Nesting Probabilistic Programs

Tom Rainforth
Department of Statistics, University of Oxford

Overview

- We investigate the statistical implications of nesting probabilistic programs
- Nesting programs allows definition of models which could not otherwise be expressed, e.g. experimental design, reasoning about reasoning [1]
- Changes are required to existing systems to ensure consistent estimation
- We delineate possible nesting methods and assert their respective correctness using recent results from nested Monte Carlo estimation [2, 3]

Take Home

- Observing the output of another query (i.e. program) is statistically sound, though semantically challenging
- Sampling from the conditional distribution of one query inside another is semantically straightforward but statistically problematic:
  - Convergence is possible but requires additional assumptions/precautions
  - Number of samples used in **each** call of a nested query must $\to \infty$
  - Convergence rate is very slow: $O\left(T^{-\frac{2}{2+D}}\right)$ for budget T and depth D
- Using estimates as first class variables is also statistically problematic

Nested Monte Carlo (NMC)

- Standard Monte Carlo

$$\gamma = \mathbb{E}\left[f(y)\right] \approx I = \frac{1}{N}\sum_{n=1}^{N} f(y_n) \quad \text{where} \quad y_n \sim p(y). \tag{1}$$

- Nested Monte Carlo (taking depth $D = 2$ as an example)

$$\gamma_0 = \mathbb{E}\left[f_0\left(y^{(0)}, \mathbb{E}\left[f_1\left(y^{(0:1)}, \mathbb{E}\left[f_2\left(y^{(0:2)}\right)\middle|y^{(0:1)}\right]\right)\middle|y^{(0)}\right]\right)\right] \tag{2}$$

$$\approx I_0 = \frac{1}{N_0}\sum_{n_0=1}^{N_0} f_0\left(y^{(0)}_{n_0}, \frac{1}{N_1}\sum_{n_1=1}^{N_1} f_1\left(y^{(0:1)}_{n_1}, \frac{1}{N_2}\sum_{n_2=1}^{N_2} f_2\left(y^{(0:2)}_{n_2}\right)\right)\right) \tag{3}$$

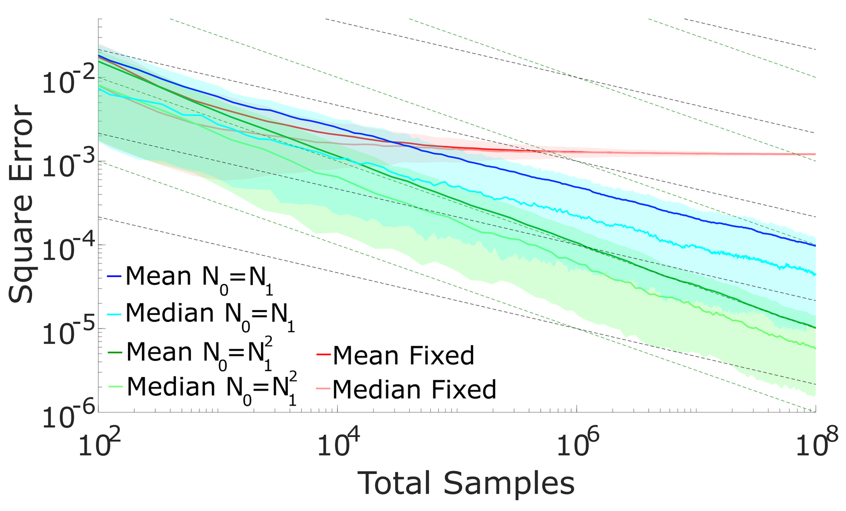
- Recently demonstrated general convergence rate for mean squared error [2]

$$\mathbb{E}\left[\left(I_0 - \gamma_0\right)^2\right] \le \frac{\varsigma_0^2}{N_0} + \left(\frac{C_0\varsigma_1^2}{2N_1} + \sum_{k=0}^{D-2}\left(\prod_{d=0}^{k} K_d\right)\frac{C_{k+1}\varsigma_{k+2}^2}{2N_{k+2}}\right)^2 + O(\epsilon). \tag{4}$$

where K_k, C_k, and ς_k are constants.

- Bound tightest when $N_0 \propto N_1^2 \propto \dots \propto N_k^2$ giving convergence rate $O\left(T^{-\frac{2}{2+D}}\right)$ where $T = N_0 N_1 \dots N_k \Rightarrow$ decreases exponentially with depth

Empirical Demonstration of Convergence of NMC



- Simple analytic model from [2]
- (Red) If $N_2 = 5$ is fixed then doesn't converge
- (Blue) Setting $N_0 \propto N_1 \propto N_2$ gives slow convergence
- (Green) Setting $N_0 \propto N_1^2 \propto N_2^2$ gives faster convergence

Special Cases

- Discrete expectations can be collapsed through enumeration
- Linear f_k can be collapsed, e.g. for linear f_1 and $D = 1$

$$\mathbb{E}\left[f_0\left(y^{(0)}, \mathbb{E}\left[f_1\left(y^{(0:1)}\right)\middle|y^{(0)}\right]\right)\right] = \mathbb{E}\left[f_0\left(y^{(0)}, f_1\left(y^{(0:1)}\right)\right)\right] \tag{5}$$

- Products of expectations can be combined to a single expectation over a distribution defined by the product of the marginal distributions
- These latter two demonstrate consistency of nested sampling when inputs to the inner query are deterministic and proper weighting is used

References

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

[2] Tom Rainforth, Robert Cornish, Hongseok Yang, Andrew Warrington, and Frank Wood. On the opportunities and pitfalls of nesting Monte Carlo estimators. *arXiv preprint arXiv:1709.06181*, 2017.

[3] Gersende Fort, Emmanuel Gobet, and Eric Moulines. MCMC design-based non-parametric regression for rare-event. application to nested risk computations. *Monte Carlo Methods Appl*, 2017.

[4] David Tolpin, Jan-Willem van de Meent, Hongseok Yang, and Frank Wood. Design and implementation of probabilistic programming language Anglican. In *Proceedings of the 28th Symposium on the Implementation and Application of Functional Programming Languages*, page 6. ACM, 2016.

Nested Sampling

Consider the following nested Anglican [4] model

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

```
(defquery outer [D]
 (let [y (sample (beta 2 3))
       z (inner y D)]
  (* y z)))
```

for which outer defines the distribution

$$\tilde{\pi}_1(z, y, D) = p(y)p(z|y)p(D|y, z) = \textsc{Beta}(y; 2, 3)\Gamma(z; y, 1)\mathcal{N}(D; z, y^2)$$

compared to the following nested query model

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

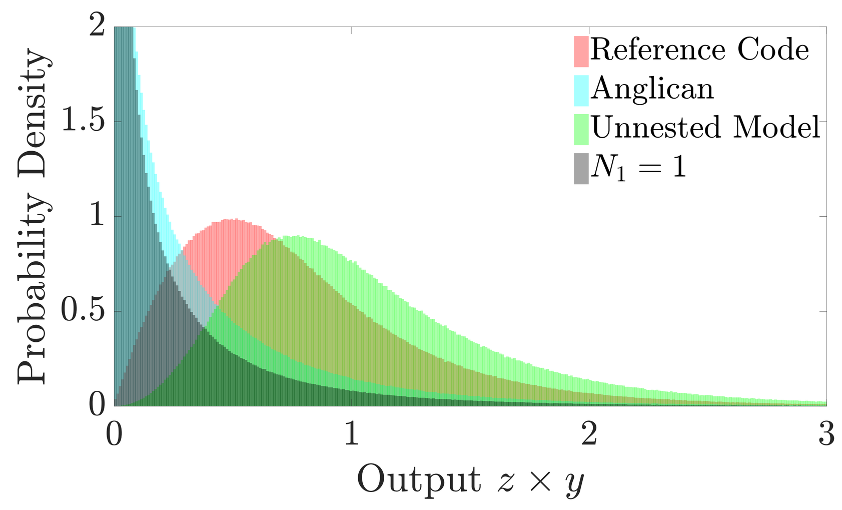
```
(defquery outer [D]
 (let [y (sample (beta 2 3))
       dist (conditional inner)
       z (sample (dist y D))]
  (* y z)))
```

where `(conditional inner)` is the conditional distribution of inner and we have

$$\tilde{\pi}_2(z, y, D) = p(y)p(z|y, D) = p(y)\frac{p(z|y)p(D|y, z)}{p(D|y)} \neq \tilde{\pi}_1(z, y, D).$$

The partial normalization constant $p(D|y)$ depends on y and so $\tilde{\pi}_2(z, y, D)$ is doubly intractable – we cannot evaluate our unnormalized target distribution exactly

How does Anglican do? Not well!



- (Red) What it should generate
- (Cyan) What it does generate
- (Green) What the unnested model generates
- (Grey) What is generated if the `observe` in inner is ignored

What went wrong?

- Consistency of sampling from nested queries follows from (4), but only if **every** $N_k \to \infty$
- `conditional` implicitly uses a fixed N_k giving asymptotic bias
- As `conditional` returns unweighted samples, Anglican is unable to catch the special cases

Nested Observation

- Instead of sampling from another query we might want to condition on it giving a certain output
- Statistical correctness follows directly from the linearity and products of expectations special cases given in [2]
- Can first of such cases as defining pseudo-marginal approaches
- There are still substantial semantical difficulties – in general it is not possible to evaluate the density on program outputs

Estimates as Variables

- One might wish to use estimates as variables in another program
- This allows arbitrary nested estimation problems to be encoded, but correctness must be assessed on a case-by-case basis

```
(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))
```

```
(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))
```

Acknowledgements

- BP, ESPRC

rainforth@stats.ox.ac.uk, http://www.robots.ox.ac.uk/~twgr/