Consistent Sampling with Replacement

Ronald L. Rivest
MIT CSAIL
{mayuri,rivest}@mit.edu

August 15, 2018

Abstract

We describe a sample method for "consistent sampling" that allows for sampling with replacement.

1 Introduction

We describe a method for "consistent sampling."

Python 3 code for this method is given in Github:

`https://github.com/ron-rivest/consistent_sampler`

We assume the existence of a finite population $\mathcal{I} = \{1, 2, \ldots, n\}$ of n items from which we wish to draw a sample S of size s, where sampling may be drawn "with replacement" or "without replacement."

The method works as shown in Figure 1.

We assume that the function $f(i, u)$ produces a result that is uniformly distributed over the interval $(0, 1)$ (as u varies and i remains fixed).

We more generally assume that for any i the sequence

$$\tau_{i,1}, \tau_{i,2}, \tau_{i,3}, \ldots$$

is indistinguishable from a sequence

$$x_1, x_2, x_3, \ldots \tag{1}$$

where x_i is chosen uniformly from the real interval $(0, 1)$ and for $i > 1$, x_i is chosen uniformly from the interval $(x_{i-1}, 1)$.

The method works for sampling with replacement, since when a ticket with number τ is drawn from Q because it has the minimum ticket number, then all of the remaining tickets in Q have ticket numbers that may be assumed to be uniformly distributed in $(\tau, 1)$, conditional on having just drawn

1. Select a seed u at random from a large universe $\mathcal{U}$ of possible seeds.
2. Create a "first ticket" $(\tau_{i,1}, i, 1)$ for each item i, for $i = 1, 2, \ldots, n$ where the first ticket number

$$\tau_{i,1} = f(i, u)$$

   is the result of applying pseudorandom function f to inputs i and u to yield a result uniformly distributed in the interval $(0, 1)$.
3. Initialize set Q to contain the set of tickets so created.
4. Initialize the sample S to be the empty set ϕ.
5. While S has size less than s:
   (a) Extract from Q the ticket t with the least ticket number.
   (b) Let $t = (\tau_{i,j}, i, j)$. Place item i into set S.
   (c) If we are drawing with replacement, then
   - Add ticket t' to Q, where $t' = (\tau_{i,j+1}, i, j + 1)$, and where

$$\tau_{i,j+1} = g(\tau_{i,j})$$

     for a suitable pseudo-random function g whose result is uniformly distributed in the interval $(\tau_{i,j}, 1)$.
6. Return S as the desired sample of size s.

Figure 1: The consistent sampling method, based on pseudorandom functions f and g. The priority queue Q contains exactly one ticket (τ, i, j) for each item i. The value j is the "generation number" for the ticket, saying how many tickets have been generated for this item so far. If we are sampling without replacement, tickets only have generation number 1. Otherwise, tickets with generation number greater than 1 are replacement tickets.

a ticket with number τ. So adding a replacement ticket with ticket number drawn uniformly from $(\tau, 1)$ makes the new ticket indistinbuishable from those already there.

Another useful way of looking what happens with sampling replacement is to view Q as being initialized with an infinite number of tickets for each item, one for each possible generation. Then sampling from this Q without replacement is equivalent to sampling from the original Q with replacement.

Suitable functions f and g are constructible from, say, the cryptographic hash function SHA256. (See the python code in Github for details.) These functions can be implemented in an efficient manner, with only a few calls to the underlying SHA256 hash function required per invocation of f or g. The function g does not need to take seed u as an input if the values τ are computed in a way that preserves the full output entropy of the SHA256 hash function.

The consistent sampling method puts the elements of $\mathcal{I}$ into a shuffled order. A sample is then just a prefix of that order.

The method of assigning a number (which we call a "ticket number") to each element is not new, nor is the term "consistent sampling." (See references [2, 1] and the citations therein.)

The sampling method is consistent in the sense that if $\mathcal{I}$ is a population of items, and if $\mathcal{I}'$ is a subset of $\mathcal{I}$, then the order produced for $\mathcal{I}'$ is a subsequence of the order produced for $\mathcal{I}$.

2 Discussion

The extension of consistent sampling to handle sampling with replacement (step 5(c) in the figure) might be new.

It is interesting to ask about the relationship between the number s of items drawn for the sample and the ticket number (call it τ_s) of the last ticket drawn. Or similarly, if one draws all items with ticket number less than a limit λ, one may be interested in the distribution of the number s of items drawn.

In this direction, we note that if we define

$$y_i = 1 - x_i \tag{2}$$

where the xs are as in (1), then y_k is distributed as the product of k independent uniform variates $z_1, \ldots, z_k$. Since

$$\ln(z_i) \sim -\text{Exp}(1),$$

we have

$$\ln(y_k) \sim -\text{Gamma}(k, 1),$$

and

$$E(\ln(y_k)) = -k \ .$$

Therefore, if the proposed method is to be used for sampling with replacement where a given item may be selected and replaced many (perhaps hundreds) of times, then the representations of $\tau(i, j)$ should have sufficient precision to handle numbers that are extremely close to 1 (or if the ys are represented instead of the xs, to handle numbers with large negative exponents). That is to say, the

number of bits needed to represent $\tau(i,j)$ grows linearly with j. The representation in the python code

`https://github.com/ron-rivest/consistent_sampler`

uses variable-length numbers with no upper limit on the precision.

References

[1] Mohammad Bavarian, Badih Ghazi, Elad Haramaty, Pritish Kamath, Ronald L. Rivest, and Madhu Sudan. The optimality of correlated sampling. *CoRR*, abs/1612.01041, 2016.

[2] Konstantin Kutzkov and Rasmus Pagh. Consistent subset sampling. *CoRR*, abs/1404.4693, 2014.