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 Data Science Summit
October 24, 2016

University of California, Berkeley
Department of Statistics

BIDS
Berkeley Institute for Data Science

PRNGs

Pseudorandom number generator: a deterministic algorithm that produces sequences that are computationally indistinguishable from the uniform distribution

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

The good, the bad, and the ugly

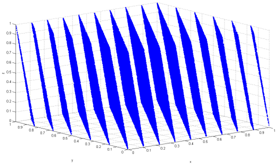
(Knuth, 1997)

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

The good, the bad, and the ugly

(Knuth, 1997)

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



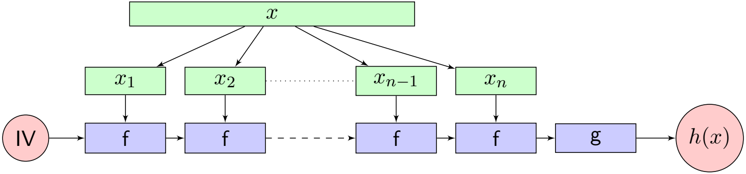
Triples of RANDU lie on 15 planes in 3D space

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

(Wikipedia)

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

- Do “good” PRNGs produce all samples with equal probability? All permutations?

Open questions

- Do “good” PRNGs produce all samples with equal probability? All permutations?
- Do departures from uniformity introduce bias?

Open questions

- Do “good” PRNGs produce all samples with equal probability? All permutations?
- Do departures from uniformity introduce bias?
- Replace the default PRNGs in Python
  `https://www.github.com/statlab/cryptorandom`

Open questions

- Do “good” PRNGs produce all samples with equal probability? All permutations?
- Do departures from uniformity introduce bias?
- Replace the default PRNGs in Python
  `https://www.github.com/statlab/cryptorandom`
- Results apply more broadly to computer simulations: permutation tests, bootstrapping, MCMC, etc.

Thanks!

https://github.com/kellieotto/msdse-summit-talk-2016