Geometric

sample space of X: 1, 2, 3, ...

Scenario: X represents the number of trials when the *first* success occurs, assuming that 1) trials are independent from each other and 2) each trial has the same success probability.
Important: Another common way to specify the geometric distribution and negative binomial distribution (next slide) is to let X be the number of *failures* instead of number of trials. This is the case in R.

sample space: 0, 1, ...

If #trials = 5
then #failures = 4

Parameter: p = success probability

Pmf: $f(x) = p(1-p)^{x-1}$, $x = 1, 2, \ldots$

$P(X=x)$ In R: dgeom(x, prob = __)

# of failures

Simulating from geometric:

In R: rgeom (n=100, prob = __) → # of failures

Negative binomial

sample space of X (# trials) $k, k+1, \cdots$

(# failures) $0, 1, 2, \cdots$

Scenario: X represents the number of trials when the *kth* success occurs, assuming that 1) trials are independent from each other and 2) each trial has the same success probability. (Geometric distribution is the special case when $k = 1$.)

Parameter: $p =$ success probability, $k =$ number of successes

Pmf: $f(x) = \binom{x-1}{k-1} p^k (1-p)^{x-k}$, $x = \underline{k, k+1, k+2,} \ldots$

In R: dnbinom(x, prob = __, size = __)

$\hookrightarrow$ # of failures

Simulating from ~~geometric~~:

negative binomial:

rnbinom(n=100, prob = __, size = __)

Example

- If the probability is 0.4 that a person exposed to a certain contagious disease will catch it, what is the probability that the fifth person exposed to it will be the first to catch it?

Let X be # of people exposed when the first person caught it.

$X \sim$ geom (prob = 0.4) $P(X=5)$ → in R dgeom(4 , prob = 0.4) = 0.05184

- Suppose 45% of a given population support a presidential candidate A and you conduct a survey by randomly sampling from the population. What is the probability that the 10th person you survey is the 3rd person to support the candidate?

Let X be # of people surveyed when you have found the 3rd supporter

$\sim$ neg. bin (k = 3, prob = 0.45)

failures

$P(X=10)$ → in R dnbinom(7 , prob = 0.45, size = 3)

= 0.0499

Poisson

Scenario: X represents the number of time an event occurs during a fixed time (or space) interval, assuming 1) each event occurs independently and 2) event occurs at a constant rate.

Parameter: $\lambda =$average number of occurrences during the fixed interval

Pmf: $f(x) = \frac{\lambda^x \exp(-\lambda)}{x!}$, $x = 0, 1, 2, \ldots$

$P(X = x) =$ (in R) $=$ dpois (x, lambda= ___)

Simulating from Poisson:

rpois (n = 100, lambda = ___)

Example

- Suppose, on average, 20 customers arrive at a ice cream shop per hour on a Friday night. The number of customers arriving in the next 30 min can be modeled by:

$\text{poisson}(\lambda = 10)$

- Suppose you get spam emails at a constant rate of 5 emails per day. What is the probability that you get fewer than 3 spam emails tomorrow?

X = #emails you get tomorrow

$\sim \text{poisson}(\lambda = 5)$

$P(X < 3) = P(X=0) + P(X=1) + P(X=2)$

sum(dpois(c(0,1,2), lambda=5))

Poisson $\approx$ Bin if n is large and p is small

ex 2% click rate. $n = 1000$ people.

$\# \text{Clicks} \sim \text{Bin}(n = 1000,\ p = 0.02) \approx \text{Poisson}(\lambda = 20)$

Verify Probabilities are similar for $x = 15, \cdots, 25$

```
dbinom( c(15:25), prob=0.02, size = 1000)
vs dpois( c(15:25), lambda = 20)
```

to get $P(X=15)$

$P(X=16)$

$\vdots$

$P(X=25)$

DATA 252/DATA 551
Modeling and Simulation

Lecture 3: Review of loops

February 3, 2020

Basic idea

Loops are used to repeat a chunk of code.

for loop Repeat for a *fixed* number of time

while loop Repeat, *until* a certain condition is met

Open RStudio or RStudio Cloud, try the following code.

```
a=2
for(i in 2:10){  #i is created after this step
  a=a+3
  print(paste('Step',i,'value of a:',a))
}

while(a < 40){
  a=a+3
  print(paste('Step',i,'value of a:',a))
}
```

for loop

Ingredients of a **for loop**: index and code chunk. For instance, the index can be $i = 1, 2, 5$, then the code chunk will run with $i = 1$, and again with $i = 2$, and again with $i = 5$.

Basic structure in R:

```
for( i  in  ___ ){

    code chunk

}
```

"Fake coding" examples

1. Repeatedly do something for 10 times.

```
for ( i   in   1:10 ) {
        do  something
  }
```

2. Repeatedly run a simulation for 1000 times, storing the result each time.

```
nsim = 1000
result = rep ( NA, nsim )    -> create an empty vector of length nsim
for ( i  in  1 : nsim) {
     do  simulation
     result [i] = result of simulation
}
```

"Fake coding" examples

3. Extract 3rd, 6th, 9th, etc. cat from a large dataset; for each cat, run a function `weighCat( )`; store the weights in a new vector.

```
result = numeric( )        # create an empty vector
for( i  in  seq. start = 3, end = #cats, by a jump of 3 ){
        Add  weighCat( ith cat)  to  result

}
```

Caution: Don't do result[i] = weighCat(i) because i = 3, 6, ...

4. In a large dataset of cats, weigh and store the weights of the female cats.

```
result = numeric(  )
for (i  in  1: #cats){
        if ( ith cat  is  female) { Add  weighCat(ith cat)  to result }


}
```

while loop

Ingredients of a **while loop**: initialization, condition and code chunk. Code chunk will run as long as the condition is met.

Basic structure in R:

```
while ( condition ) {
    code chunk
}
```

R checks condition

$\downarrow$ T $\downarrow$ runs code

$\searrow$ F $\rightarrow$ Stop

while loop

It's important to *properly update* objects involved in the while loop condition. It is highly recommended to set a safety upper bound for how long you allow the while loop to run.

```
Count =1
count max = 10000

while ( Condition  & count < countmax) {

  count = count +1

}
```

"Fake coding" examples

1. Play a game until you win 5 dollars.

```
winning=0  < add:  count=1 ;  countmax=1000
while ( winning < 5 & count < countmax ) {
    keep. playing
    winning = winning + current game result  -> update step
}  < add  count = count+1
```

2. Play a game until you win 5 or lose 100 dollars.

```
winning=0  < add:  count=1 ;  countmax=1000
while ( winning < 5 & count < countmax  add & winning > -100 ) {
    keep. playing
    winning = winning + current game result  -> update step
}  < add  count = count+1
```

Exercise

Use any language and *loop structures* to do the following.

1. Print the sentence "Hello, world" for 10 times.
2. Print a sequence of texts: "1 moose 4 legs", "2 moose 8 legs", ..., "30 moose 120 legs."
3. Create a (character) sequence: $1, 2, p, 4, 5, p, 7, ..., p, 100$. Replace the number with "p" if the number is a multiple of 3 or if the number contains 3 (like 13). Verify that the resulting vector has 45 p's.
4. Calculate and print a sequence of numbers: $1\sqrt{2}$, $2\sqrt{3}$, $3\sqrt{4}$, ... as long as the last number is smaller than 2000. What is the last number? (1992.304)
5. Create a vector that continues to generate and store the *Fibonacci sequence*: $1, 1, 2, 3, 5, 8, 13, \ldots$, as long as the last number is smaller than 10000. Verify that the resulting vector has mean 885.5.

If you are done with the above, write code to implement your simulation designs for homework 1: problems 2, 4, and 5.