Simulation in R:

**Random Numbers**

1. **Rnorm**: generates normal random variables with a given mean and st dev (defaults to a mean of zero and a st dev of 1)
   1. Formatted as rnorm(n, mean = 0, sd = 1) where n is number of numbers to be generated
2. **Dnorm**: evaluates the normal probability density (with a given mean/st dev) at a point or vector of points
   1. Formatted as dnorm(x, mean =0, sd = 1, log = FALSE)
   2. Log = FALSE is because we can set it to evaluate on the log scale of the density if we desire
3. **Pnorm:** evualuates the cumulative distribution function for a normal Distribution
   1. Formatted as pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
   2. If Φ is the cumulative distribution function for the Normal Distribution, then pnorm(q) = Φ(q) and qnorm(p) = Φ-1(p)
   3. Log.p = FALSE is because we can set it to evaluate on the log scale if we desire
   4. Lower.tail = the part that goes to the left (from, say the value x at the sd of -infinity to the value q specified). Setting to FALSE calculate from the value q to the value x at the standard deviation of +ininity
4. **Qnorm:** calculates the quantiles for the normal distribution.
   1. Formatted as qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
   2. Log.p = FALSE is because we can set it to evaluate on the log scale if we desire
   3. Lower.tail = the part that goes to the left (from, say the value x at the sd of -infinity to the value p specified). Setting to FALSE calculate from the value p to the value x at the standard deviation of +ininity
5. **Rpois**: generates random Poisson variables at a given rate
   1. Nice because it can create random integer
6. **Distribution Functions**: have 4 functions associated with with
   1. Functions prefixed with a “d” stand for density functions
   2. Functions prefixed with an “r” create random numbers
   3. Functions prefixed with a “p” calculate the cumulative distribution function
      1. In other words, what is the probability that a number “p” is less than or equal to a given probability, given a mean and st dev
   4. Functions prefixed with an “q” calculate the quantile function - e.g. “qnorm,” “qpois”

**Example of Generating random numbers:**

* X <- rnorm(10) # gives 10 random normal variables with a mean of 0 and a st dev of 1
* Y <- rnorm(10, 20, 2) #gives 10 random normal variables with a mean of 20 and a st dev of 2
* summary(x) and summary(y) will give the min, max, median, and 1st and 3rd quantile

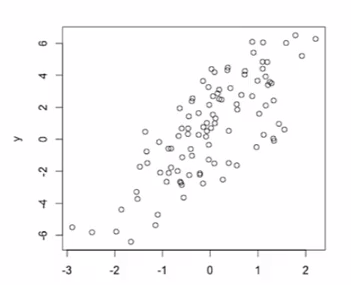
**Setting a seed**

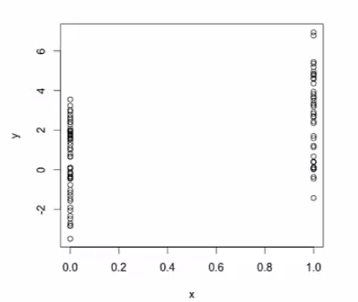
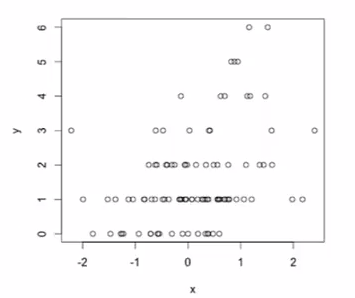
1. You can set a seed in order to get the same “random” variables if you’re running a simulation that needs to be reproduced
   1. This implies the random variables are not “random,” but computationally they’re close enough.
   2. Command: **set.seed(1)**

X <- rnorm(5)

**Simulating a Linear Model**

1. **Generating Random numbers from a linear model**
   1. Example: Where *B0* and *B1* are assigned random numbers and is a vector containing rnorm(100,0,2). For this example, *B0* is set to 2 and the slope *B1*is set to 0.5. We’ll look at *x* over the standard normal distribution.
   2. This will give us a roughly linear set of data, as shown in the scatter plot



1. Generating Random Numbers using binary random variables (such as gender, treatment groups vs control, etc)
   1. Example: use , but set x’s range to be binomial random variables, such as (0.5 means there is an equal chance of 0’s or 1’s)
   2. This gives us data that is different  
      
   3. Generating Random Numbers using a Generalized Linear model, such as a Poisson model (used for count variables as opposed to continuous variables)
      1. Example: let and and set x to be x = rnorm(100). The exponential gives the mean of the log in the Poisson
      2. This data will be discrete steps when plotted:  
         

**Random Sampling**

1. Use the sample function to draw numbers randomly from a set of scalars in R.
   1. Example: **sample(1:10, 4)** gives 4 random variables from the vector 1-10
   2. Example: **sample(letters, 5)** gives 5 random letters from the alphabet
   3. Example: **sample(1:10)** gives all 10 numbers back in a random order
   4. Example: **sample(1:10, replace = TRUE)** gives back 10 numbers where the numbers may be repeated
   5. Note: set.seed will work on the sample function
2. Other standard distributions built into R:
   1. Normal
   2. Poisson
   3. Binomial
   4. Exponential
   5. Gamma