_0}\sum_{n_0=1}^{N_0}\mathbb{E}\left[\hat{f}_{n_o}-\gamma_0\right]\right)^2$$

$$= \frac{1}{N_0^2}\sum_{n_0=1}^{N_0} n_0 \text{Var}\left[I_0(n_0)\right] + \left(\frac{1}{N_0}\sum_{n_0=1}^{N_0}\mathbb{E}\left[I_0(n_0)-\gamma_0\right]\right)^2$$

Substituting in for the variance and bias terms from (3) now gives

$$\mathbb{E}\left[(J_0-\gamma_0)^2\right] \leq O(\epsilon) + \frac{\varsigma_0^2}{N_0} + \left(\frac{1}{N_0}\sum_{n_0=1}^{N_0}\left(\frac{C_0\varsigma_1^2}{2\tau_1(n_0)} + \sum_{k=0}^{D-2}\left(\prod_{d=0}^{k}K_d\right)\frac{C_{k+1}\varsigma_{k+2}^2}{2\tau_{k+2}(n_0)}\right)\right)^2 \tag{50}$$

Here ς_0^2/N_0 clearly tends to zero as $N_0 \to \infty$. For the bias squared term, which we denote $S(N_0)^2$, we use the assumption that $\tau_k(n_0) \geq A\left(\log(n_0)\right)^\alpha, \forall n_0 > B$. In the following analysis, we will assume that $\alpha < 2$, noting that if the result of the overall theorem holds for α_1, then it trivially holds for $\alpha_2 > \alpha_1$. We now have

$$S(N_0)^2 \leq \left(\frac{\lfloor B\rfloor}{N_0}S(\lfloor B\rfloor) + \frac{1}{N_0}\sum_{n_0=\lceil B\rceil}^{N_0}\frac{\beta}{A\left(\log(n_0)\right)^\alpha}\right)^2$$

$$\leq 2\left(\frac{\lfloor B\rfloor S(\lfloor B\rfloor)}{N_0}\right)^2 + 2\left(\frac{1}{N_0}\sum_{n_0=\lceil B\rceil}^{N_0}\frac{\beta}{A\left(\log(n_0)\right)^\alpha}\right)^2$$

$$\leq 2\left(\frac{\lfloor B\rfloor S(\lfloor B\rfloor)}{N_0}\right)^2 + \frac{2\beta^2}{A^2N_0^\alpha}\left(\sum_{n_0=\lceil B\rceil}^{N_0}\frac{1}{\left(\log(n_0)\right)^2}\right)^\alpha$$

where β is as per (19). Here the first term clearly goes to zero because the assumption $\tau_k(n_0) \in \mathbb{N}^+$ ensures $\lfloor B\rfloor S(\lfloor B\rfloor)$ is a finite constant. For the second term, we first note from using a condensation test that

$$\sum_{n_0=\lceil B\rceil}^{N_0}\frac{1}{n_0\left(\log(n_0)\right)^2} < \infty. \tag{51}$$

Now by invoking Kronecker's lemma, namely that $\lim_{N\to\infty}\frac{1}{N}\sum_n^N X_n = 0$ if $\sum_{n=1}^\infty X_n/n < \infty$, it follows that this term tends to zero. Note that because we are examining the bound itself, rather than any random variables, this is a result which holds surely. We have thus shown that all non-dominated terms in (50) tend to zero as $N_0 \to \infty$, giving the required result. □

Theorem 3. *If each $\tau_k(n_0) \geq An_0^\alpha$, $\forall n_0 > B$ for some constants $A, B, \alpha > 0$ and each f_k is continuously differentiable, then*

$$\mathbb{E}\left[(J_0-\gamma_0)^2\right] \leq \frac{\varsigma_0^2}{N_0} + \left(\frac{\beta g(\alpha, N_0)}{AN_0^\alpha}\right)^2 + O(\epsilon), \tag{17}$$

$$\textit{where}\quad g(\alpha, N_0) = \begin{cases} 1/(1-\alpha), & \alpha < 1 \\ \log(N_0)+\eta, & \alpha = 1 \\ \zeta(\alpha)N_0^{\alpha-1}, & \alpha > 1 \end{cases}; \tag{18}$$

$$\beta = \frac{C_0\varsigma_1^2}{2} + \sum_{k=0}^{D-2}\left(\prod_{d=0}^{k}K_d\right)\frac{C_{k+1}\varsigma_{k+2}^2}{2}; \tag{19}$$

$\eta \approx 0.577$ is the Euler–Mascheroni constant; ζ is the Riemann–zeta function; and C_k, K_k, and ς_k are constants defined as per the corresponding NMC bound given in (3).

Proof. Starting at (50) and following on in the same manner as the proof for Theorem 2, we have

$$S(N_0)^2 \leq \left(\frac{\lfloor B\rfloor}{N_0}S(\lfloor B\rfloor) + \frac{1}{N_0}\sum_{n_0=\lceil B\rceil}^{N_0}\frac{\beta}{An_0^\alpha}\right)^2 = \left(\frac{\beta H_\alpha[N_0]}{AN_0}\right)^2 + O(\epsilon)$$

where $H_\alpha[N_0] := \sum_{n_0}^{N_0} n_0^{-\alpha}$ is the N_0-th generalized harmonic number of order α. For $\alpha = 1$ and $\alpha > 1$, it is well known that $H_1[N_0] \to \log(N_0)+\eta$ and $H_\alpha[N_0] \to \zeta(\alpha)$ respectively. For, $\alpha < 1$, we apply the Euler-Maclaurin formula giving

$$H_\alpha[N_0] = 1 + \int_{n_0=1}^{n_0=N_0} n_0^{-\alpha}dn_0 + \frac{N_0^{-\alpha}-1}{2} + R_1 \to N_0^{1-\alpha}/(1-\alpha).$$

where the dominant term originates from the integral. Putting everything back together, namely substituting in turn for the bound on $S(N_0)^2$ and then this bound into (50), now yields the desired result. □

Corollary 1. *Let J_0 be an ONMC estimator setup as per Theorem 3 with N_0 outermost samples and let I_0 be an NMC estimator with a matched overall sample budget. Defining $c = (1+\alpha D)^{(-1/(1+\alpha D))}$, then*

$$\text{Var}[J_0] \to c\text{Var}[I_0] \quad \textit{as} \quad N_0 \to \infty.$$

Further, if the NMC bias decreases at a rate proportional to that implied by the bound given in (3)*, namely*

$$|\mathbb{E}[I_0-\gamma_0]| = \frac{b}{M_0^\alpha} + O(\epsilon) \tag{20}$$

for some constant $b > 0$, where M_0 is the number of outermost samples used by the NMC sampler, then

$$|\mathbb{E}[J_0-\gamma_0]| \leq c^\alpha g(\alpha, N_0)\,|\mathbb{E}[I_0-\gamma_0]| + O(\epsilon).$$

Proof. We first consider how to match the sample budgets between the two estimators. Noting that asymptotically, the computational cost is dominated by calculations for

the innermost estimator (see (Rainforth, 2017, Appendix G)), we have for large N_0,

$$\text{Cost}_{\text{ONMC}} \rightarrow \sum_{n_0}^{N_0} \prod_{k=1}^{D} \tau_k(n_0) = A^D \sum_{n_0}^{N_0} n_0^{\alpha D} = A^D H_{-\alpha D}[N_0].$$

The respective asymptotic cost for an NMC using M_0 outermost samples is

$$\text{Cost}_{\text{NMC}} \rightarrow A^D M_0^{1+\alpha D}.$$

Thus matching the computational budgets gives

$$M_0 = \left(H_{-\alpha D}[N_0]\right)^{\frac{1}{1+\alpha D}}. \tag{52}$$

Now by applying the Euler-Maclaurin formula to $H_{-\alpha D}[N_0]$ in similar manner to Theorem 3, we get

$$H_{-\alpha D}[N_0] \rightarrow \frac{N_0^{1+\alpha D}}{1+\alpha D}, \quad \text{and thus} \quad M_0 \rightarrow cN_0.$$

Using (3) and Theorem 3 we thus have

$$\text{Var}\left[J_0\right] \rightarrow \varsigma_0^2/N_0 \rightarrow c\varsigma_0^2/M_0 \rightarrow c\text{Var}\left[I_0\right].$$

Now considering the biases,

$$\begin{aligned}\left|\mathbb{E}[J_0 - \gamma_0]\right| &= \left|\frac{1}{N_0}\sum_{n_0=1}^{N_0} \mathbb{E}\left[I_0(n_0) - \gamma_0\right]\right| \\ &\leq \frac{1}{N_0}\sum_{n_0=1}^{N_0} \left|\mathbb{E}\left[I_0(n_0) - \gamma_0\right]\right|\end{aligned}$$

and whenever (20) holds,

$$\begin{aligned}&= \frac{1}{N_0}\sum_{n_0=1}^{N_0} \frac{b}{n_0^\alpha} + O(\epsilon) \\ &= \frac{bH_\alpha[N_0]}{N_0} + O(\epsilon) \\ &= \frac{bg(\alpha, N_0)}{N_0^\alpha} + O(\epsilon).\end{aligned}$$

By comparison, (20) also gives us

$$\left|\mathbb{E}[I_0 - \gamma_0]\right| = \frac{b}{c^\alpha N_0^\alpha} + O(\epsilon)$$

and so

$$\left|\mathbb{E}[J_0 - \gamma_0]\right| \leq c^\alpha g(\alpha, N_0)\left|\mathbb{E}[I_0 - \gamma_0]\right| + O(\epsilon)$$

as required. □

B OBSERVING THE OUTPUT OF A NESTED QUERY

As discussed in Section 3, one can construct nested inference problems where one observes the output of, rather than sampling from, the nested query. For example, we could think about adjusting our previous example to the following

```
(defquery inner [y D]
 (let [z (sample (gamma y 1))]
  (observe (normal y z) D)
  z))
```

```
(defquery outer [D]
 (let [y (sample (beta 2 3))
       x (sample (gamma 1 1))
       dist (conditional inner)]
  (observe (dist y D) x)
  (* y x)))
```

Statistically, this problem is still well defined and can be represented in the same form as (7); Anglican's **sample** and **observe** have the same impact on the distribution defined by a program, varying only in whether the variable already exists or not (Rainforth et al., 2016b).

However, in general we are not able to evaluate even the unnormalized density of a program's outputs due to change-of-variables complications (Rainforth, 2017, Chapter 4). This creates an ABC-style problem (Csilléry et al., 2010), wherein we can generate weighted samples from the inner query, but we cannot evaluate its density for a given output. This creates a substantial computational issue for actually observing a nested query that must be dealt with on top of any complications from the nested estimation. Dealing with these is beyond the scope of this paper and is left to future work.

C DISCRETE OR DETERMINISTIC INPUT VARIABLES

One special case where consistency can be maintained without requiring infinite computation for each nested call is when the variables passed to the inner query can only take on, say C, finite possible values. Of particular note, is the case when only deterministic variables are passed to the inner query, corresponding to $C = 1$, which, for example, forms the theoretical basis for the "programs as proposals" approach of Cusumano-Towner and Mansinghka (2018). As per Theorem 5 of Rainforth et al. (2018), we can rearrange such problems to C separate estimators such that the standard Monte Carlo error rate can be achieved. This is perhaps easiest to see by noting that for such problems, $\int \pi_i(y, z)dz$ can only on C distinct values, leading to a separate, non nested, inference problem through enumeration. For repeated nesting, the rearrangement can be recursively applied until one achieves a complete set of non-nested estimators. To avoid inferior NMC convergence rates, this special case requires explicit rearrangement or a specialist consideration by the language back-end (as done by e.g. Stuhlmüller and Goodman (2012, 2014); Cornish et al. (2017)). For example, one can dynamically catch the inner query being called with the same inputs, e.g. using memoization, and then exploit the fact that all such cases target the same inference problem. Care is required in these approaches to ensure the correct combination with outer query, e.g. returning properly weighted samples and ensuring the budget of the inner queries remains fixed.

D EXACT SAMPLING

It may, in fact, be possible to provide consistent estimates for many nested query problems without requiring infinite computation for each nested call by using exact sampling methods such as rejection sampling or coupled Markov chains (Propp and Wilson, 1996). Such an approach is taken by Church (Goodman et al., 2008), wherein no sample ever returns until it passes its local acceptance criterion as a hierarchical rejection sampler. Church is able to do this because it only supports hard conditioning on events with finite probability, allowing it to take a guess-and-check process that produces an exact sample in finite time, simply sampling from the generative model until the condition is satisfied. Although the performance still clearly gets exponentially worse with nesting depth, this is a change in the constant factor of the computation, not its scaling with the number of samples taken: generating a single exact sample of the distribution has a finite expected time using rejection sampling which is thus a constant factor in the convergence rate.

Unfortunately, most problems require conditioning on measure zero events because they include continuous data – they require a soft conditioning akin to the inclusion of a likelihood term – and so cannot be tackled using Church. Constructing a practical generic exact sampler for soft conditioning in an automated way is likely to be insurmountably problematic in practice. Nonetheless, it does open up the intriguing prospect of a hypothetical system that provides a standard Monte Carlo convergence rate for nested inference. This assertion is a somewhat startling result: it suggests that Monte Carlo estimates made using nested exact sampling methods have a fundamentally different convergence rate for nested inference problems (though not nested estimation problems in general) than, say, nested self-normalized importance sampling.

E CASE STUDY: SIMULATING A POKER PLAYER

As a more realistic demonstration of the utility for allowing nested inference in probabilistic programs, we consider the example of simulating a poker player who reasons about another player; we will refer to the two players respectively as P1 and P2. Anglican code for this example is given in Figure 3. Though the model has been kept deliberately simple for exposition, one could easily envisage adapting it to a higher fidelity simulation. In particular, one could easily adapt the model to consider multiple players, additional betting options for the second player, and multiple rounds of betting (for which addition levels to the nesting might be required).

At a high level, we a trying to estimate the distribution of payoffs (i.e. the net money received) by P1 for different hands and bets. This can then in turn be used to, for example, optimize the bet made. The starting situation is that P1 is on the small blind ($\pounds 1$) and P2 on the big blind ($\pounds 2$), with no other players currently in the game. This means that P1 and P2 have already have committed (as required by the rules of the game) $\pounds 1$ and $\pounds 2$ respectively to the pot and it is P1's turn to act. P1 can now choose between three actions

Fold – P1 declines to commit any more money. P2 takes the pot giving P1 a payoff of $-\pounds 1$.

Call – P1 matches the stake from the big blind. For simplicity, we are ignoring further rounds of betting and the scenario where P2 makes a further bet. There will, therefore, be a showdown where the better hand takes the pot. Here P1's payoff is $+\pounds 2$ if they transpire to have the better hand and $-\pounds 2$ otherwise.

Bet – P1 increases their stake to between twice the big blind (i.e. $\pounds 4$) and the maximum allow bet size (which we take to be $\pounds 10$). P2 then themselves subsequently decides whether they will call this bet or fold. If they fold, P1 receives a payoff of $+\pounds 2$. If they call, a showdown occurs as before, except that the win/lose payoffs are now $\pm$ the size of P1's bet.

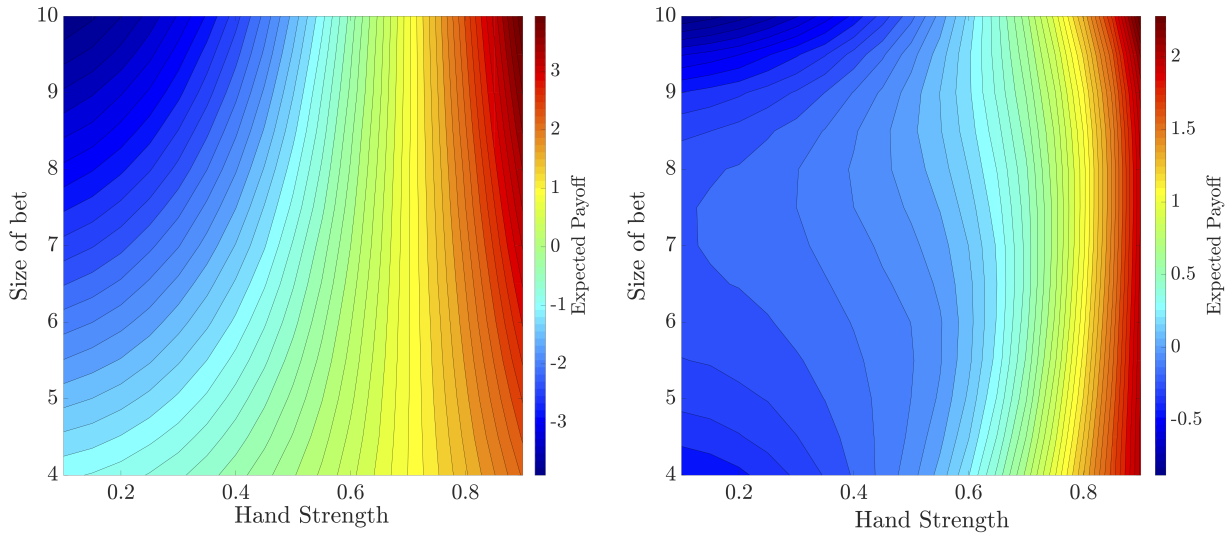
Estimating the payoff distributions for the cases where P1 folds or calls is straightforward. Folding always yields a payoff of $-\pounds 1$. Calling yields $+\pounds 2$ with probability equal to the probability that P1's hand is better than a randomly generated hand and $-\pounds 2$ otherwise. Thus if we represent hand strength as a uniform distribution between 0 and 1, the expected payoff of calling when P1 has hand strength h_1 becomes simply $2h_1 - 2(1-h_1) = 2(2h_1-1)$. Consequently, the expected payoff of calling is better than that of folding if and only if $h_1 > 0.25$

If P1 instead decides to bet, estimating the payoff distribution becomes substantially more complicated as it no longer depends only on the respective strength of the two hands, but also the action P2 takes. This action will be influenced not only by P2's hand, but also the size of P1's bet: P2 can draw inferences about likely hands for P1 using the information conveyed in P1's bet. To reflect this, our model for P2, `p2-sim`, uses a likelihood function for P1's betting, `p1-bet-dist`, to condition on the actual bet made. This likelihood is based on the, slightly naïve, sentiment that P1 will bet more with a better hand, while also allowing provision for P1 generating their bet at random as a bluff. P2 decides to call P1's bet if their hand is better than the hand they simulate for P1. Thus denoting c_2 as the boolean variable indicate if P2 calls then we have $P(\{c_2 = 1\}) = P(h_2 > h_1 | b_1) = \mathbb{E}[h_2 > h_1 | b_1]$ where b_1 represents P1's bet.[1] Note that, from P2's perspective, h_2 and b_1 are known, but h_1 is a random variable.

[1] In practice, it may be more realistic to assume that, rather

```
(defdist hand-strength []
 ;; Samples the strength of a hand
 [dist (uniform-continuous 0 1)]
 (sample* [this] (sample* dist))
 (observe* [this value] (observe* dist value)))

(defdist p1-bet-dist [hand]
 ;; Likelihood model used by player 2 to infer the strength of player
 ;; 1's hand
 [mean-bet (if (< hand 0.5) 0 (* 8 hand))]
 (sample* [this] nil) ;; No need to support sampling here
 (observe* [this value]
   (log-sum-exp
    (+ (log 0.95) (observe* (normal mean-bet 2) value))
    (+ (log 0.05) (observe* (uniform-continuous 4 10) value)))))

(with-primitive-procedures [hand-strength p1-bet-dist]
 (defm calc-payoff [p1-hand p1-bet p2-hand p2-call]
  ;; Calculate payoff given actions and hands.
  (let [small-blind 1
        big-blind 2]
   (case (< p1-bet big-blind)
     true (- small-blind) ;; Lose small blind if fold
     false (case p2-call
             false big-blind ;; Pick up big blinds
             true (if (> p2-hand p1-hand);; Showdown
                    (- p1-bet)
                    p1-bet)))))

 (defquery p2-sim [p2-hand p1-bet]
  ;; Simulator for player 2 who knows player 1's bet but not her
  ;; hand.  Returns boolean of whether bet is called
  (let [p1-hand (sample (hand-strength))] ;; Simulate a hand for player 1
   (observe (p1-bet-dist p1-hand) p1-bet) ;; Condition on player 1's known bet
   (> p2-hand p1-hand)))

 (defquery p1-payoff [p1-hand p1-bet N_1]
  ;; Estimator for distribution of player 1's payoff for given hand and action
  (let [p2-hand (sample (hand-strength)) ;; Sample hand for opponent
        dist (conditional p2-sim :smc :number-of-particles N_1)
        p2-call (sample (dist p2-hand p1-bet))] ;; Simulate player 2
   (calc-payoff p1-hand p1-bet p2-hand p2-call)))) ;; Return payoff

(defn estimate-payoff [p1-hand p1-bet N_0 N_1]
 ;; Estimates the relative probability of actions given a hand
 (let [samps (->> (doquery :importance p1-payoff [p1-hand p1-bet N_1])
                  (take N_0))]
  (empirical-distribution (collect-results samps))))
```

Figure 3: Code simulating the behavior of a poker player who reasons about the behavior of another player. Explanation provided in text.



(a) Expected payoff with $N_1 = 1$

(b) Expected payoff with $N_1 = 1500$

Figure 4: Contour plots for P1's expected payoff using the poker model given in Figure 3 as a function of their hand strength and amount bet (in £). On the left is the naïve estimator using $N_1 = 1$, which is an equivalent to ignoring the **observe** statement in `p2-sim`, such P2 bets when their hand is better than one drawn uniformly at random. On the right is the output of produced by using the empirical measured given in (9) based on self-normalized, nested importance sampling, with $N_1 = 1500$. For both models, an evenly spaced 17×13 grid (hand strength by bet size) of estimates was calculated using $N_0 = 2 \times 10^6$ outer samples, which was in turn converted to the shown contour plots using MATLAB's `contourf` plot function. Note the difference colorbar scaling between the plots.

To estimate the payoff when P1 bets, we nest this model for P2. Specifically, the payoff for P1 is given by $\mathbb{E}[\text{PAYOFF}(h_1, b_1, h_2, c_2)]$ where h_1 and b_1 are fixed, h_2 is drawn uniformly at random, and $c_2|h_2$ is sampled using a nested inference on `p2-sim`.

Keen-eyed readers may have noticed that the use of **conditional** in `p1-payoff` is distinct to elsewhere in the paper as we have explicitly used SMC inference with a provided number of particles N_1 for **conditional**. This provides a roundabout means of controlling the computational budget for calls to **conditional**, as we showed is required for convergence in Section 3.

Figure 4 shows contour plots for P1's expected payoff as a function of their hand strength and amount bet when P2 naïvely simulates P1's hand strength from the prior (left) and uses inference to try and infer P1's hand strength from their bet (right). As expected, for the naïve model then it is better for P1 to make larger bets when she has a strong hand and smaller bets when she has a weak hand. When she has a weak hand, the expected payoff of all possible bets is worse than folding or calling.

In our nested model, a number of more complex behaviors arise. Firstly, we note that the overall variation in expected payoff is less: making significant bets with a weak hand becomes less detrimental, while the expected rewards of a large bet with a strong hand are also diminished. This occurs because the act of betting portrays a stronger hand and so P2 is more likely to fold when they condition their assessment of P1's hand on the fact that P1 bet. Consequently, a bluff with a weak hand is more likely to steal the blinds, while a bet with a strong hand is less likely to get paid off by a call. In fact, we see that, for this model, it is beneficial to take a hyper-aggressive stance and always bet: P2 is sufficiently passive that the risk of betting is always worthwhile even for a very weak hand.

Another, more subtle, effect that transpires is that, when P1 has a weak hand, it is possible to both bet too much and bet too little. Too small a bet is more likely to get called – even when P2 has a weak hand, they are being offered very favorable odds to call the bet in hope that P1 is bluffing. Too large a bet exposes P1 to unnecessarily large losses when P2 transpires to have a strong hand and decides to call. A medium sized bluff thus offers the best balance between being believable and not being unnecessarily risky. A different effect is seen when P1 has a strong hand: small bets are likely to get paid-off by a large number of hands, while large bets may yield large

than aiming to call in proportion to $P(h_2 > h_1|b_1)$, P2 instead tries to directly estimate this probability and deterministically chooses to call if this estimate some threshold determined by the pot odds. This would then lead to an "estimates as variables" nested estimation, instead of a nested inference model.

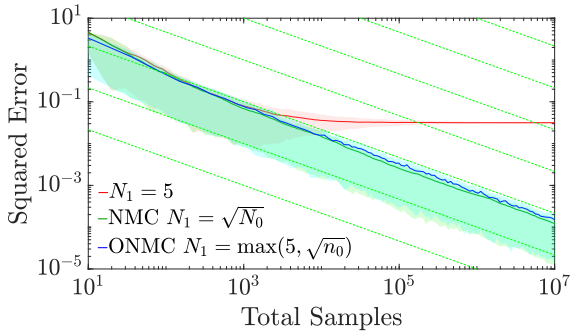


Figure 5: Convergence of ONMC, NMC, and fixed N_1 for expected payoff in poker example with hand strength set to 0.1 and bet size set to 6. Results are averaged over 1000 runs, with solid lines showing the mean and shading the 25-75% quantiles. Ground truth was estimated empirically using a large scale NMC run with $N_0 = 5 \times 10^7$ and $N_1 = 5000$. The theoretical rates for NMC are shown by the dashed lines.

rewards or potentially cause an even stronger hand to fold. Thus a mid-level bet actually becomes the worst option.

For this problem, nesting has allowed us to emulate a player that assumes simplistic play from their opponent to outsmart them. One could clearly envisage making the model even smarter by adding additional layers of nesting. Suppose that P2 is actually a good player that more explicitly reasons about the fact that P1 will be reasoning about them. We could then, for example, use the model developed so far for P2's simulation of P1, meaning that they will be more attuned to the fact that P1 might be bluffing. Amongst other things, this is then likely to make them more likely to call, knowing that P1 is playing an aggressive game and they have a good chance of catching a bluff. P1 could then in turn use this a higher fidelity model for P2, replacing the current `p-sim`. It is easy to see how such a meta-reasoning hierarchy could potentially lead to smarter and smarter players. However, the NMC converges rates tell us that doing so comes at a substantial cost in terms of the difficultly of solving the resulting nested estimation problem: the required number of samples increases exponentially with the depth of the nesting.

We finish by comparing the empirical performance of ONMC and NMC for this particular problem. Here we consider a fixed bet of £6 and hand strength of 0.1. The convergence, shown in Figure 5, demonstrates extremely similar performance for the two approaches, while again highlighting the danger of keeping N_1 fixed. Note that the slightly different setup used for τ to that used in the Gaussian example does not make any noticeable difference to the performance (not shown), with the different choice stemming from a desire to better highlight the problem of keeping N_1 fixed.

F SIMPLE ANALYTICAL MODEL DETAILS

We consider the following simple analytic model introduced by (Rainforth et al., 2018) for which the true nested expectation is $\gamma_0 = \frac{1}{2}\log\left(\frac{2}{5\pi}\right) - \frac{2}{15}$

$$y^{(0)} \sim \text{Uniform}(-1, 1), \tag{53a}$$

$$y^{(1)} \sim \mathcal{N}(0, 1), \tag{53b}$$

$$f_1(y^{(0)}, y^{(1)}) = \sqrt{\frac{2}{\pi}} \exp\left(-2(y^{(0)} - y^{(1)})^2\right), \tag{53c}$$

$$f_0(y^{(0)}, \gamma_1(y^{(0)})) = \log(\gamma_1(y^{(0)})). \tag{53d}$$

Results for this model are shown in Figure 2 in the main paper.

G EXPERIMENTAL DESIGN EXAMPLE

An example application of using estimates as first class variables if provided by Bayesian experimental design (Chaloner and Verdinelli, 1995). One can implicitly use expectation estimates as first class variables in Anglican by either calling **`doquery`** inside a **`defdist`** declaration or in a **`defn`** function passed to a query using **`with-primitive-procedures`**, a macro providing the appropriate wrappings to convert a Clojure function to an Anglican one. Anglican code using the latter approach to create generic estimator for Bayesian experimental design problems is shown in Figure 6, providing a consistent means of carrying out this class of nested estimation problems. (Rainforth et al., 2018, Figure 6) shows the convergence code equivalent to that of Figure 6 for a delay discounting model. This shows the convergence (or more specifically lack there of) in the case where $M = N_1$ is held fixed and the superior convergence achieved when exploiting the finite number of possible outputs to produce a reformulated, standard Monte Carlo, estimator. It therefore highlights both the importance of increasing the number of samples used by the inner query and exploiting our outlined special cases when possible.

```
(defm prior [] (normal 0 1))
(defm lik [theta d] (normal theta d))

(defquery inner-q [y d]
 (let [theta (sample (prior))]
  (observe (lik theta d) y)))

(defn inner-E [y d M]
 (->> (doquery :importance
        inner-q [y d])
      (take M)
      log-marginal))

(with-primitive-procedures [inner-E]
 (defquery outer-q [d M]
  (let [theta (sample (prior))
        y (sample (lik theta d))
        log-lik (observe*
                  (lik theta d) y)
        log-marg (inner-E y d M)]
   (- log-lik log-marg))))

(defn outer-E [d M N]
 (->> (doquery :importance
        outer-q [d M])
      (take N)
      collect-results
      empirical-mean))
```

Figure 6: Anglican code for Bayesian experimental design. By changing the definitions of `prior` and `lik`, this code can be used as a NMC estimator (consistent as $N, M \to \infty$) for any static Bayesian experimental design problem. Here **`observe*`** is a function for returning the log likelihood (it does not affect the trace probability), **`log-marginal`** produces a partition function estimate from a collection of weighted samples, and **`->>`** successively applies a series of functions calls, using the result of one as the last input the next. When `outer-E` is invoked, this runs importance sampling on `outer-q`, which, in addition to carrying out its own computation, calls `inner-E`. This, in turn, invokes another inference over `inner-q`, such that a MC estimate using `M` samples is constructed for each sample of `outer-q`. Thus `log-marg` is MC estimate itself. The final return is the (weighted) empirical mean for the outputs of `outer-q`.

Acknowledgements

I would like to thank Yee Whye Teh, N. Siddharth, and Benjamin Bloem-Reddy for feedback on drafts of this work. My 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. Some of the work was undertaken while I was at the Department of Engineering Science and was supported by a BP industrial grant.

References

C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725, 2009.

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

G. Casella and C. P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 1995.

R. Cornish, F. Wood, and H. Yang. Efficient exact inference in discrete Anglican programs. 2017.

K. Csilléry, M. G. Blum, O. E. Gaggiotti, and O. François. Approximate Bayesian Computation (ABC) in practice. *Trends in Ecology & Evolution*, 25(7):410–418, 2010.

M. F. Cusumano-Towner and V. K. Mansinghka. Using probabilistic programs as proposals. *arXiv preprint arXiv:1801.03612*, 2018.

G. Fort, E. Gobet, and E. Moulines. MCMC design-based non-parametric regression for rare-event. application to nested risk computations. *Monte Carlo Methods Appl*, 2017.

H. Ge, K. Xu, and Z. Ghahramani. Turing: a language for composable probabilistic inference. In *AISTATS*, 2018.

N. Goodman, V. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum. Church: a language for generative models. *UAI*, 2008.

N. D. Goodman and A. Stuhlmüller. *The Design and Implementation of Probabilistic Programming Languages*. 2014.

A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani. Probabilistic programming. In *Proceedings of the on Future of Software Engineering*. ACM, 2014.

R. Hickey. The Clojure programming language. In *Proceedings of the 2008 symposium on Dynamic languages*, page 1. ACM, 2008.

L. J. Hong and S. Juneja. Estimating the mean of a non-linear function of conditional expectation. In *Winter Simulation Conference*, 2009.

T. A. Le, A. G. Baydin, and F. Wood. Nested compiled inference for hierarchical reinforcement learning. In *NIPS Workshop on Bayesian Deep Learning*, 2016.

V. Mansinghka, D. Selsam, and Y. Perov. Venture: a higher-order probabilistic programming platform with programmable inference. *arXiv preprint*

arXiv:1404.0099, 2014.

T. Mantadelis and G. Janssens. Nesting probabilistic inference. *arXiv preprint arXiv:1112.3785*, 2011.

I. Murray, Z. Ghahramani, and D. J. MacKay. MCMC for doubly-intractable distributions. In *UAI*, 2006.

C. A. Naesseth, F. Lindsten, and T. B. Schön. Nested sequential Monte Carlo methods. In *ICML*, 2015.

L. Ouyang, M. H. Tessler, D. Ly, and N. Goodman. Practical optimal experiment design with probabilistic programs. *arXiv preprint arXiv:1608.05046*, 2016.

M. Plummer. Cuts in Bayesian graphical models. *Statistics and Computing*, 25(1):37–43, 2015.

J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random structures and Algorithms*, 9(1-2): 223–252, 1996.

T. Rainforth. *Automating Inference, Learning, and Design using Probabilistic Programming*. PhD thesis, 2017.

T. Rainforth, R. Cornish, H. Yang, and F. Wood. On the pitfalls of nested Monte Carlo. *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016a.

T. Rainforth, T. A. Le, J.-W. van de Meent, M. A. Osborne, and F. Wood. Bayesian optimization for probabilistic programs. In *NIPS*, pages 280–288, 2016b.

T. Rainforth, R. Cornish, H. Yang, A. Warrington, and F. Wood. On nesting Monte Carlo estimators. In *ICML*, 2018.

A. Scibior and Z. Ghahramani. Modular construction of Bayesian inference algorithms. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016.

D. Spiegelhalter, A. Thomas, N. Best, and W. Gilks. BUGS 0.5: Bayesian inference using Gibbs sampling manual (version ii). *MRC Biostatistics Unit, Cambridge*, 1996.

A. Stuhlmüller and N. D. Goodman. A dynamic programming algorithm for inference in recursive probabilistic programs. In *Second Statistical Relational AI workshop at UAI 2012 (StaRAI-12)*, 2012.

A. Stuhlmüller and N. D. Goodman. Reasoning about reasoning by nested conditioning: Modeling theory of mind with probabilistic programs. *Cognitive Systems Research*, 28:80–99, 2014.

D. Tolpin, J.-W. van de Meent, and F. Wood. Probabilistic programming in Anglican. Springer, 2015.

D. Tolpin, J.-W. van de Meent, H. Yang, and F. Wood. Design and implementation of probabilistic programming language Anglican. In *Proceedings of the 28th Symposium on the Implementation and Application of Functional Programming Languages*. ACM, 2016.

F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. In *AISTATS*, pages 2–46, 2014.

R. Zinkov and C.-C. Shan. Composing inference algorithms as program transformations. In *UAI*, 2017.