Demonstration

// Show program random_LCG.cpp

$x_0 = 1$, $a = 16807$, $b = 2^{31} - 1 = 2\,147\,483\,647$

$c = 0$ (Lewis-Goodman-Miller parameters)

Also $a = 3$, $b = 32$

$c = 0, 3, 4$ observe periodicity

from Klein and Godunov

$a = 65$, $b = 65\,537$, $c = 319$

```
./random_LCG a b c n | awk '{print $1, $3}'
                         | xmgrace -p p.par -
```

or

```
./random_LCG a b c n | xmgrace -p p.par -block - -bxy 1:3
```

- series looks good - show histogram (deviations should be Gaussian)
- show correlations when plotting
  x_{n+1} vs x_n (./example.sh)

- periodicity: a short periodicity is bad
  longest periodicity is b
  periodicity kicks in as soon as a number is repeated
  e.g. 57 9 13 57 9 ...

However a truly random sequence can have repeats

randomlists.com/random_numbers random.org/sequences

- setting the seed allows us to reproduce results
- changing seed gives another realization of the simulation //

Another approach: Generalized Feedback Shift Register Method

$$x_n = x_{n-p} \oplus x_{n-q}$$

n, p and q are integers, $p > q$

→ using some approach, generate p random integers

e.g. $p = 5$, $q = 3$

$$x_1 = 4, 12, \underset{x_3}{7}, 27, 20, x_6$$

$$x_6 = x_{6-5} \oplus x_{6-3} = x_1 \oplus x_3$$

where $\oplus$ is "exclusive or" operation on bits representing x_{n-p} and x_{n-q}

e.g. $4 \oplus 7 = ?$

$4_{10} = (0100)_2$
$7_{10} = (0111)_2$

4	0 1 0 0
7	0 1 1 1
	0 0 1 1 → 3_{10}

one of, but not both

result of exclusive or

$x_6 = 3$

This is an example of "bit shuffling" - requires bit manipulation f$^{\underline{n}}$s

Must choose p and q carefully (521 and 168 work well)

In C/C++ $4 \oplus 7$ is 4^7

Fortran ieor(4,7)

show xor.cpp & NR-based programs

Can combine two generators

e.g. use 2 LCGs or 1 LCG and 1 bit shuffler

- use RNG1 to get, say, $N = 256$ RNs
- store in a list
- use RNG2 to pick a random number M between 1 and 256
- replace M^{th} element by RN generated by RNG1

→ " list shuffling "

In the end, simulation results must be independent of the RNG - can confirm this by getting same results with 2 different RNGs

Even if RNG passes statistical tests, results may be sensitive to rare events

e.g. If process is sensitive to numbers in the range 0.7112 ± 0.0001, small bias in RNG will have large impact on results

Note from Numerical Recipes

- Never use LCG
- Never use method with period $< 2^{64} \simeq 2 \times 10^{19}$
- Never use method that distinguishes between randomness of low order and high order bit
- Never use built-in C and C++ generators

UPSHOT: It's hard to be random! Pay attention to the RNG you use.

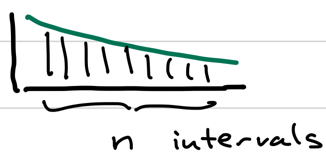
Monte Carlo (MC) Integration

Usual methods like Simpson's rule work well in
1 or 2 dimensions

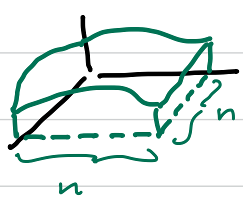
For d-dimensional integration, trapezoid rule
has error $\sim N^{-2/d}$, where N is # of integrand evaluations

For MC method, error $\sim N^{-1/2}$ – independent of
→ MC converges faster for $d > 4$
(5+ dimensions)

Classical approaches

1-D 

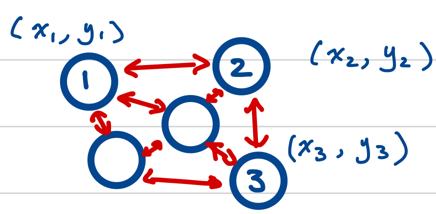
$t_{cpu} \propto n$

2-D 

$t_{cpu} \propto n^2$

d-D $t_{cpu} \propto n^d$
"curse of dimensionality"

E.g. from statistical mechanics
Consider 50 atoms on a surface
U = potential energy of whole system
$= U(x_1, y_1, x_2, y_2, \ldots x_{50}, y_{50})$



Average energy at temperature T is given by

$$\langle U \rangle = \frac{1}{Q} \int dx_1 \int dy_1 \int dx_2 \int dy_2 \cdots \int dx_{50} \int dy_{50}\, U(x_1 \ldots y_{50})\, e^{-\beta U}$$

$$\beta = \frac{1}{k_B T}$$

this is a $d = 2 \cdot 50 = 100$ - dimensional integral

For classical method $t_{cpu} \sim n^d$ (let $n = 10$)

$\sim 10^{100}$

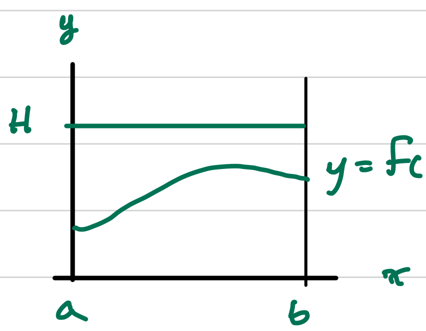
For ~ 1 GHz processor that does 6 flops per cycle $\rightarrow$ 6×10^9 operations / s

$\approx 10^{10}$ ops/s

$$\frac{10^{100} \text{ ops}}{10^{10} \text{ ops/s}} = 10^{90}\,\text{s} = 3.2 \times 10^{82}\,\text{yrs}$$

MC Approach # 1

$$F = \int_a^b f(x)\,dx$$



Generate pairs of RNs (x_i, y_i)

s.t. $a \leq x_i \leq b$

and $0 \leq y_i \leq H$ where $H > f(x)$ for all $x \in [a, b]$

probability that point (x_i, y_i) falls under $f(x)$ curve $= \dfrac{\text{Area under curve}}{\text{Area of rectangle}}$

$$\frac{n_{hit}}{n} = \frac{\int_a^b f(x)\,dx}{H(b-a)}$$