Lecture 9 – Monte Carlo Simulation

Dr. Jing Li
Department of Computing
The Hong Kong Polytechnic University

22&24 Mar 2022

Looking Back (Lecture 9)

- Simulations with R
  - How to generate random number (from a distribution).
  - Why and how to set random seeds.
  - Random sampling (from a given set of objects)
  - **Example**: simulating a linear model.
- **Case Study**: Analysis of PM 2.5 changes in U.S.
  - How to load data and pick up the data we want
  - How to analyze the data with graphs.
    - Examine the data of the entire U.S.
    - Probe into details of data collected by a monitor.
    - Compare results of different states.

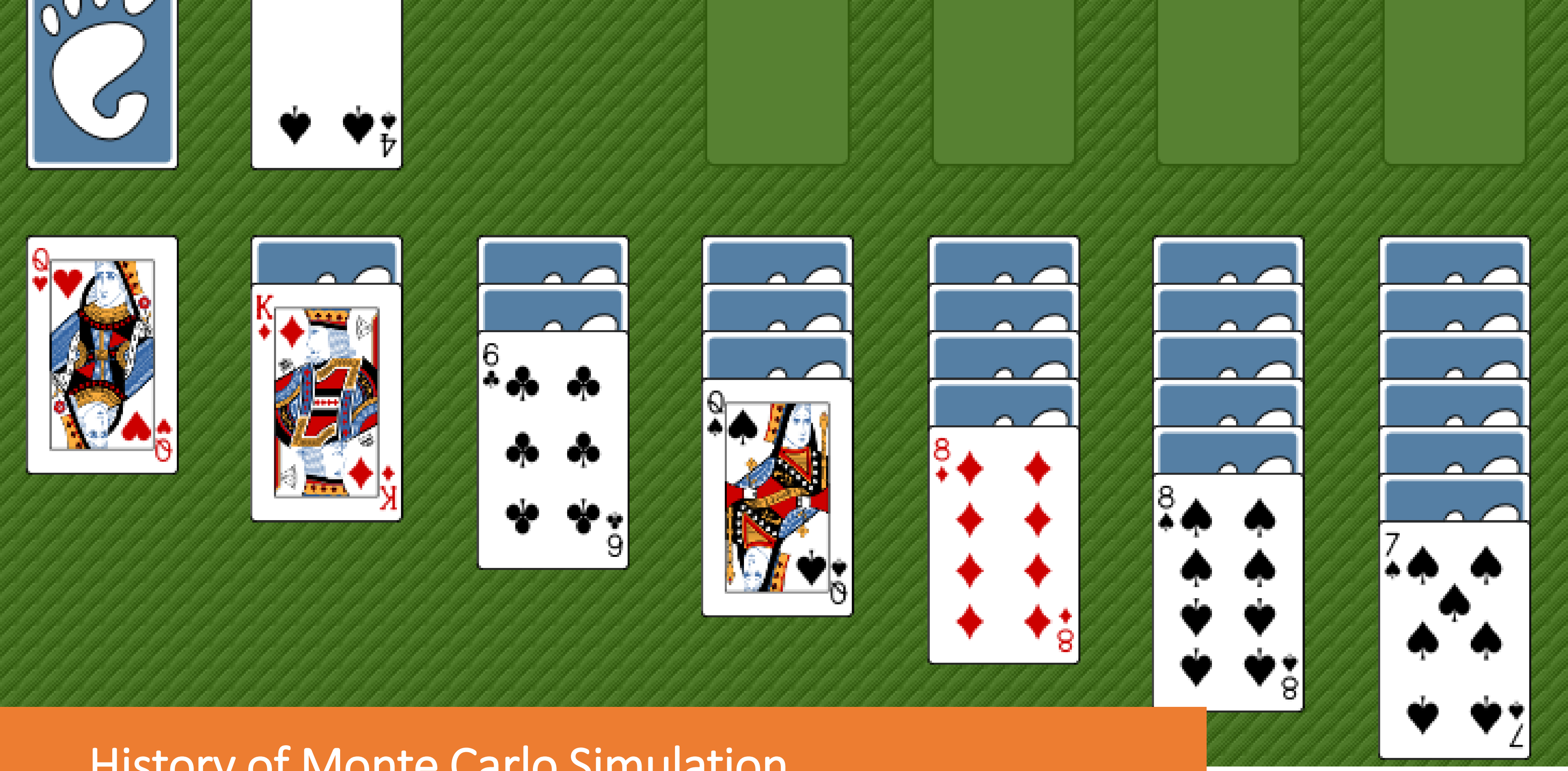
Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- Monte Carlo Simulation with R
  - Estimating π
  - Simulating Product Demand
  - Simulating Coin Flipping
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

Roadmap

- **Introduction: *History* and *Motivation***
- Key Factors in Monte Carlo Simulation
- Monte Carlo Simulation with R
  - Estimating π
  - Simulating Product Demand
  - Simulating Coin Flipping
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

Where are is it?



History of Monte Carlo Simulation

- *Stanislaw Ulam*, recovering from an illness, was playing a lot of solitaire
- Tried to figure out probability of winning and failed.
- Thought about playing lots of hands and counting number of wins but decided it would take years.
- Asked Von Neumann if he could build a program to simulate many hands on *ENIAC*.

Monte Carlo Simulation

- A method of estimating the value of an unknown quantity using the principles of ***inferential statistics***
- Inferential statistics
  - **Population**: a set of examples
  - **Sample**: a proper subset of a population
  - **Key fact**: a *random* sample tends to exhibit the same properties as the population from which it is drawn

Roadmap

- Introduction: *History* and *Motivation*
- **Key Factors in Monte Carlo Simulation**
- Monte Carlo Simulation with R
  - Estimating π
  - Simulating Product Demand
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

An example of Coin Flipping

- Given a single coin, estimate fraction of heads you would get if you flipped the coin an infinite number of times
- Consider one flip and heads on!

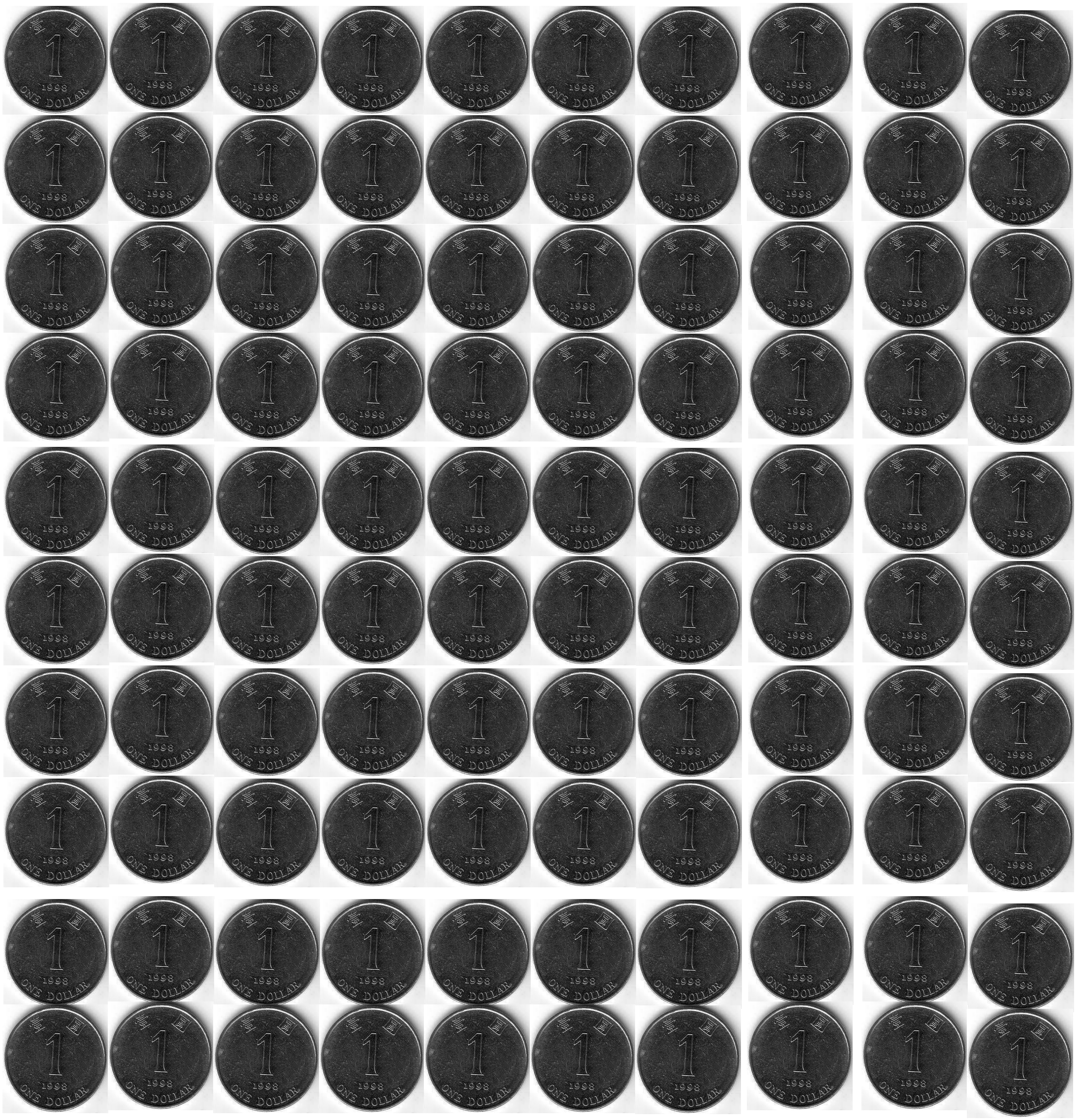
How confident would you be about answering 1.0?

An example of Coin Flipping

- Given a single coin, estimate fraction of heads you would get if you flipped the coin an infinite number of times
- Consider two flip and both with heads on!

Do you think the next will come up heads?

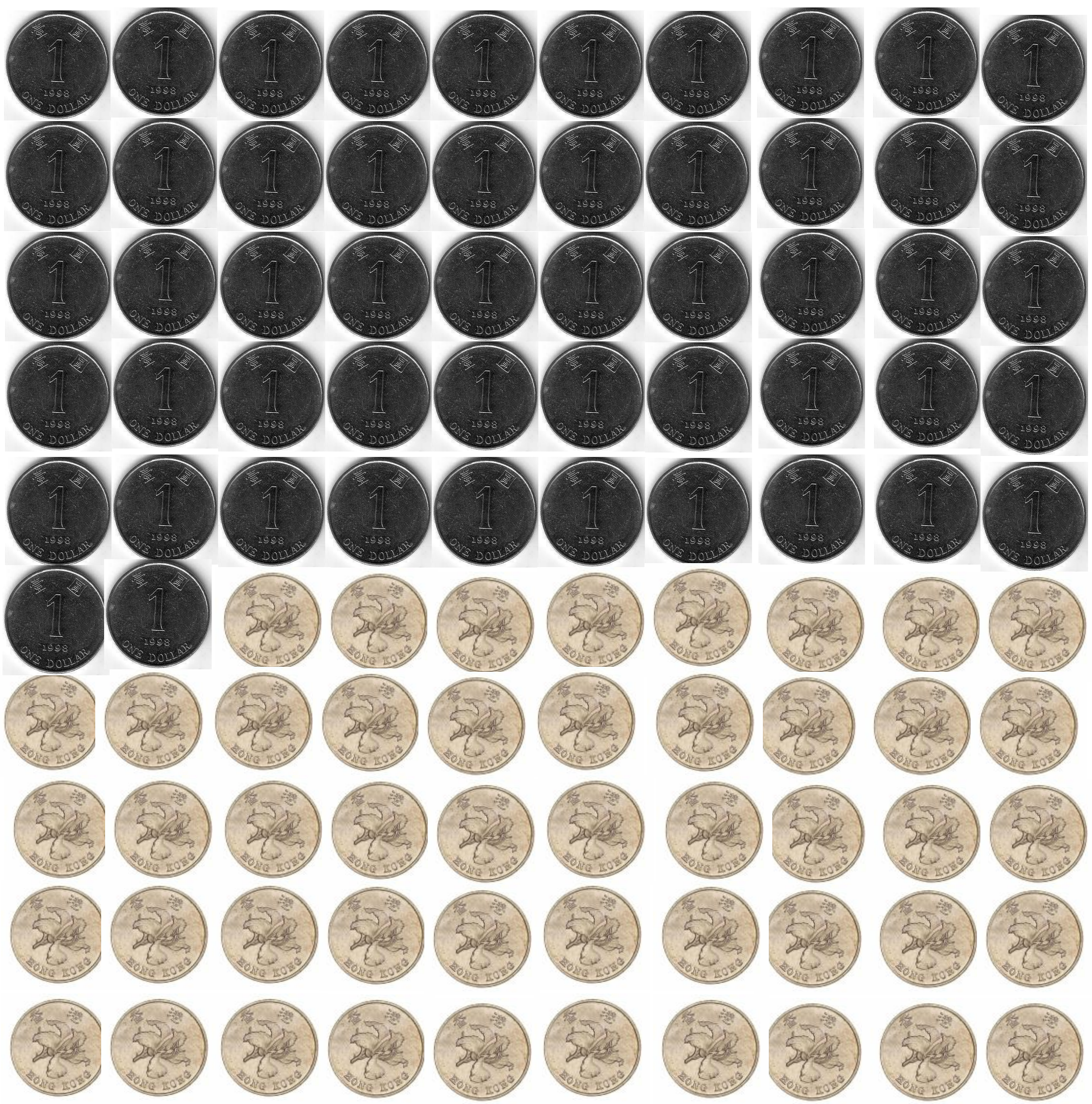
An example of Coin Flipping



Flip a coin 100 times, all with heads on.

Now do you think that the next flip will come up heads?

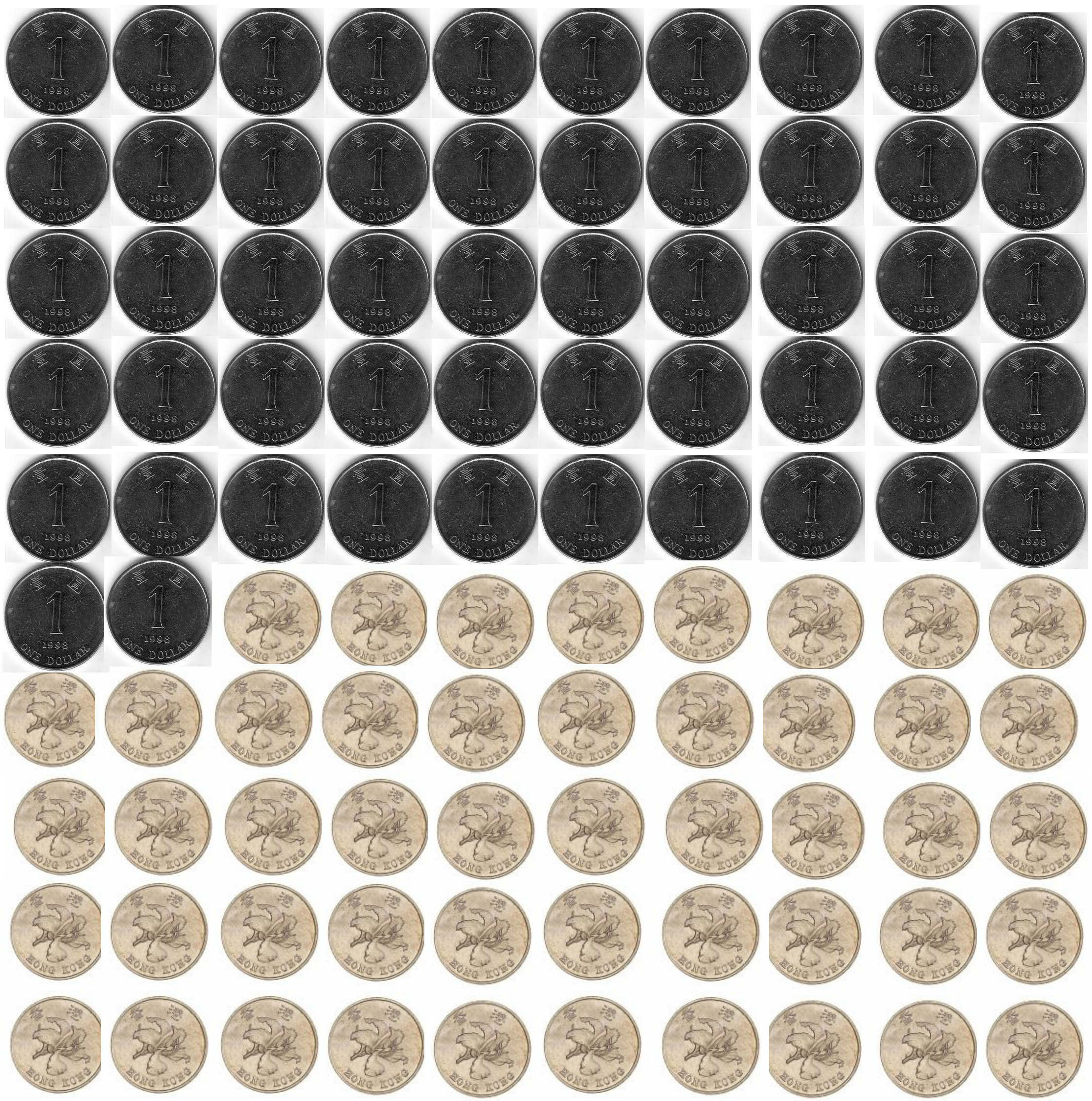
An example of Coin Flipping



Flip a coin 100 times, 52 heads and 48 tails.

Do you think that the probability of the next flip coming up heads is 52/100?

An example of Coin Flipping



Flip a coin 100 times, 52 heads and 48 tails.

Do you think that the probability of the next flip coming up heads is 52/100?

Your best guess but with low confidence!

Why the Difference in Confidence?

- Confidence in our estimate depends upon two things
  - **Size of sample** (e.g., 100 versus 2)
  - **Variance of sample** (e.g., all heads versus 52 heads)
- As the variance grows, we need larger samples to have the same degree of confidence!

Recall: probability

- *Probability* is a numerical description of *how likely an event is to occur* and or *how likely that a proposition is true*. --- From Wikipedia.
- We run a random experiment n times, during which an event A occurs m times, then we say the *frequency* of A's occurrence is $f_A = \frac{m}{n}$.
- When n is large enough, f_A will be very close to a value p, which is defined as the probability of A to occur, i.e., $\lim_{n \to +\infty} f_A \equiv P(A) = p$
  - When we toss a coin, the probability of "heads up" is 0.5

Monte Carlo Principal

- In repeated independent tests with the same actual probability p of a particular outcome in each test, the chance that the fraction of times that outcome occurs differs from p converges to zero as the number of trials goes to infinity
- ***Intuition**: if deviations from expected behaviour occur, these deviations are likely to be evened out by opposite deviations in the future.*

Law of Large Numbers (different from the one we learned in Lecture 3)

Sampling Space of Possible Outcomes

- Never possible to guarantee perfect accuracy through sampling
- **Key question**:
  - How many samples do we need to look at before we can have justified confidence on our answer?
- Depends upon **variability** in underlying distribution

Roadmap

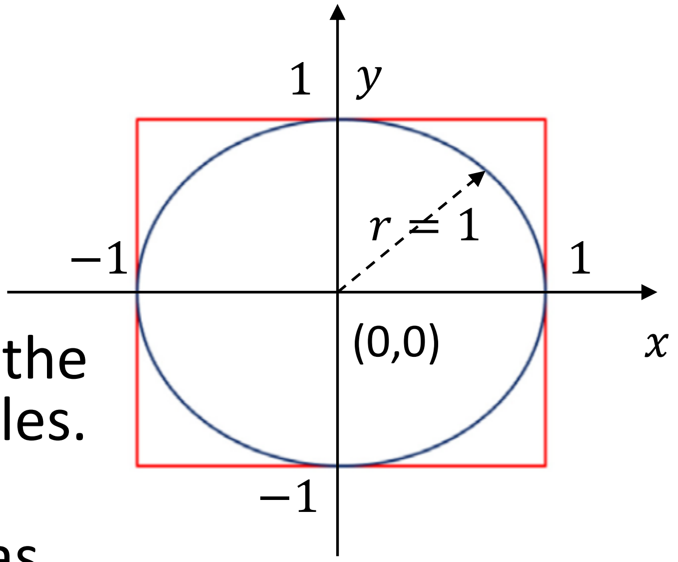
- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
  - Estimating π
  - Simulating Product Demand
  - Simulating Coin Flipping
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
  - **Estimating π**
  - Simulating Product Demand
  - Simulating Coin Flipping
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

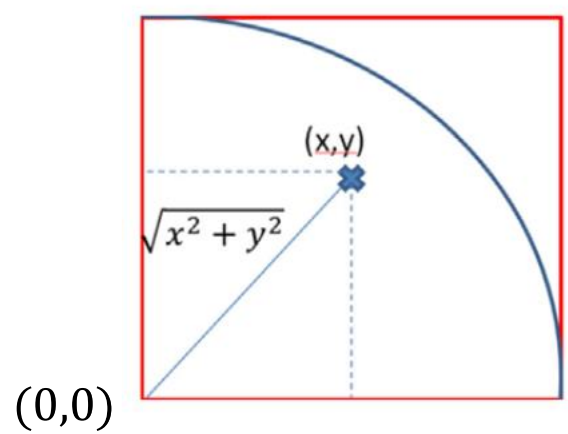
Estimating π

- What is the ratio of the area of the circle to the area of the square is?
  - $p = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4}$
- Randomly generate points (x, y)
  - $-1 \le x, y \le 1$ (let $r = 1$)
  - Count the number of objects inside the circle and the total number of samples.
  - The ratio of the two counts is an estimate of the ratio of the two areas, which is $\pi/4$.
  - Multiply the result by 4 to estimate π.



Estimating $\boldsymbol{\pi}$

- Generate 1,000 points $(x, y), -1 \le x, y \le 1$.
- Calculate the distance of each point from $(0,0)$ and determine whether each point is inside the circle.
- We may compute the ratio of samples inside the circle and all samples
- Multiply the ratio by 4 to estimate π.



Estimating π

How to get more accurate results?

```
no_of_points <- 1000
#runif samples from a uniform distribution
x<- runif(no_of_points, -1, 1)
y<- runif(no_of_points, -1, 1)

#compute the distance of each point from (0,0)
distance <-sqrt(x^2+y^2)

#boolean vector to indicate if each point is within the circle
within_circle <- ifelse(distance<1, TRUE, FALSE)

#compute proportion of points within circle/outside circle with table()
v <- table(within_circle)

#compute and print PI
pi <- v["TRUE"]/(v["TRUE"]+ v["FALSE"]) *4
print(pi)
```

table(): Count the number of unique elements (TRUE and FALSE)

output

```
TRUE
3.136
```

Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
  - Estimating π
  - **Simulating Product Demand**
  - Simulating Coin Flipping
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

Simulating Product Demand

- Suppose the demand for a product is governed by the following discrete random variable:

Demand (X)	Probability
10,000	0.1
20,000	0.35
40,000	0.3
60,000	0.25

- How to simulate daily demand for 100 days and sum them?

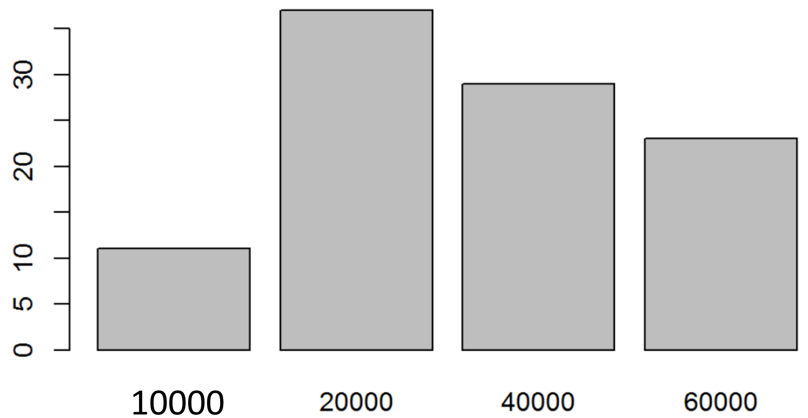
Simulating Product Demand

- How to simulate daily demand for 100 days and sum them?

Sample with replacement

```
x <- c( 10000 , 20000 , 40000, 60000)
probability <- c(0.1, 0.35, 0.3, 0.25)

demand <- sample(x, 100, replace=TRUE, prob=probability)
barplot(table(demand))
```



Sum of the demand in 100 days

```
> sum(demand)
[1] 3470000
```

Expected value: $E[X] = 3500000$

Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
  - Estimating π
  - Simulating Product Demand
  - **Simulating Coin Flipping**
  - Stock Price Prediction
  - Hits on a Website
  - Queueing System for an ATM

Simulating Coin Flipping

- Given $n = 10$ flipping of a fair coin, what is the chance that we will get $k = 4$ heads-up or less?

```r
n <- 10 # no. of coin flips
k <- 4  # no. of heads
runs <-10000 # number of trials

# one trial simulates the flipping of a coin for 10 times
trial <- function(){
  sum(sample(c(0,1), n, replace=TRUE)) # of heads
}

#conduct trials 10000 times
result <-replicate(runs, trial())
t<-table(result)
print(t)
```

One trial experiment

Output

```
## result
##    0    1    2    3    4    5    6    7    8    9   10
##   15  110  442 1167 2061 2465 2076 1118  460   73   13
```

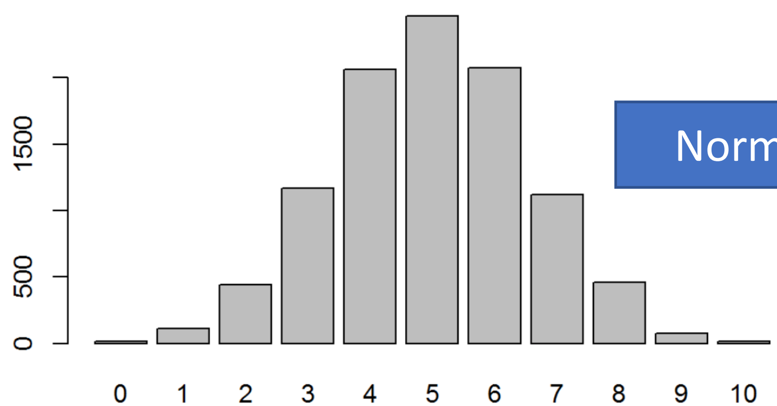
Simulating Coin Flipping

- Given $n = 10$ flipping of a fair coin, what is the chance that we will get $k = 4$ heads-up or less?

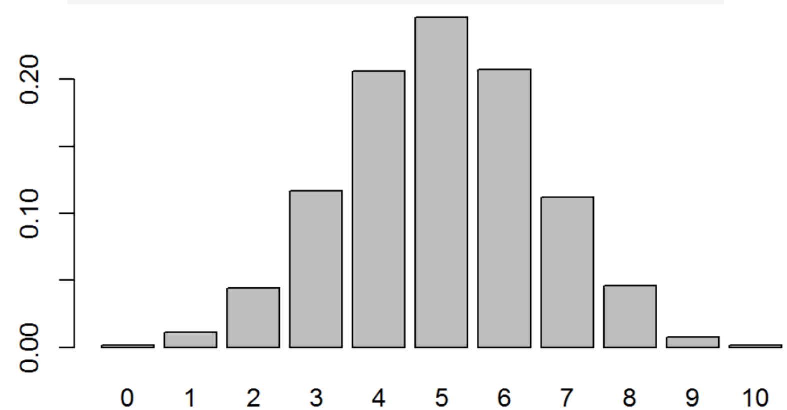
```
#conduct trials 10000 times
result <-replicate(runs, trial())
t<-table(result)
print(t)
barplot(table(result))
```

Draw the barplot to visualize the sample results

```
barplot(table(result)/runs)
```



Normalize



Simulating Coin Flipping

- ***Binomial Distribution***: The probability of getting *k* heads in *n* trials follows the binomial distribution.
  - $\binom{n}{k} p^k p^{n-k}$
- *dbinom(x, size, prob)* : Probability density function
  - ```r
    #prob(3 head out of 10 coin flips)
    dbinom(3, size=10, prob=0.5)
    ```
    ```
    ## [1] 0.1171875
    ```
- *pbinom(q, size, prob, lower.tail)* : Calculate the cumulative probability
  - **True**: $P(X \leq x)$
  - **False**: $P(X > x)$
  - ```r
    #prob(<=3 head out of 10 coin flips)
    pbinom(3, size=10, prob=0.5, lower.tail= TRUE)
    ```
    ```
    ## [1] 0.171875
    ```
  - ```r
    #prob(>3 head out of 10 coin flips)
    pbinom(3, size=10, prob=0.5, lower.tail= FALSE)
    ```
    ```
    ## [1] 0.828125
    ```
    ```

Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
  - Estimating π
  - Simulating Product Demand
  - Simulating Coin Flipping
  - **Stock Price Prediction**
  - Hits on a Website
  - Queueing System for an ATM

Stock Price Prediction

- The price of a stock today is $30
- In prior data , the price of the stock increases a factor of mean 1.001, with standard deviation of 0.005, i.e., price changes satisfy $N(1.001, 0.005^2)$
- What is the price of the stock after 365 days?
  - ```
    #generate the price change for 365 days
    days <-365
    changes <- rnorm(days, mean=1.001, sd=0.005)

    price<-cumprod(c(30, changes)) #30*1st random no.*2nd random no...
    # Plot the line chart R Base Graphics
    plot(price, type="l")
    ```

Returns a vector whose elements are the cumulative products of the elements of the argument.
    ```

Stock Price Prediction

- The price of a stock today is \$30
- In prior data , the price of the stock increases a factor of mean 1.001, with standard deviation of 0.005, i.e., price changes satisfy $N(1.001, 0.005^2)$
- What is the price of the stock after 365 days?
  -

Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte-Carlo Simulation
- **Monte-Carlo Simulation with R**
  - Estimating π
  - Simulating Product Demand
  - Simulating Coin Flipping
  - Stock Price Prediction
  - **Hits on a Website**
  - Queueing System for an ATM

Hits on a Website

- We are interested to estimate the number of hits on a certain website during a fixed time interval.
- The average rates per minute is 5
- ***Poisson distribution*** is often used to model rare events that are extremely unlikely to occur within a very short period of time or simultaneously (e.g. within 0.0001s).
  - **Examples**: arrivals of jobs to printers, arrivals of buses, occurrences of earthquake/traffic accidents/errors in software, arrival of customers to a queue.

Hits on a Website

- ***Poisson distribution*** describes the probability of a given number of events occurring in a fixed interval of time and/or space
  - These events occur independently with a known average rate of the time since the last event
  - Poisson distribution has a single parameter $\lambda > 0$, being the average number of occurrence of the considered events.
  - $P(k\ events\ in\ an\ interval) = e^{-\lambda}\frac{\lambda^k}{k!}$

Discrete or Continuous?

Hits on a Website

- We are interested to estimate the number of hits on a certain website during a fixed time interval.
- The average rates per minute is 5
- **Task**: Find the probability that there will be exactly 17 hits in the next 3 minutes.

$k = 17$

Since the events occur independently, $\lambda = 3 \times 5 = 15$.

Hits on a Website

- We are interested to estimate the number of hits on a certain website during a fixed time interval.
- The average rates per minute is 5
- **Task**: Find the probability that there will be exactly 17 hits in the next 3 minutes. $\lambda = 3 \times 5 = 15$
  - *dpois(x, lambda)* : Probability density function

```
> dpois(17, 15) #probability of 17 events with rate=15
[1] 0.08473555
```

  - *ppois(x, lambda, lower.tail)* : Cumulative probability function

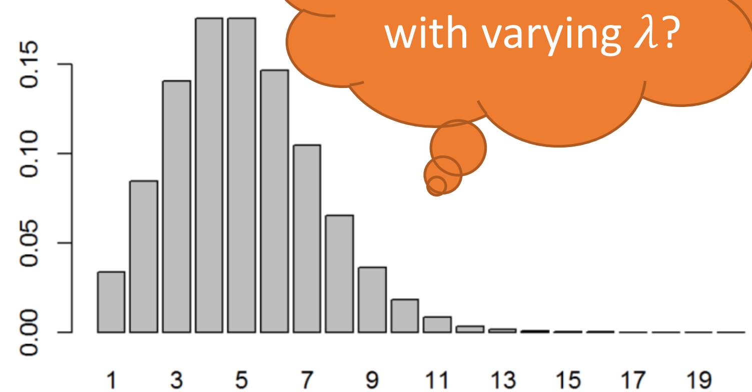
```
> ppois(17, 15, lower.tail=TRUE) #cumulative probability
[1] 0.7488588
```

Hits on a Website

- We are interested to estimate the number of hits on a certain website during a fixed time interval.
- The average rates per minute is 5
- **Task**: Create a barplot to show the probability of having *n visitors in 1 minute*.

```
rate <- 5
x <-1:20
p <- dpois(x, rate)
barplot(p, names.arg=x)
```

Output



How will the shape change with varying λ?

Hits on a Website

- What if we want to estimate the waiting time for the next hit?
- ***Exponential Distribution***: used to model the time that elapses before an event occurs, e.g., the time between two events.
- **Examples**:
  - Waiting time for the failure of a light bulb
  - Waiting time between customers coming into a store
  - Waiting time between phone calls to a hotline.
  - The length of a phone call

Discrete or Continuous?

Hits on a Website

- What if we want to estimate the waiting time for the next hit?
- ***Exponential Distribution***: used to model the time that elapses before an event occurs, e.g., the time between two events.
- Exponential Distribution VS. Poisson Distribution
  - The inter-arrival times of events in a Poisson process with rate λ is exponential and mean $1/\lambda$.
  - What is the average time between two hits (*5 hits/min*)?



Hits on a Website

- What if we want to estimate the waiting time for the next hit?

Generates n random number from exponential distribution with certain rate λ

```
#Waiting time for the next hit to the website
waitingTimeWebsite<-rexp(50, 5)
waitingTimeWebsite
```

```
##  [1] 0.09693668 0.14672284 0.04367192 0.03609941 0.06439198 0.39497901
##  [7] 0.05534583 0.45604551 0.32502948 0.15767379 0.07978570 0.68853209
## [13] 0.60078705 0.27334693 0.06470556 0.05386009 0.06896579 0.03624342
## [19] 0.26621996 0.62889533 0.06780421 0.15871193 0.15979114 0.21233654
## [25] 0.47490147 0.01712544 0.10421384 1.02176814 0.06462670 0.06672129
## [31] 0.19380114 0.03002739 0.26337777 0.27922920 0.04365630 0.02090912
## [37] 0.03615915 0.59206056 0.05625460 0.00253461 0.17629357 0.81852976
## [43] 0.00953266 0.42251408 0.09151426 0.65567584 0.35261606 0.37254870
## [49] 0.08170725 0.62902729
```

How to visualize it for result analysis?

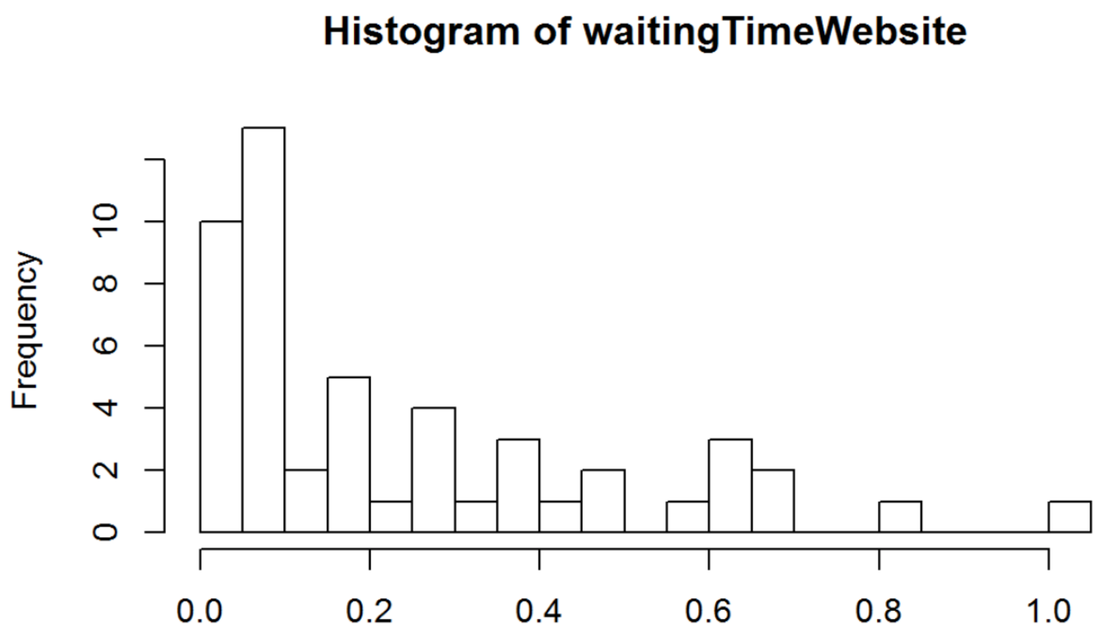
1. Barplot
2. Histogram
3. Scatterplot

Hits on a Website

- What if we want to estimate the waiting time for the next hit?

Assign the elements into 20 bins

```
hist(waitingTimeWebsite, breaks=20)
```



Roadmap

- Introduction: *History* and *Motivation*
- Key Factors in Monte Carlo Simulation
- **Monte Carlo Simulation with R**
    - Estimating π
    - Simulating Product Demand
    - Simulating Coin Flipping
    - Stock Price Prediction
    - Hits on a Website
    - **Queueing System for an ATM**

Queueing System for ATM

- We will model a simple queuing system that represents customers arriving at and using an ATM.
- Customers arrive at an ATM and perform a number of ATM transactions.
- One customer can use ATM at a time. After completing their transactions, the customers takes their cards and leave.
- When the ATM is in use, the next customers have to wait until the previous customer has left.

Queueing System for an ATM

- The first customer arrives at the ATM at 1 min and takes 1.3 minutes to complete his transactions.
- The second customer arrives at time 2 min and takes 0.5 minutes to complete his transaction.
- Quick Questions:
  - At what time will the first customer leave?
  - How long will the second customer wait before he can use the ATM?
  - At what time will the second customer leave?

Queueing System for an ATM

- A customer's waiting time depends on his arrival time and the time at which the previous customer will leave the ATM.
- There is no need for the first customer to wait.
- When determining if customer i should wait, we need to check when customer $i\ -1$ have completed his transactions when customer i arrives.

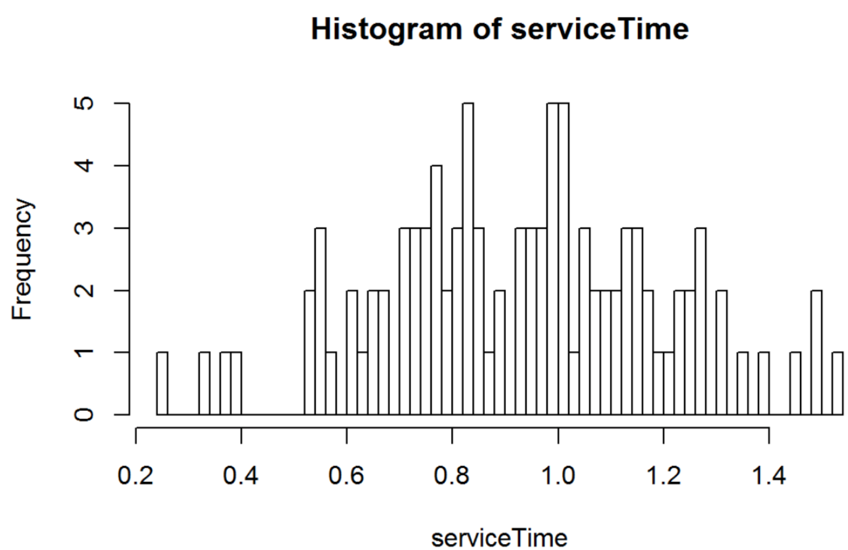
Queueing System for an ATM

- Suppose that we want to ***simulate*** *100 customers* coming to the ATM.
- The ***service time*** follows a normal distribution with a mean of 0.9 minutes and standard deviation of 0.25 minutes $N(0.9, 0.25^2)$.
- The ***inter-arrival time of customers*** follows an *exponential distribution* with a rate of 1 customer per minute ($\lambda = 1$)
- How to do the simulation?

Queueing System for an ATM

- Service times can be sampled from the normal distribution using the *rnorm(100, mean=0.9, sd=0.25)* function

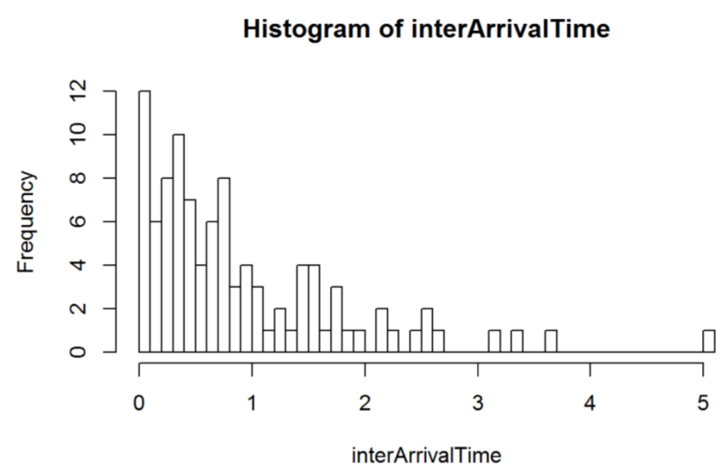
```
serviceTime<-rnorm(100, 0.9, 0.25)
hist(serviceTime, breaks=50)
```



Queueing System for an ATM

- The interarrival time of the customers can be sampled using *rexp(100,* rate=*1)*.
- This may simulate the time for customer arrived after the previous customer.

```
interArrivalTime<-rexp(100, 1)
hist(interArrivalTime, breaks=50)
```



Queueing System for an ATM

- Use the function *cumsum()* to compute the actual arrival time for the 100 customers.

```
arrivalTime<-cumsum(rexp(100, 1))
arrivalTime
```

In increasing order

```
> arrivalTime
  [1]   0.9760055   1.0091946   1.0889435   1.3630373   1.4842585   3.7292324
  [7]   5.4970826   6.0633743   6.3316084   6.5905567  10.4530757  11.9483586
 [13]  12.1467582  12.2360948  14.1136987  15.0155603  15.5390824  17.2292949
 [19]  18.2237556  19.3294594  20.2104632  20.3775642  21.1634930  24.2176364
 [25]  24.3553744  25.1767660  25.3426509  25.9323925  27.0038995  29.0078009
 [31]  29.2887315  29.4327757  32.2866256  32.3242688  32.3741418  34.1917064
 [37]  34.2460832  34.2564507  39.3157243  40.5592686  42.4860609  43.8888613
 [43]  45.7021042  46.8687582  48.4617725  48.6487968  49.5481331  49.7236396
 [49]  49.9856995  50.9172966  51.0778695  51.5505652  52.9261202  53.7540612
 [55]  54.3703905  55.7960763  56.7781005  59.3511263  60.0448332  61.6198472
 [61]  62.4108903  64.2093477  65.7146795  67.1512490  68.1908882  68.8950934
 [67]  69.5691299  70.1948541  71.4499252  72.4913475  73.5774646  75.1575037
 [73]  76.0861064  77.7882944  78.3753304  78.3861295  81.5810221  82.2756851
 [79]  83.7405187  86.2334013  86.4867768  86.5845150  87.4119343  91.2065208
 [85]  92.5837602  92.8227321  93.6039122  94.3334029 102.6586363 104.0988170
 [91] 105.2980914 105.3457986 107.3845934 107.4975768 108.4143292 108.7886023
 [97] 108.8879160 109.3589560 110.4997033 110.6833009
```

Queueing System for an ATM

- Let a_i and s_i be the arrival time and service time for customer i.
- Also, let l_{i-1} be the leaving time of customer $i-1$.
- Then, the waiting time for customer i is computed as follows:
  - If $a_i > l_{i-1}$, customer i will arrive after customer $i-1$ has left. Then, the waiting time for customer i is $w_i = 0$.
  - Else, $w_i = l_{i-1} - a_i$.
- So, $w_i = \max(0, l_{i-1} - a_i)$
- The customer i will leave at $l_i = a_i + w_i + s_i$

A slide to takeaway

- **Concept**: What is Monte Carlo Simulation?
- **Methods**: How to design Monte Carlo simulation methods to estimate a model (a probability distribution)?
- **Implementation**: How to use R programing to implement Monte Carlo simulation?