Simple Random Sampling: Not So Simple

Kellie Ottoboni
with Philip B. Stark and Ron Rivest

Department of Statistics, UC Berkeley
Berkeley Institute for Data Science

Moore-Sloan DSE Summit
October 24, 2016

PRNGs

- **Simple random sampling:** drawing $k \leq n$ items from a population of n items, in such a way that each of the $\binom{n}{k}$ subsets of size k is equally likely.
- Difficult to obtain truly random samples. Instead, use **pseudorandom number generators (PRNGs)** to select items
- **Pseudorandom**: computationally indistinguishable from the uniform distribution

Good PRNGs produce pseudorandom sequences. Do they give simple random samples with equal probabilities?

The good, the bad, and the ugly

Knuth [1997]

"Random numbers should not be generated with a method chosen at random."

RANDU: the sequence (x_n) given by

$$x_{n+1} = (65539 x_n) \mod 2^{31}.$$

$0.003051898, 0.018310966, 0.082398718, 0.329593616, 0.235973230, \ldots$

The good, the bad, and the ugly

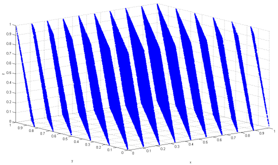
Knuth [1997]

"Random numbers should not be generated with a method chosen at random."

RANDU: the sequence (x_n) given by

$$x_{n+1} = (65539 x_n) \mod 2^{31}.$$

$0.003051898, 0.018310966, 0.082398718, 0.329593616, 0.235973230, \ldots$



Triples of RANDU lie on 15 planes in 3D space (Wikipedia)

Pigeons and Pigeonholes

Theorem (Pigeonhole Principle)

If there are n pigeonholes and $m > n$ pigeons, then there exists at least one pigeonhole containing more than one pigeon.

(Wikipedia)

Pigeons and Pigeonholes

Theorem (Pigeonhole Principle)

If there are n pigeonholes and $m > n$ pigeons, then there exists at least one pigeonhole containing more than one pigeon.

(Wikipedia)

Corollary (Too few pigeons)

If $\binom{n}{k}$ is greater than the size of a PRNG's state space, then the PRNG cannot possibly generate all samples of size k from a population of n.

Pigeons and Pigeonholes

Does it matter in practice?

Pigeons and Pigeonholes

Does it matter in practice?

Period of 32-bit linear congruential generators (e.g. RANDU):
$2^{32} \approx 4 \times 10^9$
Samples of size 10 from 50: $\binom{50}{10} \approx 10^{10}$
More than half of samples cannot be generated

Pigeons and Pigeonholes

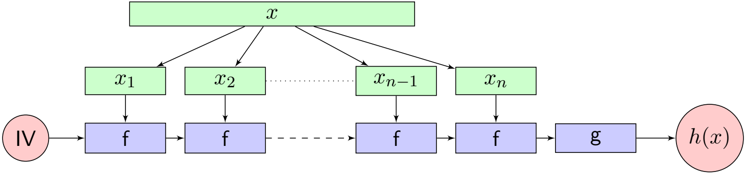
Does it matter in practice?

Period of 32-bit linear congruential generators (e.g. RANDU):
$2^{32} \approx 4 \times 10^9$
Samples of size 10 from 50: $\binom{50}{10} \approx 10^{10}$
More than half of samples cannot be generated

Period of Mersenne Twister (standard PRNG in Statistics):
$2^{32\times 624} \approx 2 \times 10^{6010}$
Permutations of 2084 objects: $2084! \approx 3 \times 10^{6013}$
Less than 0.01% of permutations can be generated

A better alternative

One solution: Find a class of PRNGs with infinite state space



Cryptographic hash functions:

- computationally infeasible to invert
- difficult to find two inputs that map to the same output
- small input changes produce large, unpredictable changes to output
- resulting bits are uniformly distributed

Choice of seed

Preliminary results: the distribution of simple random samples is less uniform if you use a stupid seed

PRNG	p-value (seed $=$ 100)	p-value (seed $=$ 233424280)
RANDU	0	0
Super-Duper LCG	0.1798	1
Mersenne Twister*	0.0858	0.4741
Mersenne Twister	0.1996	0.6143
SHA-256 PRNG	0.1710	0.8584

* using `np.random.choice` to sample

Open questions

- Sampling algorithms: do some give samples with unequal probability?
- For PRNGs with sufficiently large state space, do they produce all samples with equal probability? All permutations?
- Are departures from uniformity large enough to bias statistics of interest?
- Replace the default PRNGs in Python
`https://www.github.com/statlab/cryptorandom`
- Results apply more broadly to computer simulations: permutation tests, bootstrapping, MCMC, etc.

References

Donald E. Knuth. *Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley Professional, Reading, Massachusetts, 3rd edition, November 1997.