Random numbers and Monte Carlo methods

Randomness

What types of problems can we solve with the help of random numbers?

We can compute (potentially) complicated averages:

1. Where does "the average" web surfer end up? (PageRank)
2. How much is my stock portfolio/option going to be worth?
3. What are my odds to win a certain competition?

Random number generators

- Computers are deterministic - operations are reproducible
- How do we get random numbers out of a determinist machine?

Demo "Playing around with random number generators"

- Pseudo-random numbers
  - Numbers and sequences appear random, but they are in fact reproducible
  - Good for algorithm development and debugging

- How truly random are the pseudo-random numbers?

Example: Linear congruential generator

$x_o = seed$

$x_{n+1} = (a\, x_n + c\,)\ (mod\ M)$

a: $multiplier$
c: $increment$
M: $modulus$

- If we keep generating numbers using this algorithm, will we eventually get the same number again? Can we define a period?

Demo "Random numbers"

Good random number generator

- Random pattern
- Long period
- Efficiency
- Repeatability
- Portability

Random variables

We can think of a random variable X as a function that maps the outcome of unpredictable (random) processes to numerical quantities.

Examples:

- How much rain are we getting tomorrow?
- Will my buttered bread land face-down?

We don't have an exact number to represent these random processes, but we can get something that represents the **average** case.

To do that, we need to know how likely each individual value of X is.

Discrete random variables

Each random value X takes values x_i with probability p_i

for $i = 1, \ldots, m$ and $\sum_{i=1}^{m} p_i = 1$

Example:

- random variable X: value that appears on top of the dice after each roll
- X can assume the values 1,2,3,…,6
- Each value x_i has probability $p_i = 1/6$

Coin toss example

Random variable X: result of a toss can be heads or tails

X =1: toss is heads
X = 0: toss is tail

For each individual toss, x_i is 0 or 1 and each x_i has probability $p_i = 0.5$

The **expected value** of a discrete random variable is defined as:

$$E(X) = \sum_{i=1}^{m} p_i \, x_i$$

So for a coin toss:

$$E(X) = 1 * 0.5 + 0 * 0.5 = 0.5$$

Coin toss example

Let's toss a "fair" coin 1000 times, and record the number of times we get heads.

The recorded number would likely land close to the expected value 0.5.

If we run this 1000 coin toss experiment N times (let's say N = 100), the results will look like a normal distribution, with the majority of the results close to 0.5.

Texas Holdem Game

Question: for each starting pair of cards, what is the probability of winning?

- **One Game: set of 7 cards**

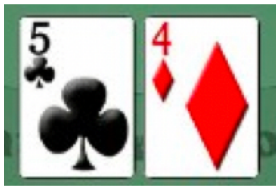
Starting hand
Opponent hand
Dealer hand
Compares the cards and decides who wins the game

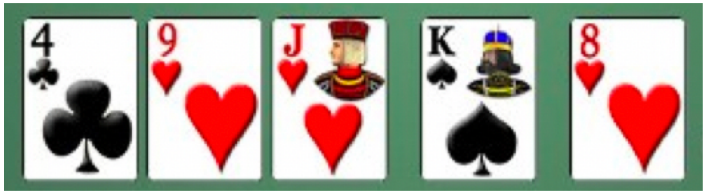
- **One numerical experiment:**

"Play" N games and record the result of each one of them

Texas Holdem Game

Question: for each starting pair of cards, what is the probability of winning?

Starting hand (deterministic variable **S**): 

Dealer hand (random variable **D**): 

Opponent hand (random variable **O**): 

Define function $best(\boldsymbol{D}, \boldsymbol{O}\ or\ \boldsymbol{S})$: gets the best hand (5 cards) out of a set of 7 cards

Function: $\boldsymbol{X} = Win(\boldsymbol{S}, \boldsymbol{O}, \boldsymbol{D})$

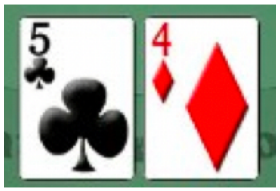
Texas Holdem Game

$$\boldsymbol{X} = Win(\boldsymbol{S}, \boldsymbol{O}, \boldsymbol{D})$$

$X = [1,0,0]$: starting hand wins

$X = [0,1,0]$: starting hand loses (opponent wins)

$X = [0,0,1]$: tie



GAME

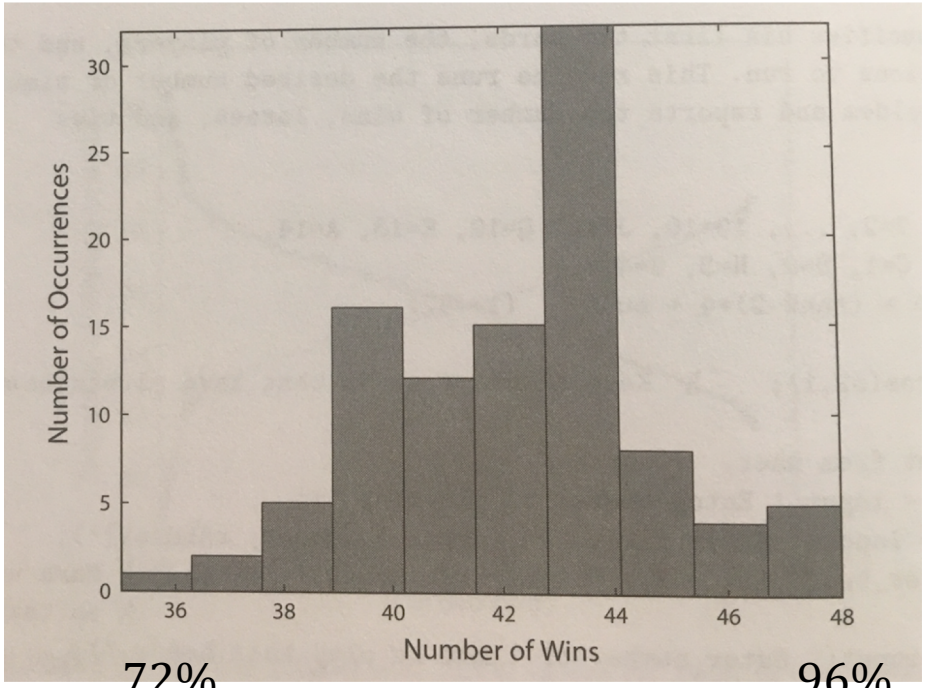
Let's say we now run 1,000 "games" with the starting hand 5 clubs and 4 of diamonds. The experiment produces 350 wins, 590 losses and 60 ties.

ODDS: W=0.35, L=0.59, T=0.06

If we run this same numerical experiment again, would we get the same results (odds)?

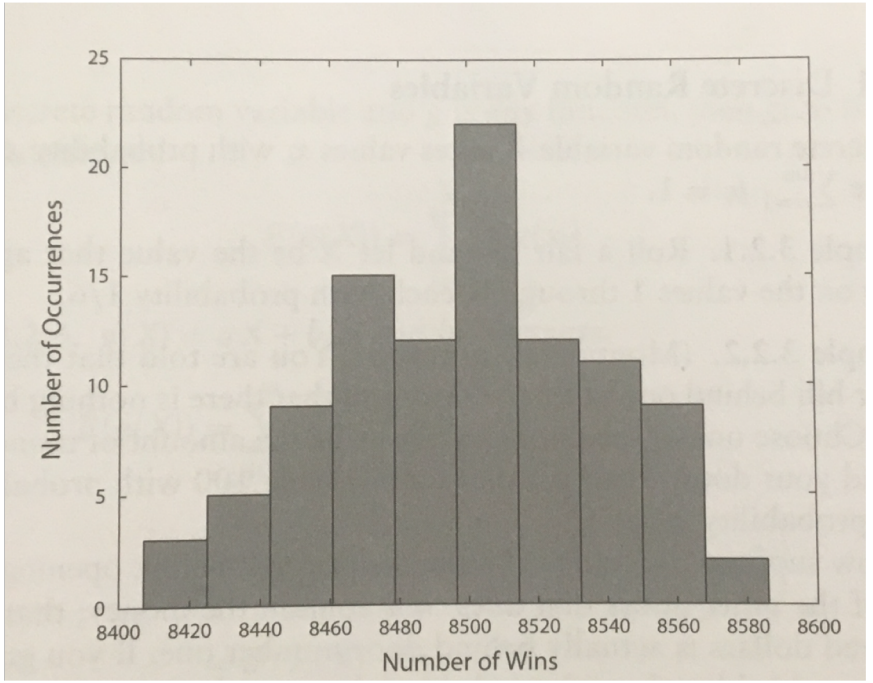
Texas Holdem Game

Starting hand: pair of aces
Plotting the number of wins for 100 numerical experiments



72% 84% 96%

50 games



10,000 games

Monte Carlo methods

- You just implemented an example of a Monte Carlo method!

- Algorithm that compute APPROXIMATIONS of desired quantities based on randomized sampling

Monte Carlo Methods

To approximate integration problems

$$\mu = \int_{x_o}^{x_1} \int_{y_o}^{y_1} f(x,y)\, dx\, dy$$

We sample points uniformly inside the domain $D = [x_o, x_1] \times [y_o, y_1]$

$$\overline{f_N} = \frac{1}{N} \sum_{i=1}^{N} f(x_i, y_i) \qquad (x_i, y_i) \sim U(D)$$

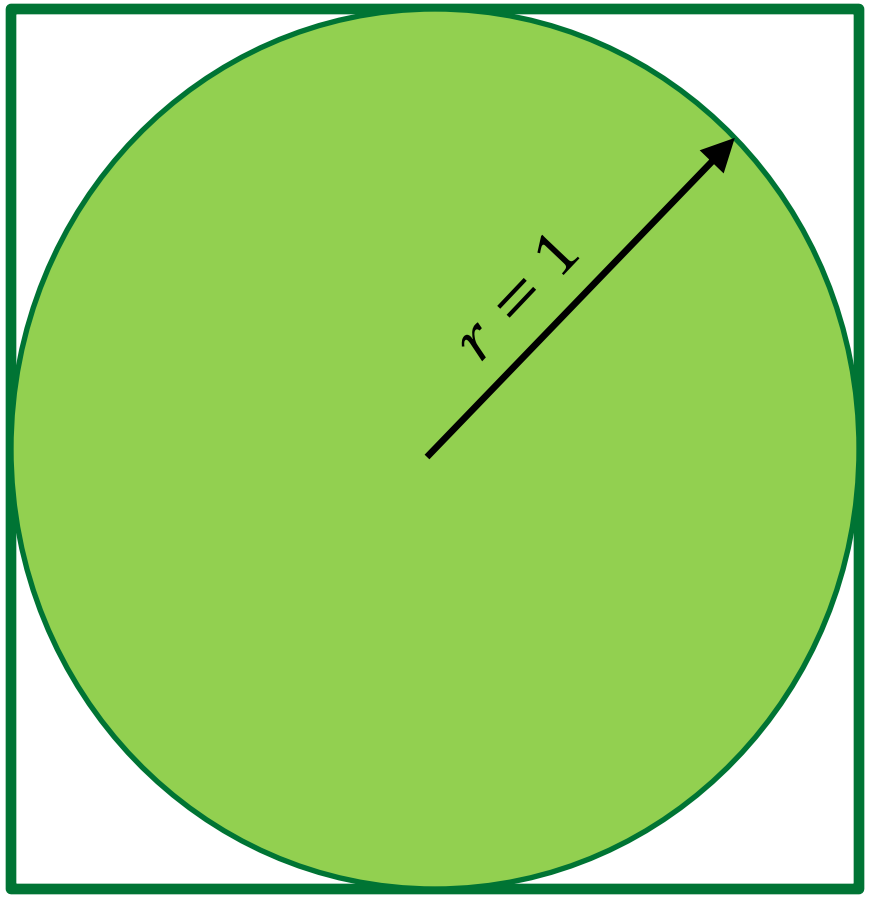
Mark the correct answer:

A) $\overline{f_N} \to \mu$ as $N \to \infty$

B) $\overline{f_N}(x_1 - x_o)(y_1 - y_o) \to \mu$ as $N \to \infty$

C) None of the above

Example: Approximate the number π



$$A_C = \pi$$

$$A_S = 4$$

$$N_C \propto A_C$$

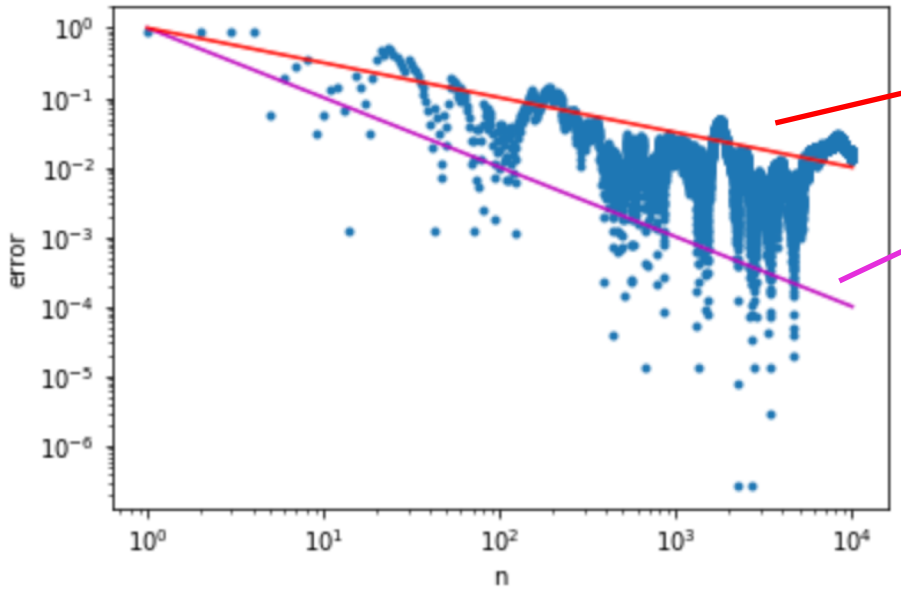
$$N_S = N \propto A_S$$

$$\frac{N_C}{N_S} \approx \frac{A_C}{A_S}$$

$$A_C \approx 4\frac{N_C}{N}$$

What can we learn about this simple numerical experiment?

- What is the cost of this numerical experiment? What happens to the cost when we increase the number of sampling points (n)?
- Does the method converge? What is the error?



$$error = O\left(\frac{1}{\sqrt{n}}\right)$$

$$error = O\left(\frac{1}{n}\right)$$

- CONS: Slow convergence rate when using Monte Carlo Methods
- PROS: Efficiency does not degrade with increase in the dimension of the problem (try to modify the demo to approximate the area of an sphere)