How to Sample Unbounded Monotone Density Functions

This version of the document is dated 2020-07-26.

Peter Occil

This short note shows a preprocessing algorithm to generate a random number in [0, 1] from a distribution whose density function—

- is continuous in the interval [0, 1],
- is monotonically decreasing in [0, 1], and
- has an unbounded peak at 0.

The trick here is to sample the peak in such a way that the result is either forced to be 0 or forced to belong to the bounded part of the density function. This algorithm does not require the area under the curve of the density in [0, 1] to be 1; in other words, this algorithm works even if the density function is known up to a normalizing constant. The algorithm is as follows.

1. Set i to 1.
2. Calculate the cumulative probability of the interval [0, 2^{-i}] and that of [0, $2^{-(i-1)}$], call them p and t, respectively.
3. With probability p/t, add 1 to i and go to step 2. (Alternatively, if i is equal to or higher than the desired number of fractional bits in the result, return 0 instead of adding 1 and going to step 2.)
4. At this point, the density function at [2^{-i}, $2^{-(i-1)}$) is bounded from above, so sample a random number in this interval using any appropriate algorithm, including rejection sampling. Because the density is monotonically decreasing, the peak of the density at this interval is located at 2^{-i}, so that rejection sampling becomes trivial.

It is relatively straightforward to adapt this algorithm for monotonically increasing density functions with the unbounded peak at 1, or to density functions with a different domain than [0, 1].

This algorithm is similar to the “inversion-rejection” algorithm mentioned in section 4.4 of chapter 7 of Devroye’s *Non-Uniform Random Variate Generation* (1986)**(1)**. I was unaware of that algorithm at the time I started writing this article. The difference here is that it assumes the whole distribution (including its density function and cumulative distribution function) is supported on the interval [0, 1], while the algorithm presented in this article doesn’t make that assumption (e.g., the interval [0, 1] can cover only part of the density’s support).

By the way, this algorithm arose while trying to devise an algorithm that can generate an integer power of a uniform random number, with arbitrary precision, without actually calculating that power (a naïve calculation that is merely an approximation and usually introduces bias); for more information, see my other article on **partially-sampled random numbers**. Even so, the algorithm I have come up with in this note may be of independent interest.

In the case of powers of a uniform [0, 1] random number X, namely X^n, the ratio p/t in this algorithm has a very simple form, namely $(1/2)^{1/n}$, which is possible to simulate using a so-called *Bernoulli factory* algorithm without actually having to calculate this ratio. Note that this formula is the same regardless of i. This is found by taking the density function $f(x) = x^{1/n}/(x * n)$ and finding the appropriate p/t ratios by integrating f over the two intervals mentioned in step 2 of the algorithm.

Notes

[1] Devroye, L., ***Non-Uniform Random Variate Generation***, 1986.

License