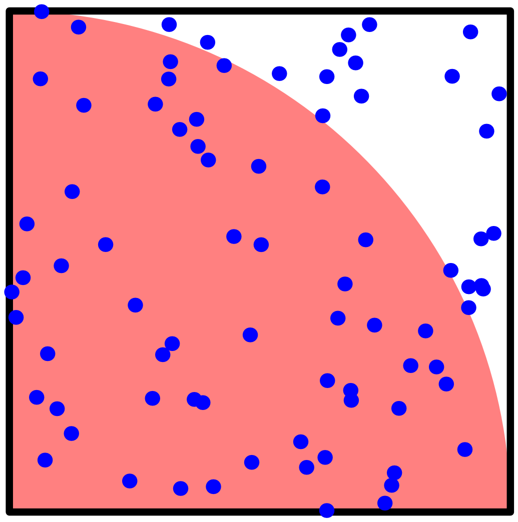
CS145: Probability & Computing

Lecture 16: Monte Carlo Methods

Figure credits:
*Bertsekas & Tsitsiklis, **Introduction to Probability**, 2008*
*Pitman, **Probability**, 1999*

CS145: Lecture 16 Outline

- Monte Carlo Methods



The (Weak) Law of Large Numbers

$X_1, X_2, \ldots$ i.i.d.
finite mean μ and variance σ^2

$$M_n = \frac{X_1 + \cdots + X_n}{n}$$

sample mean or empirical mean

$$E[M_n] = \mu \qquad \mathrm{Var}[M_n] = \frac{n\sigma^2}{n^2} = \frac{\sigma^2}{n}$$

- Chebyshev's inequality bounds distance between the true mean and the "empirical" or "sample" mean:

$$\mathbf{P}(|M_n - \mu| \geq \epsilon) \ \leq \ \frac{\mathrm{Var}(M_n)}{\epsilon^2} \ = \ \frac{\sigma^2}{n\epsilon^2}$$

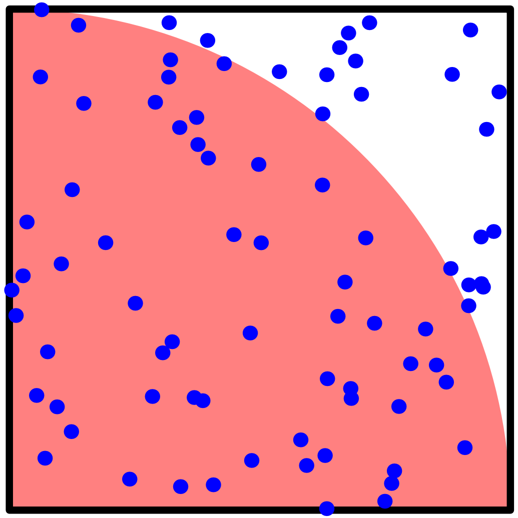
- *The empirical mean converges to the true mean in probability*

$$\lim_{n \to \infty} P(|M_n - \mu| \geq \epsilon) = 0$$

- True even if variance not finite, but proof more challenging.

Example: Monte Carlo Estimation of π

1. For $i = 1$ to N
   1. Choose X and Y uniformly at random from $[0, 1]$
   2. If $X^2 + Y^2 \leq 1$ then $Z_i = 1$ else $Z_i = 0$.
2. $Z = \sum_{i=1}^{N} 4Z_i$
3. $S = \frac{1}{N} \sum_{i=1}^{N} 4Z_i$

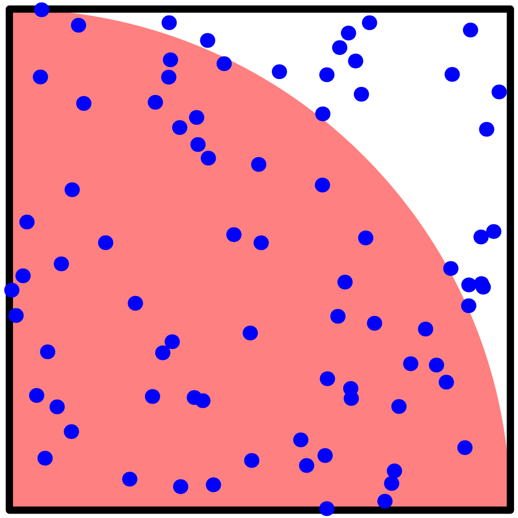


Z_i is a 0-1 r.v. with $Pr(Z_i = 1) = \frac{\pi}{4}$.

$$E[Z_i] = \frac{\pi}{4} \qquad Var[Z_i] = \frac{\pi}{4}(1 - \frac{\pi}{4})$$

Example: Monte Carlo Estimation of π

Z_i is a 0-1 r.v. with $Pr(Z_i = 1) = \frac{\pi}{4}$.



$$E[Z] = \sum_{i=1}^{N} E[4Z_i] = N\pi$$

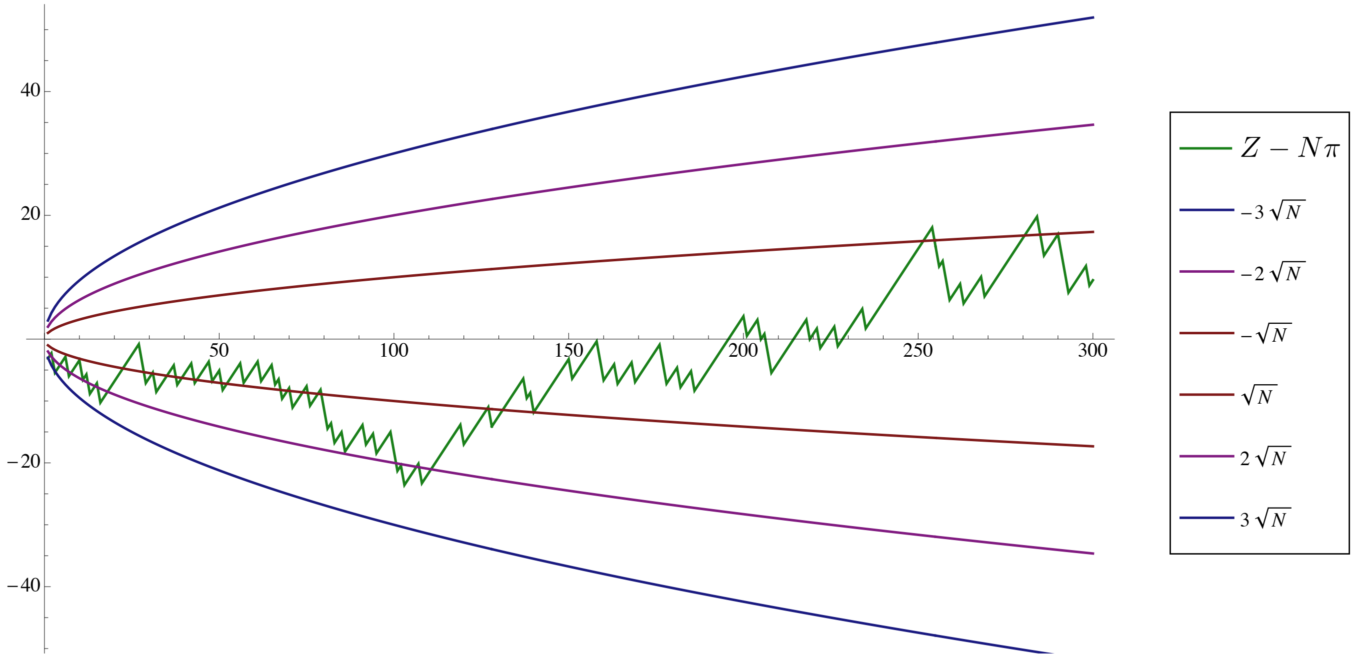
$$Var[Z] = \sum_{i=1}^{N} Var[4Z_i] = 16N\frac{\pi}{4}(1-\frac{\pi}{4})$$

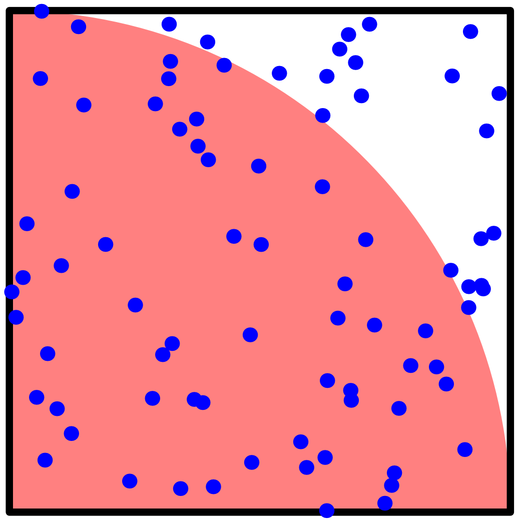
$E[Z - N\pi] = 0$

$$Var[Z - N\pi] = Var[Z] = 16N\frac{\pi}{4}(1 - \frac{\pi}{4})$$

$$\sigma[Z] = 4\sqrt{N\frac{\pi}{4}(1 - \frac{\pi}{4})} \approx 1.64\sqrt{N}$$

Example: Monte Carlo Estimation of π





$$E[Z - N\pi] = 0$$

$$Var[Z - N\pi] = Var[Z] = 16N\frac{\pi}{4}(1 - \frac{\pi}{4})$$

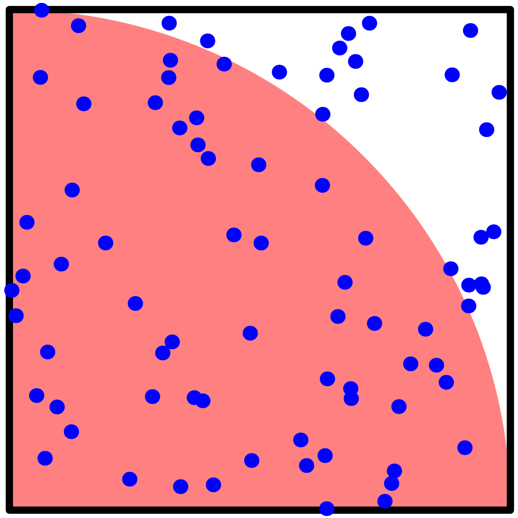
$$\sigma[Z] = 4\sqrt{N\frac{\pi}{4}(1 - \frac{\pi}{4})} \approx 1.64\sqrt{N}$$

Example: Monte Carlo Estimation of π

Z_i is a 0-1 r.v. with $Pr(Z_i = 1) = \frac{\pi}{4}$.

$$E[S] = \frac{1}{N} \sum_{i=1}^{N} 4E[Z_i] = \pi$$

$$Var[S] = \frac{1}{N^2} \sum_{i=1}^{N} Var[4Z_i] = \frac{16\frac{\pi}{4}(1 - \frac{\pi}{4})}{N}$$



Averaging many observations maintains the average but reduces the variance (standard deviation).

```
octave:1> S=12; a=rand(S,2); 4*mean(sum(a.*a,2)<1)
ans = 3.3333
octave:2> S=1e7; a=rand(S,2); 4*mean(sum(a.*a,2)<1)
ans = 3.1418
```

I. Murray, MLSS 2009

How Good is the Estimate?

Chebyshev's Inequality:

Theorem

For **any** *random variable* X, *and any* $a > 0$,

$$Pr(|X - E[X]| \geq a) \leq \frac{Var[X]}{a^2}.$$

$$E[S] = \frac{1}{N} E[4Z_i] = \pi,$$

$Var[4Z_i] \leq 16$, since $0 < 4Z_i < 4$. $Var[S] = \frac{16}{N}$

$$Pr(|S - \pi| \geq \epsilon) \leq \frac{16}{N\epsilon^2}$$

For $N \geq 128,000$,

$$Pr(|S - \pi| \geq 0.05) \leq 0.05$$

How Good is the Estimate?

Theorem (Hoeffding's Inequality)

Let $X_1, \ldots, X_n$ be independent random variables such that for all $1 \leq i \leq n$, $E[X_i] = \mu$ and $Pr(a \leq X_i \leq b) = 1$. Then

$$Pr\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \geq \epsilon\right) \leq 2e^{-2n\epsilon^2/(b-a)^2}$$

$E[S] = \frac{1}{N}\sum_{i=1}^{N} E[4Z_i] = \pi$, and $0 \leq 4Z_i \leq 4$

$P(|S - \pi| \geq \epsilon) \leq 2e^{-2n\epsilon^2/4^2}$

For $\epsilon = \sqrt{\frac{8\ln(2/\delta)}{n}}$, $\quad P(|S - \pi| \geq \epsilon) \leq \delta$

For $n = 12,000$, $\quad P(|S - \pi]| \geq 0.05) \leq 0.05$

Numerical Integration

We can also estimate π as $4\int_0^1 \sqrt{1-x^2}dx$

Let $a = t_0 \leq x_1 \leq t_1 \leq x_2 \leq t_1, ..., \leq x_n \leq t_n = b$. Let $\Delta_i = t_i - t_{i-1}$, and $\Delta = Max_{1 \leq i \leq n} \Delta_i$

$\int_a^b f(x)dx = \lim_{\Delta \to 0} \sum_{i=1}^n f(x_i)\Delta_i$ (when exists and unique..)

To estimate $V = \int_a^b f(x)dx$, choose $x_1, \ldots, x_n$ uniformly at random in $[a, b]$, and compute:

$\tilde{V} = \frac{1}{n}\sum_{i=1}^n f(x_i)$

$E[\tilde{V}] = V$, $Var[\tilde{V}] = \frac{Var[f(x_i]}{N}$

Monte Carlo Methods

$$E[g] = \int g(x) f_X(x)\, dx$$

For many complex models, integral is intractable but we can still:

- *Simulate* the target distribution: $P(X_i \leq x_i) = F_X(x_i)$
- *Evaluate* the target function: $g_i = g(x_i)$

A *Monte Carlo method* uses computer simulation to approximate:

$$E[g] \approx \frac{1}{n} \sum_{i=1}^{n} g(x_i) = M_n \qquad P(X_i \leq x_i) = F_X(x_i)$$

Selecting $x_1, \ldots, x_n$ according to the distribution $F_X(x)$

Monte Carlo Methods

$$E[g] = \int g(x) f_X(x)\, dx$$

Desirable properties of Monte Carlo estimates:

- *Unbiased* for any sample size *n*: $E[M_n] = E[g]$
- *Law of large numbers*: $\mathrm{Var}[M_n] = \frac{1}{n}\mathrm{Var}(g(X))$

$$\lim_{n\to\infty} P(|M_n - E[g]| \geq \epsilon) = 0$$

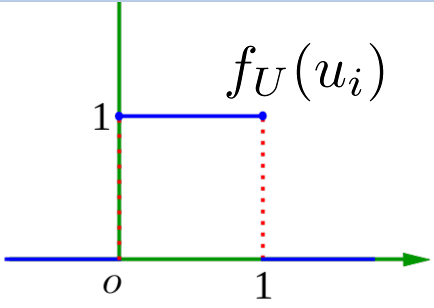
A *Monte Carlo method* uses computer simulation to approximate:

$$E[g] \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i) = M_n \qquad P(X_i \leq x_i) = F_X(x_i)$$

PROBLEM: How do we simulate the target distribution on a computer?

Monte Carlo: The Standard Input

- Assumption: We have a source of independent random variables which follow a continuous *uniform* distribution on [0,1]: $U_1, U_2, U_3, \ldots$
- In Matlab, the **rand** function provides this

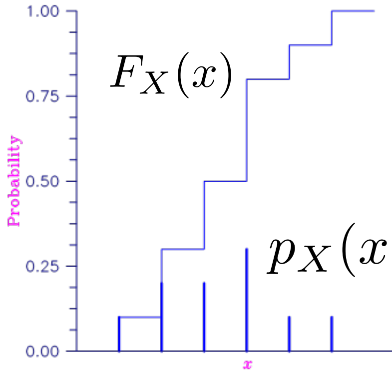


Coming later: Methods for generating uniform variables…

Discrete Random Number Generation

- Input: Independent uniform variables $U_1, U_2, U_3, \ldots$

Assume that the distribution has k values, $a_1,\ a_2,\ \ldots, a_k,$ with probabilities $p_1,\ p_2,\ \ldots, p_k.$ $\sum_1^k p_i = 1$



Partition the interval [1,0] to *k* segments of lengths $p_1,\ p_2,\ \ldots, p_k$. If U_i is in the k segment output a_k

$F_X(a_i) = \sum_1^i p_i$. Segment i = $[F_X(a_{i-1})\ F_X(a_i))$

$$a = h(u) = \min\{x \mid F_X(x) \geq u\}$$

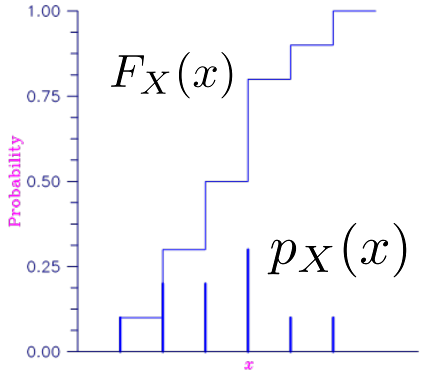
$$P(X_i = k) = P(F_X(k-1) < U_i \leq F_X(k)) = F_X(k) - F_X(k-1) = p_X(k)$$

This function transforms uniform variables to our target distribution!

Discrete Random Number Generation

- Input: Independent uniform variables $U_1, U_2, U_3, \ldots$
- We can use these to exactly sample from any discrete distribution using the cumulative distribution function:

$$F_X(x) = P(X \leq x) = \sum_{k \leq x} p_X(k)$$



- Define the "inverse" of this discrete CDF:

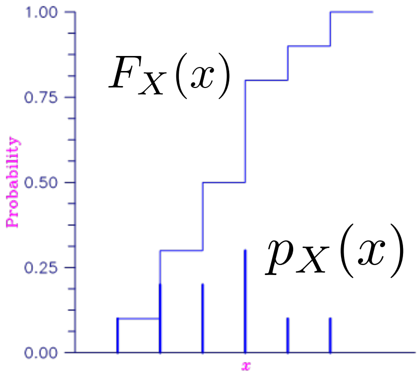
$$h(u) = \min\{x \mid F_X(x) \geq u\}$$
$$X_i = h(U_i)$$

$$P(X_i = k) = P(F_X(k-1) < U_i \leq F_X(k)) = F_X(k) - F_X(k-1) = p_X(k)$$

This function transforms uniform variables to our target distribution!

Discrete Random Number Generation

Example:
Generating Binomial random variables : $X_1, X_2, X_3, \ldots$
Input: Independent U[0,1]: $U_1, U_2, U_3, \ldots$



B(8, 0.25):

n =	0	1	2	3	4	5	6	7	8
P(n) =	0.100	0.267	0.311	0.208	0.087	0.023	0.003	0.001	0.000
$F_X(n)$ =	0.100	0.367	0.678	0.886.	0.973.	0.996	0.999.	1.000	1.000

If $0.367 < U_i \leq 0.678$, then $X_i = 2$ $\Pr(X_i = 2) = 0.678 - 0.367 = 0.311$

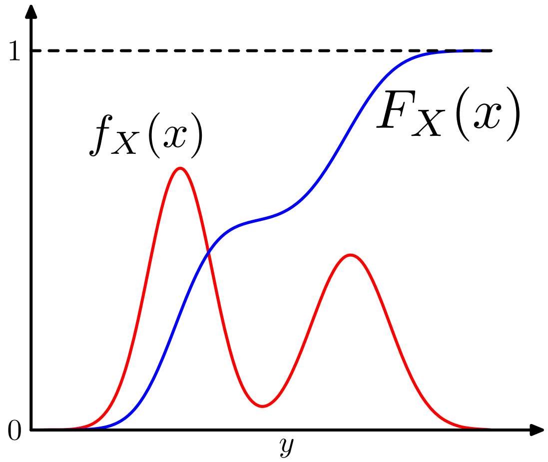
$$h(u) = F_X^{-1}(u)$$

$$P(X_i = k) = P(F_X(k-1) < U_i \leq F_X(k)) = F_X(k) - F_X(k-1) = p_X(k)$$

Continuous Random Number Generation

- Input: Independent uniform variables $U_1, U_2, U_3, \ldots$
- We can use these to exactly sample from any continuous distribution using the cumulative distribution function:

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f_X(z)\, dz$$



- Assuming continuous CDF is invertible:

$$h(u) = F_X^{-1}(u)$$

$$X_i = h(U_i)$$

$$P(X_i \leq x) = P(h(U_i) \leq x) = P(U_i \leq F_X(x)) = F_X(x)$$

This function transforms uniform variables to our target distribution!

Continuous Random Number Generation

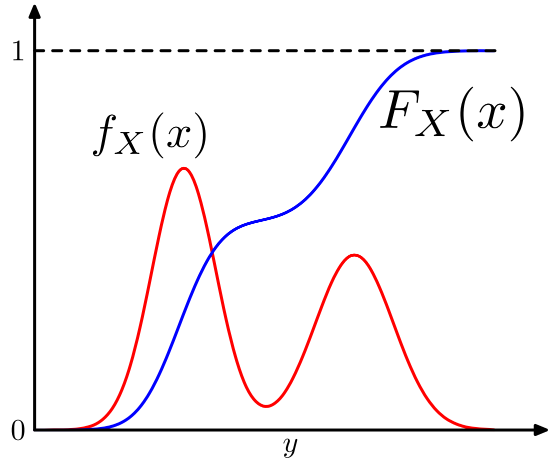
Input: Independent U[0,1] random values $U_1, U_2, U_3, \ldots$

Output: Independent Exp(θ) random values $X_1, X_2, X_3, \ldots,$

We use $U_i = 1 - e^{-\theta x_i}$ to compute:

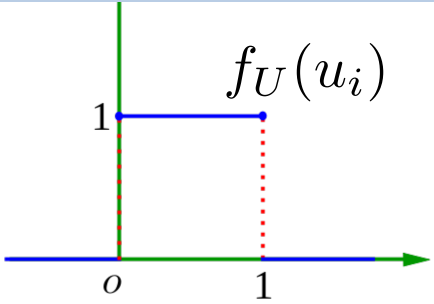
$$u = F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f_X(z)\, dz$$

$$h(u) = F_X^{-1}(u) \qquad X_i = h(U_i)$$



$$P(X_i \leq x) = P(h(U_i) \leq x) = P(U_i \leq F_X(x)) = F_X(x)$$

Uniform Random Number Generation



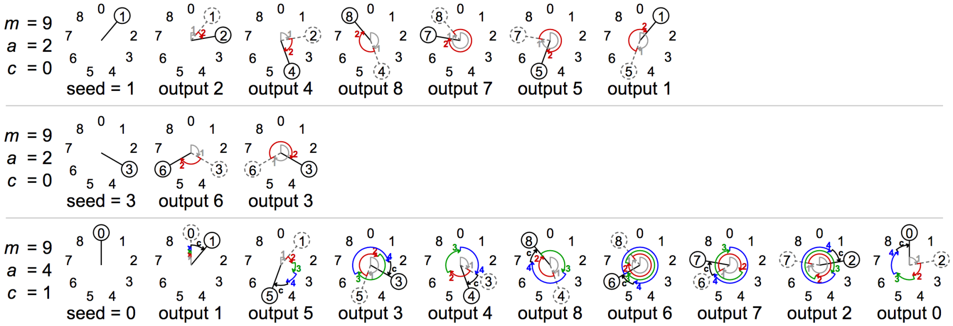
- Assumption: We have a source of independent random variables which follow a continuous *uniform* distribution on [0,1] : $U_1, U_2, U_3, \ldots$
- In Matlab, the **rand** function provides this
- Chaotic dynamical systems are used to generate sequences of *pseudo-random numbers* whose distribution is approximately uniform on [0,1]
- Simplest examples are *linear congruential generators*, but try to *use more sophisticated methods!*

$$\bar{u}_{i+1} = (a\bar{u}_i + c) \bmod m \qquad u_i = \frac{1}{m}\bar{u}_i$$

c and *m* should be *relatively prime and large* (plus more).

Examples of Uniform Generators

Linear Congruential Generator



$$\bar{u}_{i+1} = (a\bar{u}_i + c) \bmod m$$

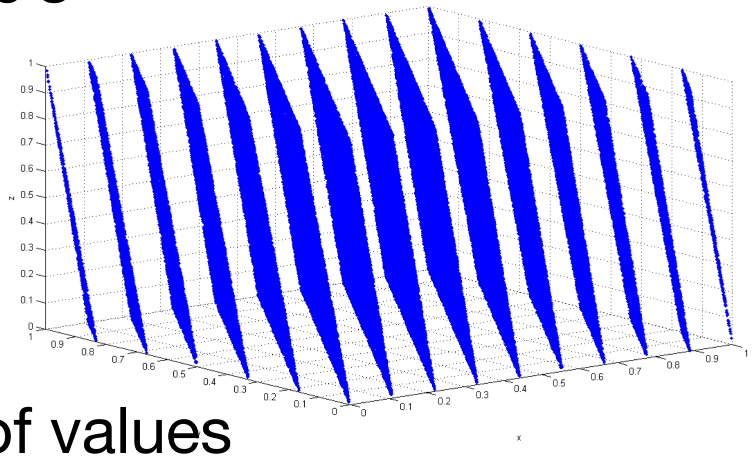
The seed determines starting point.

Wikipedia

RANDU: A catastrophically bad random number generator that was fairly widely used in the 1970's

$$\bar{u}_{i+1} = 65539 \cdot \bar{u}_i \bmod 2^{31}$$

$$u_i = 2^{-31}\bar{u}_i$$



- Constants picked for ease of hardware implementation, BUT
- Strong correlations among triples of values