Near-optimal exact sampling and optimal approximate sampling for probabilistic programming

Posters 14 & 15

Feras Saad, Cameron Freer, Martin Rinard, Vikash Mansinghka

Efficient exact and approximate sampling

As uncertainty continues to play an increasingly prominent role in a range of computations and as programming languages move towards more support for random sampling as one way of dealing with this uncertainty, we anticipate that trade-offs between entropy consumption, sampling accuracy, numerical precision, and wall-clock runtime will form an important set of design considerations for sampling procedures. This work was published as:

Exact sampler (FLDR): *The Fast Loaded Dice Roller: A near-optimal exact sampler for discrete probability distributions,* Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, PMLR 108:1036–1046, 2020.

Approximate sampler (OAS): *Optimal approximate sampling from discrete probability distributions,* Proceedings of the ACM on Programming Languages 4, POPL, 36:1–36:31, 2020.

Rejection sampler algorithms

Rejection sampling operates as follows: given a *target distribution* $(p_1, \ldots, p_n)$ and *proposal distribution* $(q_1, \ldots, q_l)$ (with $n \le l$), first find a *rejection bound* $A > 0$ such that $p_i \le Aq_i$ $(i = 1, \ldots, n)$. Next, sample $Y \sim q$ from the proposal and flip a Bernoulli$(p_Y/(Aq_Y))$ coin: if the outcome is heads accept Y, otherwise repeat. The probability of halting in a given round is:

$$\Pr\left[\text{Bernoulli}\left(\frac{p_Y}{Aq_Y}\right) = 1\right] = \sum_{i=1}^{n} p_i/(Aq_i)q_i = 1/A.$$

Algorithm 1 Uniform rejection sampler

```
  // PREPROCESS
  Let D ← max(a_1, ..., a_n);
  // SAMPLE
  while true do
    i ~ FastDiceRoller(n);    (Lumbroso [16, p. 4])
    x ~ Bernoulli(a_i, D);    (Lumbroso [16, p. 21])
    if (x = 1) then return i;
```

Algorithm 2 Dyadic rejection sampler (lookup table)

```
  // PREPROCESS
  Let k ← ⌈log m⌉;
  Make size-m table T with a_i entries i (i = 1, ..., n);
  // SAMPLE
  while true do
    Draw k bits, forming integer W ∈ {0, ..., 2^{k-1}};
    if (W < m) then return T[W];
```

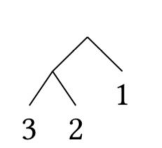
Algorithm 3 Dyadic rejection sampler (binary search)

```
  // PREPROCESS
  Let k ← ⌈log m⌉;
  Make size-n array T of running sums of a_i;
  // SAMPLE
  while true do
    Draw k bits, forming integer W ∈ {0, ..., 2^{k-1}};
    if (W < m) then return min{j | W < T[j]};
```

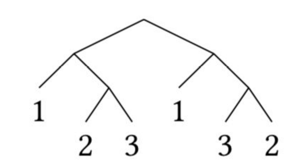
Discrete Distribution Generating (DDG) trees

$$\begin{bmatrix}p_1\\p_2\\p_3\end{bmatrix} = \begin{bmatrix}1/2\\1/4\\1/4\end{bmatrix} = \begin{bmatrix}.1\,0\\.0\,1\\.0\,1\end{bmatrix}$$

(a) Binary probability matrix



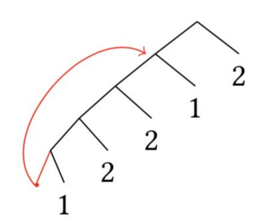
(b) Entropy-optimal DDG tree



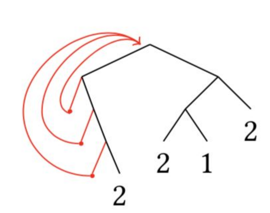
(c) Entropy-suboptimal DDG tree

$$\begin{bmatrix}p_1\\p_2\end{bmatrix} = \begin{bmatrix}3/10\\7/10\end{bmatrix} = \begin{bmatrix}.0\,\overline{1\,0\,0\,1}\\.1\,\overline{0\,1\,1\,0}\end{bmatrix}$$

(a) Binary probability matrix



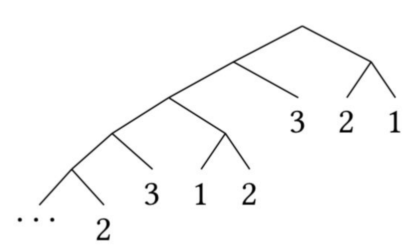
(b) Entropy-optimal DDG tree



(c) Entropy-suboptimal DDG tree

$$\begin{bmatrix}p_1\\p_2\\p_3\end{bmatrix} = \begin{bmatrix}1/\pi\\1/e\\1-1/\pi-1/e\end{bmatrix} = \begin{bmatrix}.0\,1\,0\,1\,0\,0\,0\ldots\\.0\,1\,0\,1\,1\,1\,1\ldots\\.0\,1\,0\,1\,0\,0\,0\ldots\end{bmatrix}$$

(a) Binary probability matrix



(b) Entropy-optimal DDG tree

Theorem 2.9 (Knuth and Yao [1976]). *Let $\mathbf{p} := (p_1, \ldots, p_n)$ be a probability distribution for some integer $n \ge 1$. Let A be an entropy-optimal sampler with DDG tree T whose output distribution is equal to $\mathbf{p}$. Then $H(\mathbf{p}) \le \mathbb{E}[L_T(A)] < H(\mathbf{p}) + 2$. Further, T contains exactly 1 leaf node labeled i at level j if and only if $p_{ij} = 1$, where $(0.p_{i1}p_{i2}\ldots)_2$ denotes the concise binary expansion of each p_i.*

The Fast Loaded Dice Roller (FLDR)

Idea: combine rejection sampling with inversion sampling to obtain a smaller DDG tree

Rejection Sampling	Inversion Sampling
Given probabilities $(M_i/Z)_{i=1}^n$: (1) Let k be such that $2^{k-1} < Z \le 2^k$. (2) Draw a k-bit integer $W \in \{0, \ldots, 2^k - 1\}$. (3) If $W < Z$, return integer $j \in [n]$ such that $\sum_{i=1}^{j-1} M_i \le W < \sum_{i=1}^{j} M_i$; else go to 2.	Given probabilities $(M_i/Z)_{i=1}^n$, precision k: (1) Draw a k-bit integer $W \in \{0, \ldots, 2^k - 1\}$. (2) Let $U' := W/2^k$. (3) Return smallest integer $j \in [n]$ such that $U' < \sum_{i=1}^{j} M_i/Z$.

Algorithm 4 Fast Loaded Dice Roller (sketch)

1. Let $k := \lceil \log m \rceil$ and define the proposal distribution $q := (a_1/2^k, \ldots, a_n/2^k, 1 - m/2^k)$.
2. Simulate $X \sim q$, using an entropy-optimal sampler for the proposal distribution (Theorem 2.1).
3. If $X \le n$, then return X, else go to step 2.

Algorithm 5 A sparse matrix based implementation of the Fast Loaded Dice Roller using integer arithmetic

Input: Positive integers $(a_1, \ldots, a_n)$, $m := \sum_{i=1}^{n} a_i$.
Output: Random integer i with probability a_i/m.

Table 1: Number of PRNG calls and wall-clock time when drawing 10^6 samples from $n = 1000$ dimensional distributions, using FLDR & approximate floating-point samplers.

Method	Entropy (bits)	Number of PRNG Calls	PRNG Wall Time (ms)
FLDR	1	123,607	3.69
	3	182,839	4.27
	5	258,786	5.66
	7	325,781	8.54
	9	383,138	8.42
Floating Point	all	1,000,000	21.51

Entropy near-optimality

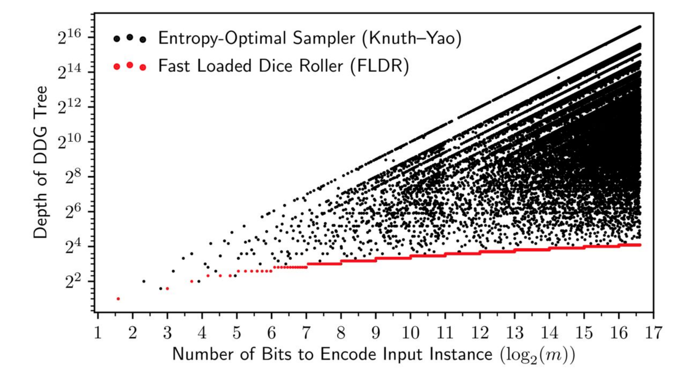


Figure 2: Depth of DDG tree for a distribution having an entry $1/m$, using the Knuth and Yao entropy-optimal sampler (black) and FLDR (red) for $m = 3, \ldots, 10^5$ (computed analytically). The y-axis is on a logarithmic scale: the entropy-optimal sampler scales exponentially (Thm. 3.5) and FLDR scales linearly (Thm. 5.1).

Theorem 5.2. *The DDG tree T of FLDR in Alg. 4 satisfies*

$$0 \le \mathbb{E}[L_T] - H(p) < 6. \tag{2}$$

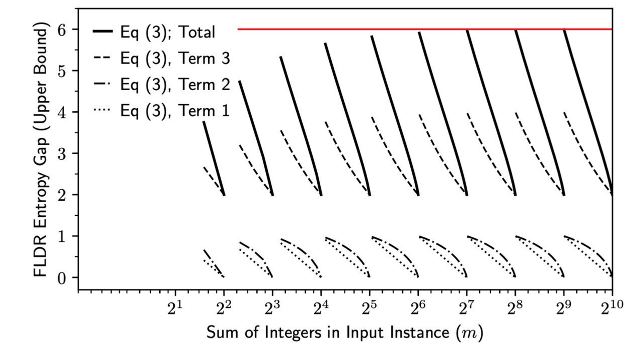
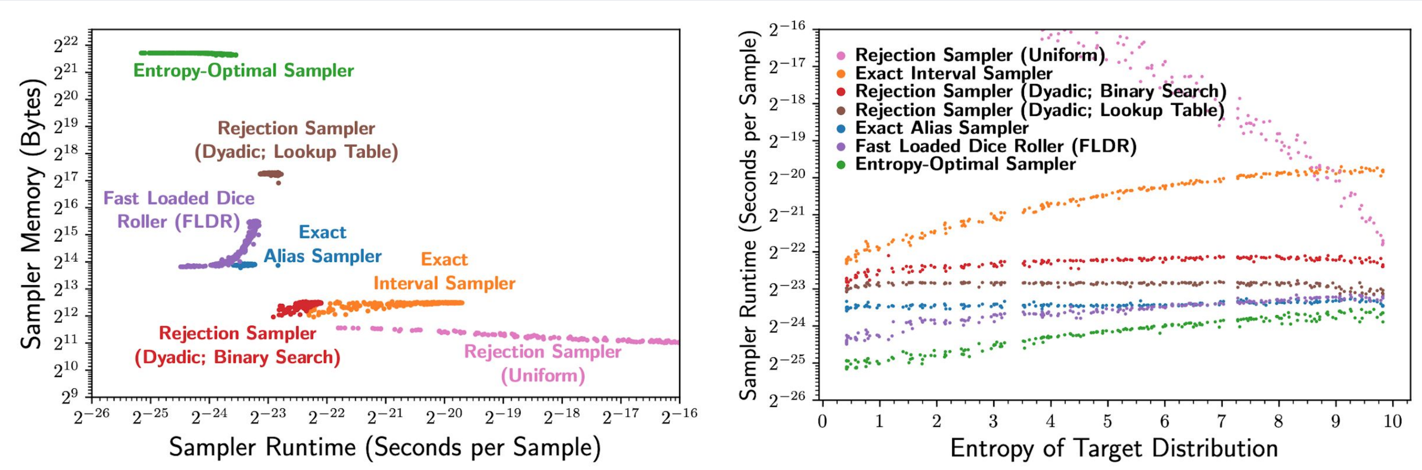


Figure 3: Plot of the three terms in Eq. (3) in the entropy gap (y-axis) from Thm. 5.2, for varying m (x-axis).

Memory, runtime, and preprocessing performance



(a) (b)

Figure 4: Comparison of memory and runtime performance for sampling 500 random frequency distributions over $n = 1000$ dimensions with sum $m = 40000$, using FLDR and six baseline exact samplers. (a) shows a scatter plot of the sampler runtime (x-axis; seconds per sample) versus sampler memory (y-axis; bytes); and (b) shows how the sampler runtime varies with the entropy of the target distribution, for each method and each of the 500 distributions.

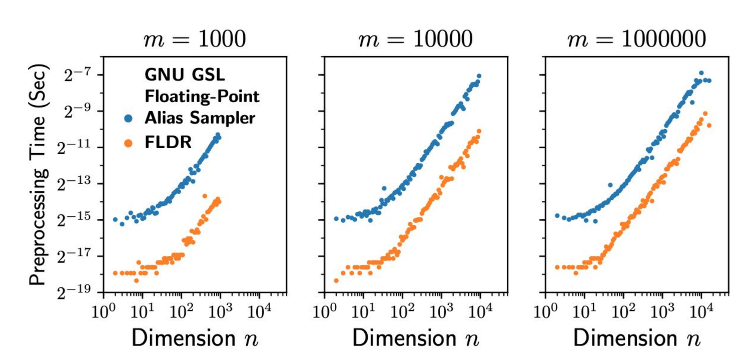


Figure 5: Comparison of the preprocessing times (y-axes; wall-clock seconds) of FLDR with those of the alias sampler, for distributions with dimension ranging from $n = 10^0, \ldots 2 \times 10^4$ (x-axes) and normalizers $m = 1000$, 10000, and 1000000 (left, center, and right panels, respectively).

Table 1: Number of PRNG calls and wall-clock time when drawing 10^6 samples from $n = 1000$ dimensional distributions, using FLDR & approximate floating-point samplers.

Method	Entropy (bits)	Number of PRNG Calls	PRNG Wall Time (ms)
FLDR	1	123,607	3.69
	3	182,839	4.27
	5	258,786	5.66
	7	325,781	7.90
	9	383,138	8.68
Floating Point	all	1,000,000	21.51

Optimal Approximate Sampler (OAS)

Problem 2.15 (Informal). *Given a target probability distribution $\mathbf{p} := (p_1, \ldots, p_n)$, a measure of statistical error Δ, and a precision bound $k \ge 1$, construct a k-bit entropy-optimal sampler $\hat{T}$ whose output probabilities $\hat{\mathbf{p}}$ achieve the minimal possible error $\Delta(\mathbf{p}, \hat{\mathbf{p}})$.*

Table 4. Precision, entropy consumption, and sampling error of rejection sampling, exact Knuth–Yao sampling, and optimal approximate sampling at various levels of precision for the Binomial(50, 61/500) distribution.

Method	Precision $k_{(l)}$	Bits/Sample	Error (L_1)
Exact Knuth and Yao Sampler (Thm. 2.9)	$5.6 \times 10^{104}_{(100)}$	5.24	0.0
Exact Rejection Sampler (Alg. 1)	$449_{(448)}$	735	0.0
Optimal Approximate Sampler (Alg. 3+7)	$4_{(4)}$	5.03	2.03×10^{-1}
	$8_{(4)}$	5.22	1.59×10^{-2}
	$16_{(0)}$	5.24	6.33×10^{-5}
	$32_{(12)}$	5.24	1.21×10^{-9}
	$64_{(29)}$	5.24	6.47×10^{-19}

Proposition 2.16. *Given a target $\mathbf{p} := (p_1, \ldots, p_n)$, an error measure Δ, and $k \ge 1$, suppose $\hat{T}$ is a k-bit entropy-optimal sampler whose output distribution is a Δ-closest approximation to $\mathbf{p}$. Then $\hat{\mathbf{p}}$ is closer to $\mathbf{p}$ than the output distribution $\tilde{\mathbf{p}}$ of any sampler $\tilde{T}$ that halts after consuming at most k random bits from the source.*

Table 2. Comparison of the average number of input bits per sample used by inversion sampling, interval sampling, and the proposed method, in each of the six parameterized families using $k = 16$ bits of precision.

Distribution	Average Number of Bits per Sample		
	Inversion Sampler (Alg. 2)	Interval Sampler [Uyematsu and Li 2003]	Optimal Sampler (Alg. 7)
Benford	16	6.34	5.71
Beta Binomial	16	4.71	4.16
Binomial	16	5.05	4.31
Boltzmann	16	1.51	1.03
Discrete Gaussian	16	6.00	5.14
Hypergeometric	16	4.04	3.39

f-divergences: statistical error measures

Table 1. Common statistical divergence measures expressed as f-divergences.

Divergence Measure	Formula $\Delta_g(\mathbf{p}, \mathbf{q})$	Generator $g(t)$
Total Variation	$\frac{1}{2}\sum_{i=1}^{n} \lvert q_i - p_i \rvert$	$\frac{1}{2}\lvert t - 1 \rvert$
Hellinger Divergence	$\frac{1}{2}\sum_{i=1}^{n} (\sqrt{p_i} - \sqrt{q_i})^2$	$(\sqrt{t} - 1)^2$
Pearson Chi-Squared	$\sum_{i=1}^{n} (q_i - p_i)^2/q_i$	$(t - 1)^2$
Triangular Discrimination	$\sum_{i=1}^{n} (p_i - q_i)^2/(p_i + q_i)$	$(t - 1)^2/(t + 1)$
Relative Entropy	$\sum_{i=1}^{n} \log(q_i/p_i)q_i$	$t \log t$
α-Divergence	$4(1 - \sum_{i=1}^{n}(p_i^{(1-\alpha)/2} q_i^{1+\alpha}))/(1 - \alpha^2)$	$4(1 - t^{(1+\alpha)/2})/(1 - \alpha^2)$

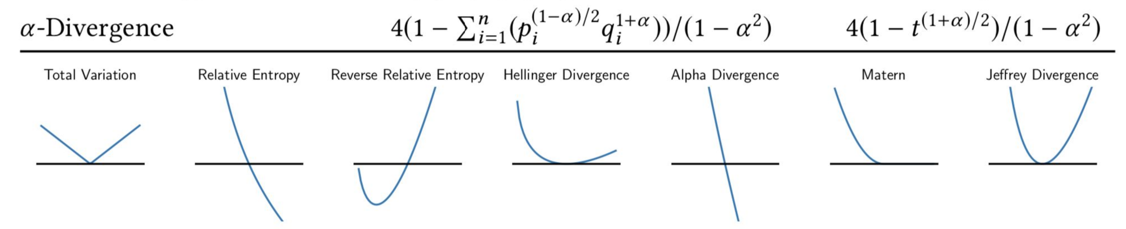


Fig. 4. Plots of generating functions g for various f-divergences, a subset of which are shown in Table 1.

Performance comparison on particular distributions

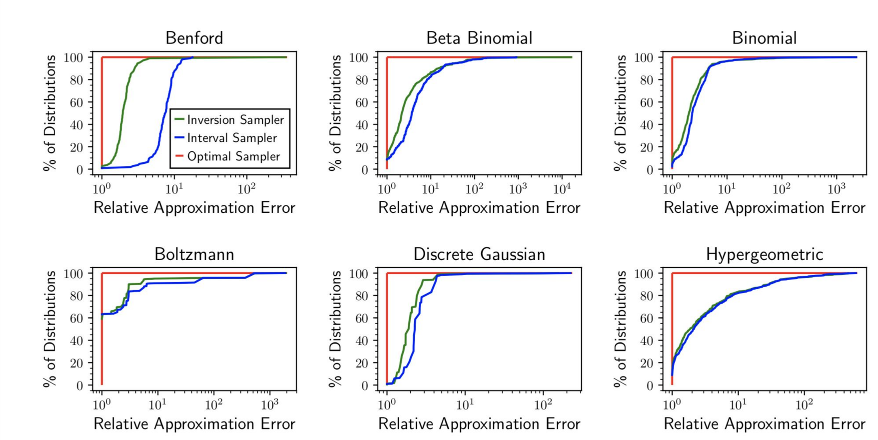


Fig. 6. Comparison of the approximation error of limited-precision implementations of interval sampling (green) and inversion sampling (blue) relative to error obtained by the optimal sampler (red), for six families of probability distributions using $k = 16$ of bits precision. The x-axis shows the approximation error of each sampler relative to the optimal error. The y-axis shows the fraction of 500 distributions from each family whose relative error is less than or equal to the corresponding value on the x-axis.

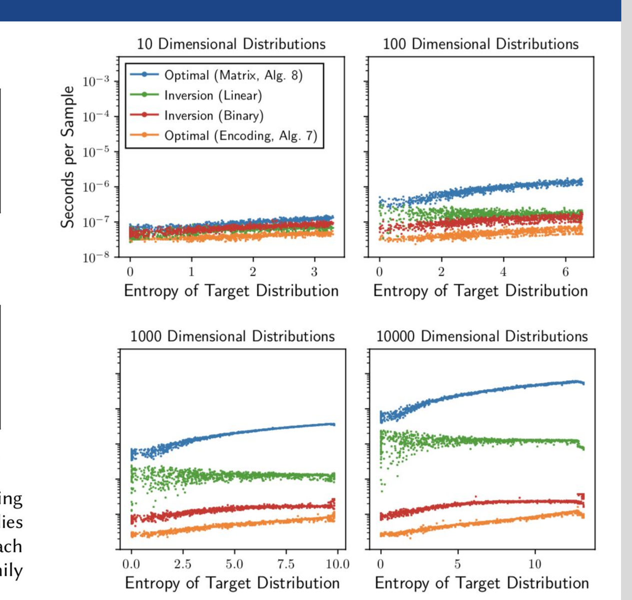
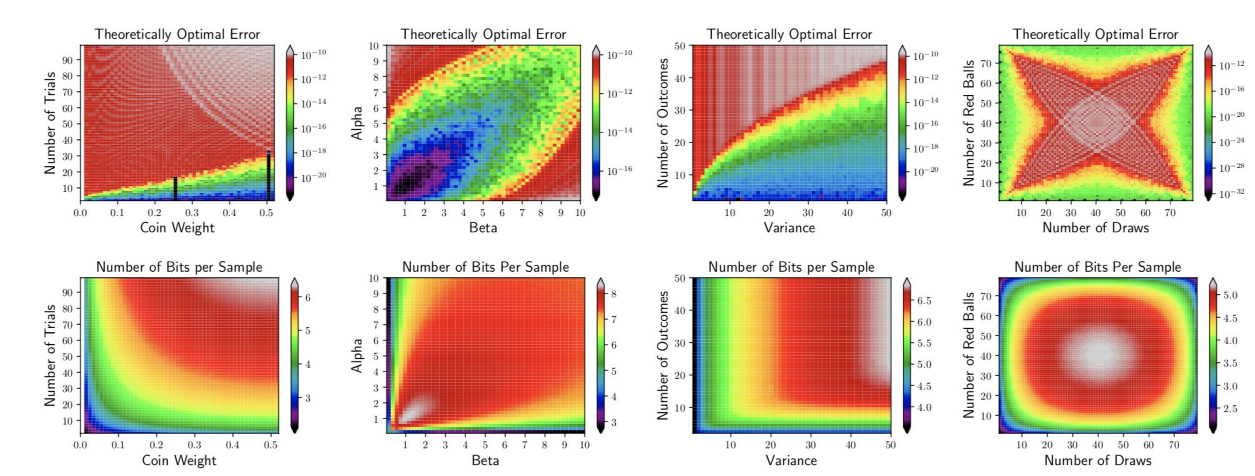


Fig. 7. Comparison of wall-clock time per sample and order of growth of two implementations of the optimal samplers (using Algorithms 7 and 8) with inversion sampling (using linear and binary search in Algorithm 2).



(a) Binomial (b) Beta Binomial ($n = 80$) (c) Discrete Gaussian (d) Hypergeometric ($n = 80$)

Fig. 5. Characterization of the theoretically optimal approximation error (top row) and average number of bits per sample (bottom row) for four common families of probability distributions using $k = 32$ bits of precision.