Problem Set 2

Ariel Huckabay, Kendall Weistroffer

January 25, 2018

These questions were rendered in R markdown through RStudio (https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf, http://rmarkdown.rstudio.com).

Please complete the following tasks regarding the data in R. Please generate a solution document in R markdown and upload the .Rmd document and a rendered .doc, .docx, or .pdf document. Your work should be based on the data's being in the same folder as the .Rmd file. Please turn in your work on Canvas. Your solution document should have your answers to the questions and should display the requested plots.

For use in the problems 1, 2, 3, and 4 below, generate a vector of 100,000 values from the binomial distribution with n=20 and p=0.5 using the seed 7654321.

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --
```

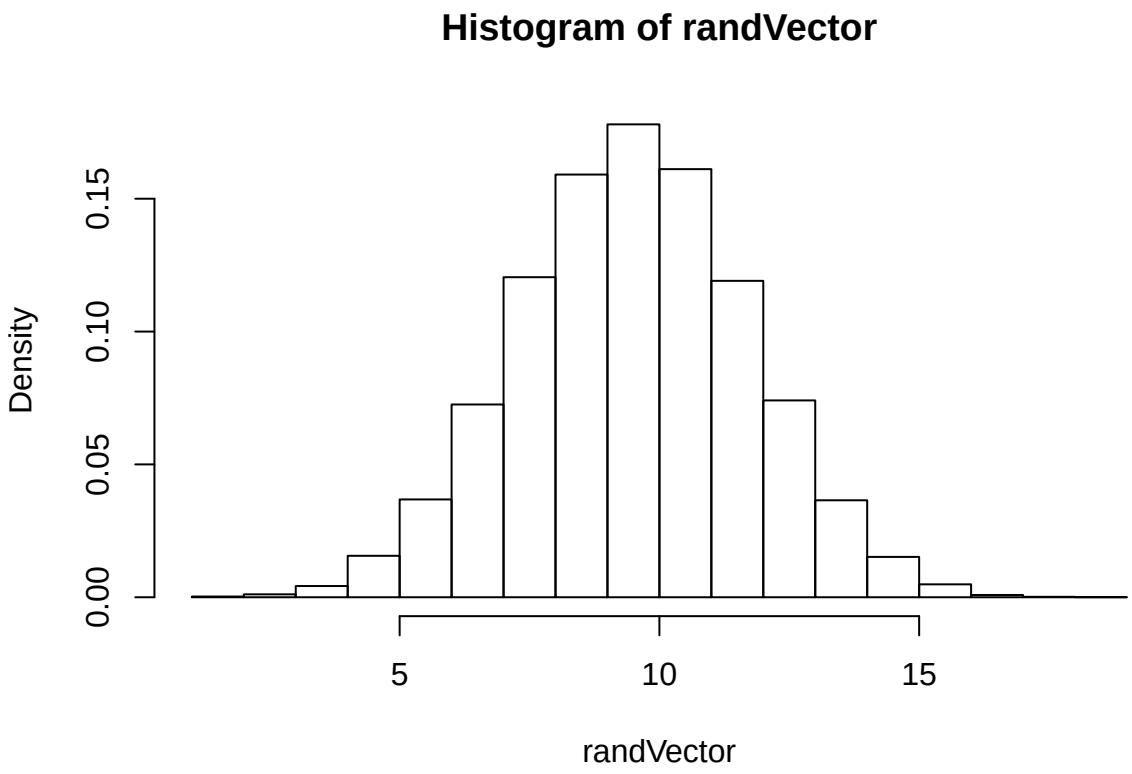
```
## v ggplot2 2.2.1     v purrr   0.2.4
## v tibble  1.4.1     v dplyr   0.7.4
## v tidyr   0.7.2     v stringr 1.2.0
## v readr   1.1.1     v forcats 0.2.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(knitr)

set.seed(7654321)
randBinom <- rbinom(100000, 20, .5)
randVector<-vector("numeric", 100000)
randVector = randBinom
```

1. Present a histogram of the results using the techniques in distributions.Rmd.(10 points)

```
hist(randVector, prob=TRUE)
```

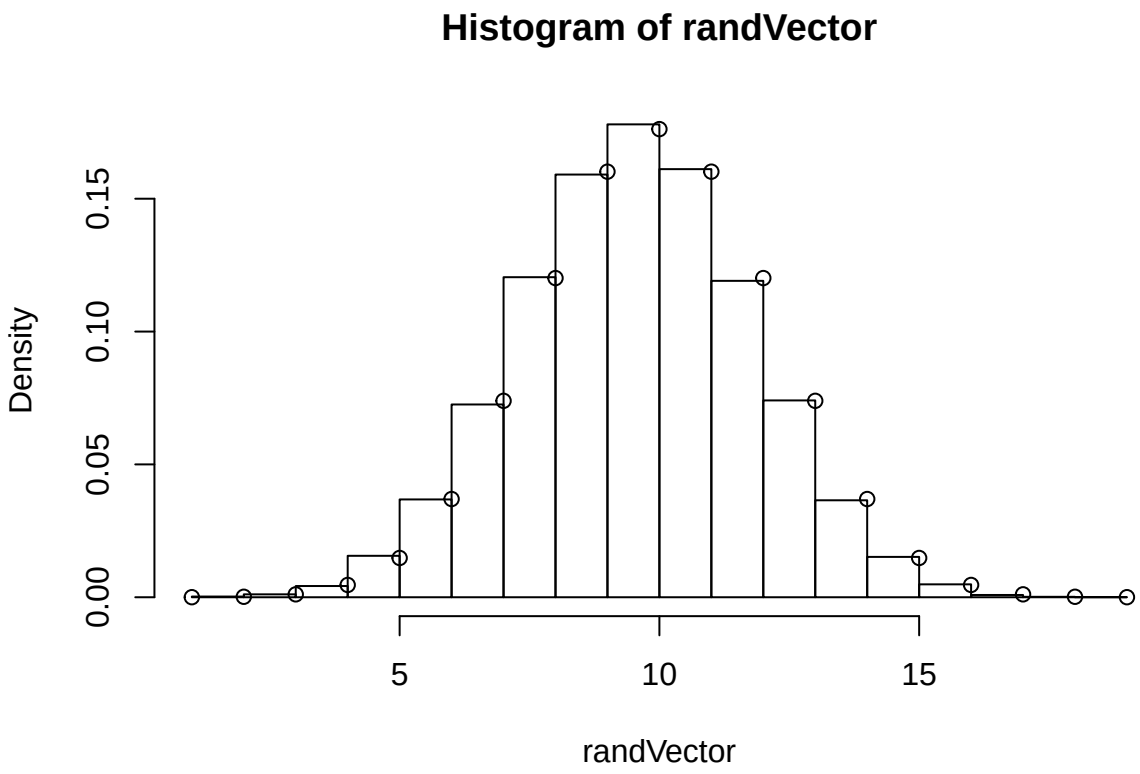


2. Discuss the appearance of the histogram in relation to the probability density function of the binomial distribution with n=20 and p=.5. (10 points)

Response

The histogram conforms to the shape of the probability density function of this binomial distribution. It is plotted as points over the histogram. This is what one would expect from binomial distribution.

```
rMean <- mean(randVector)
stdDev<-sd(randVector)
NormPlot<-dbinom(0:20, 20, 0.5)

hist(randVector, prob=TRUE)
points(0:20, NormPlot)
```



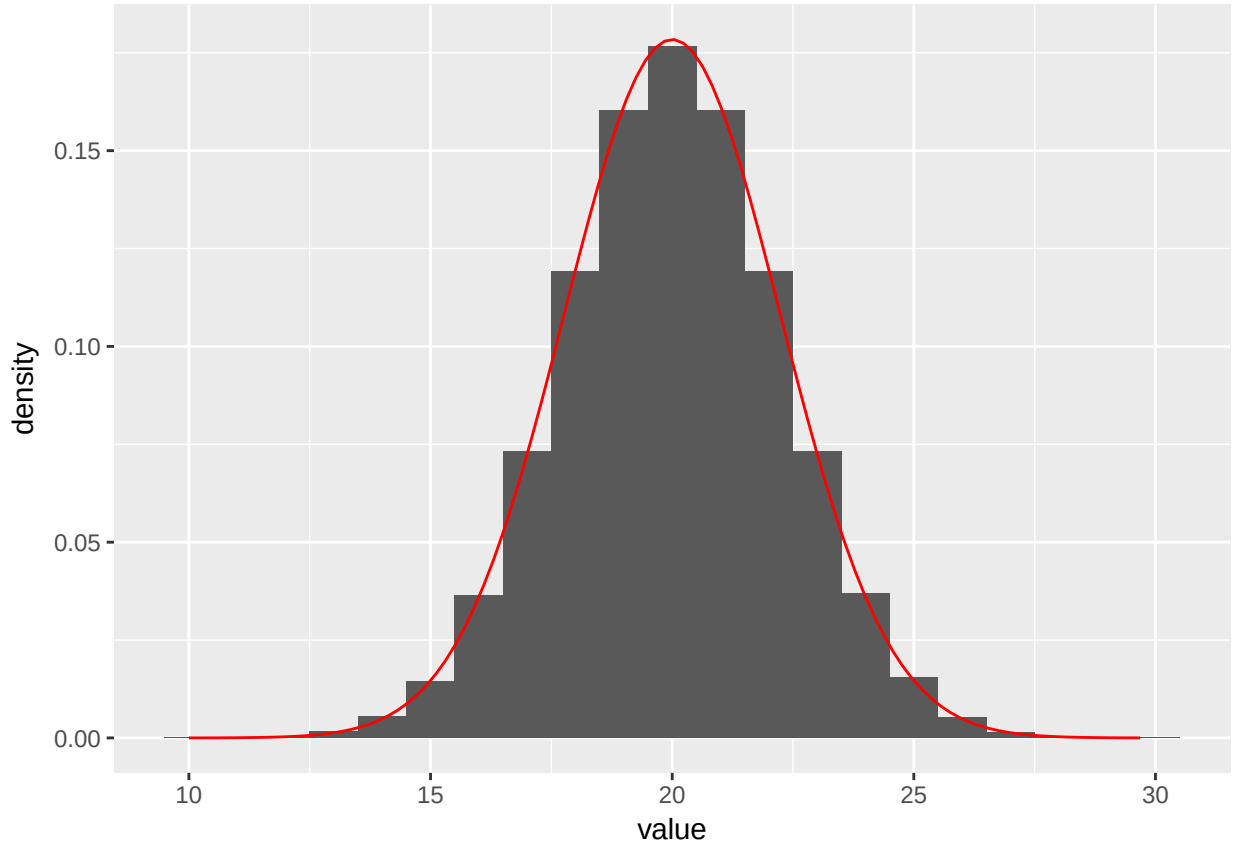
The histogram approximates a normal distribution (bell-shaped curve). Half of the values fall beyond the mean of 10.00213, and half fall beore it by visual inspection.

3. Create a plot with the histogram from above with the density of the normal distribution with mean equal to 20 and sigma equal sqrt(20*.5*(1-.5)). Do this binomial distribution and this normal distribution appear related? (10 points)

Response

Both of these historgrams have a bell-curve shape. The plot above illusrates the close relationship.

```
NormPlot= rnorm(100000, 20, sqrt(20 *0.5*(1-0.5)))

NormPlot <- data.frame(value = NormPlot)

g<- ggplot(NormPlot, aes(x=value)) +
        geom_histogram(aes(y = ..density..),binwidth=1) +
        stat_function(fun = dnorm, colour = "red",args = list(mean =20, sd = sqrt(20*.5*(1-.5))))
g
```



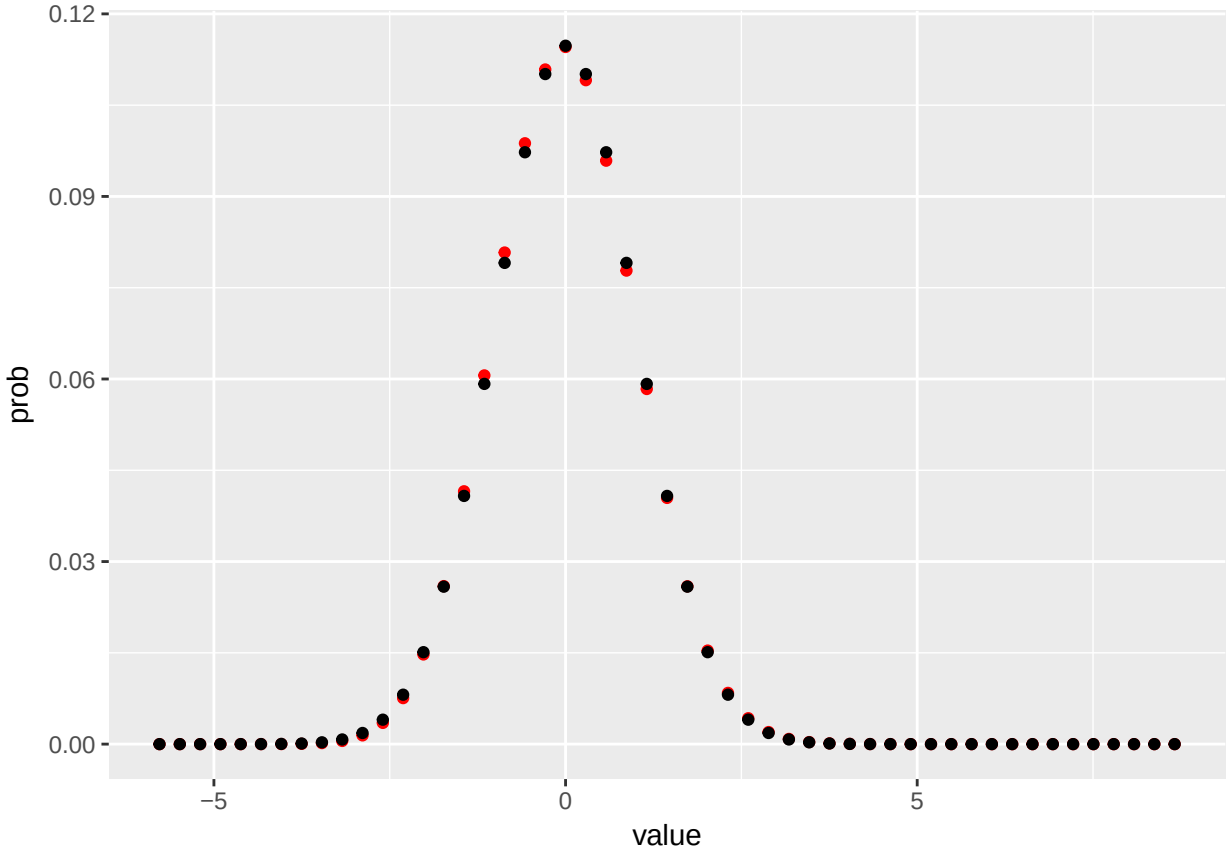
4. Now consider the probability space generated by applying the random variable X with X(y)=(y-50*.4)/sqrt(50*.4*(1-.4)) to the Binomial distribution with size=50 and probability=.4 Please plot the density of this distribution. Also, setting d=1/sqrt(50*.4*(1-.4)), plot the points with x-values equal to the outcomes of the new distribution and y-values equal to the probability that a sample from the standard normal distribution lies between x-d/2 and x+d/2 for each of these x's. Do the two distributions seem to be closely related? (10 points)

Response

Inspecting the graph, these two distributions seem very related with a nearly identical shape. The distribution plotted in black is from the result of aplying the function X, and the one in red is the result of applying the function pNormFun which finds the probability of falling between x+d/2 and x-d/2.

```
#set d
d=1/sqrt(50*0.4*(1-0.4))

#random variable

X<-function(x){(x-50*0.4)/(sqrt(50*0.4*(1-0.4)))}
pNormfun<-function(x, d){
  abs(pnorm(x+d/2) - pnorm(x-d/2))
}
var1 <- X(0:50)
probVar1 <- dbinom(0:50, 50, .4)
var1DF <- data.frame(value = var1, prob = probVar1)
pNormVec<-sapply(var1, pNormfun, d=d)
```

```
pNormDat<-data.frame(pNormVec)
g <- ggplot(var1DF, aes(x = value, y = prob)) + geom_point(colour="red")


#apply pnorm to each value in var1 +- d/2
#loop through in vector form, apply pnorm to a vector
#feed pnorm a vector as an x argument, plot points

pNormVec<-sapply(var1, pNormfun, d=d)
pNormDat<-data.frame(pNormVec)
g1<-g + geom_point(data = pNormDat, aes(x=var1, y=pNormVec))
g1
```

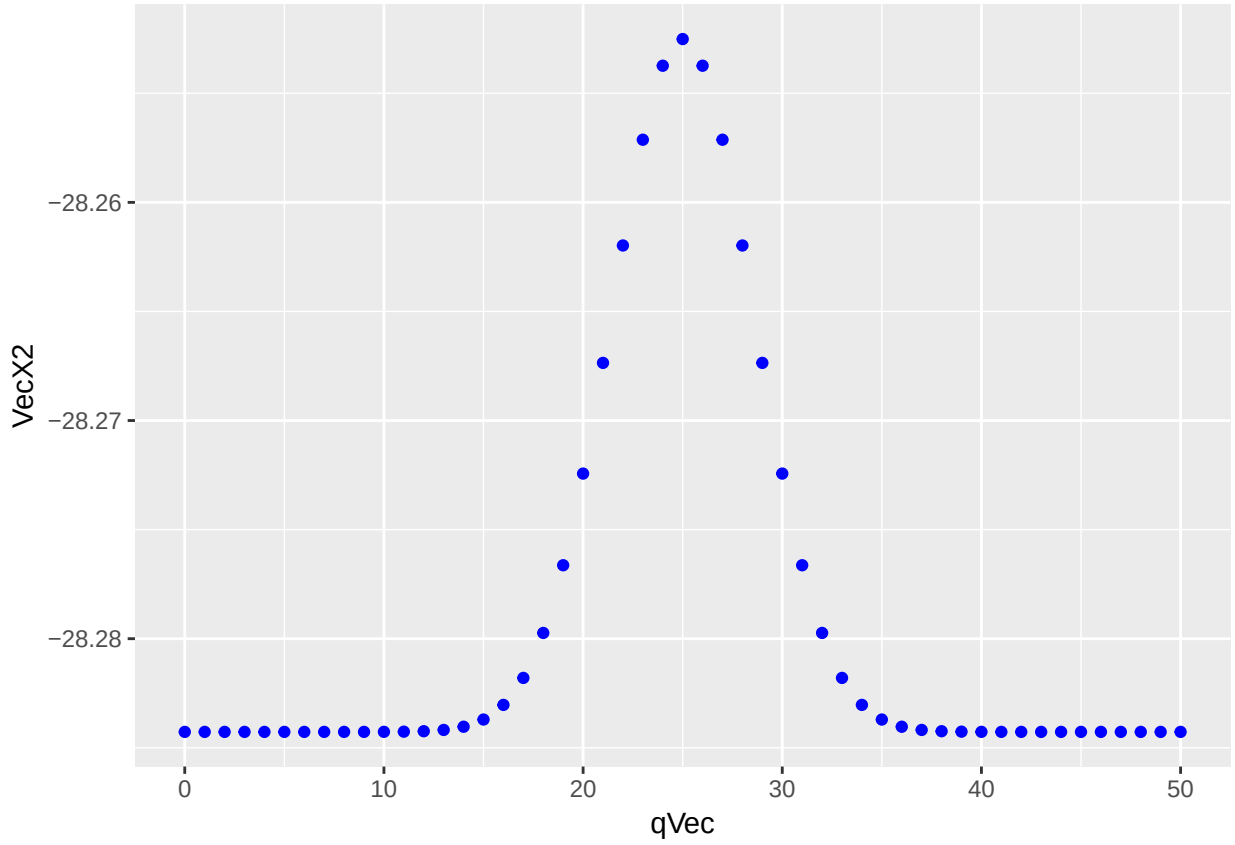


5. For some values of size s and probability p, the standard normal distribution is a good approximation to the distribution that arises from applying the random variable X defined by X(y)=(y-s*p)/sqrt(s*p*(1-p)) to the binomial distribution with size s and probability p. For which of the following pairs (s,p): (50,.5), (50,.1), (10,.1), (50,.01),(5000,.01) does that seem to be the case? Please support your conclusion with visualizations such as the one in question 4. You may use density functions without any sampling. You may find it helpful to extend the x-values for the normal distribution to be symmetric around zero using z<-unique(c(x,-x)), for example. Also, you can control the limits of the x-axis with g<-g+xlim(-3,3), say (10 points)

Response

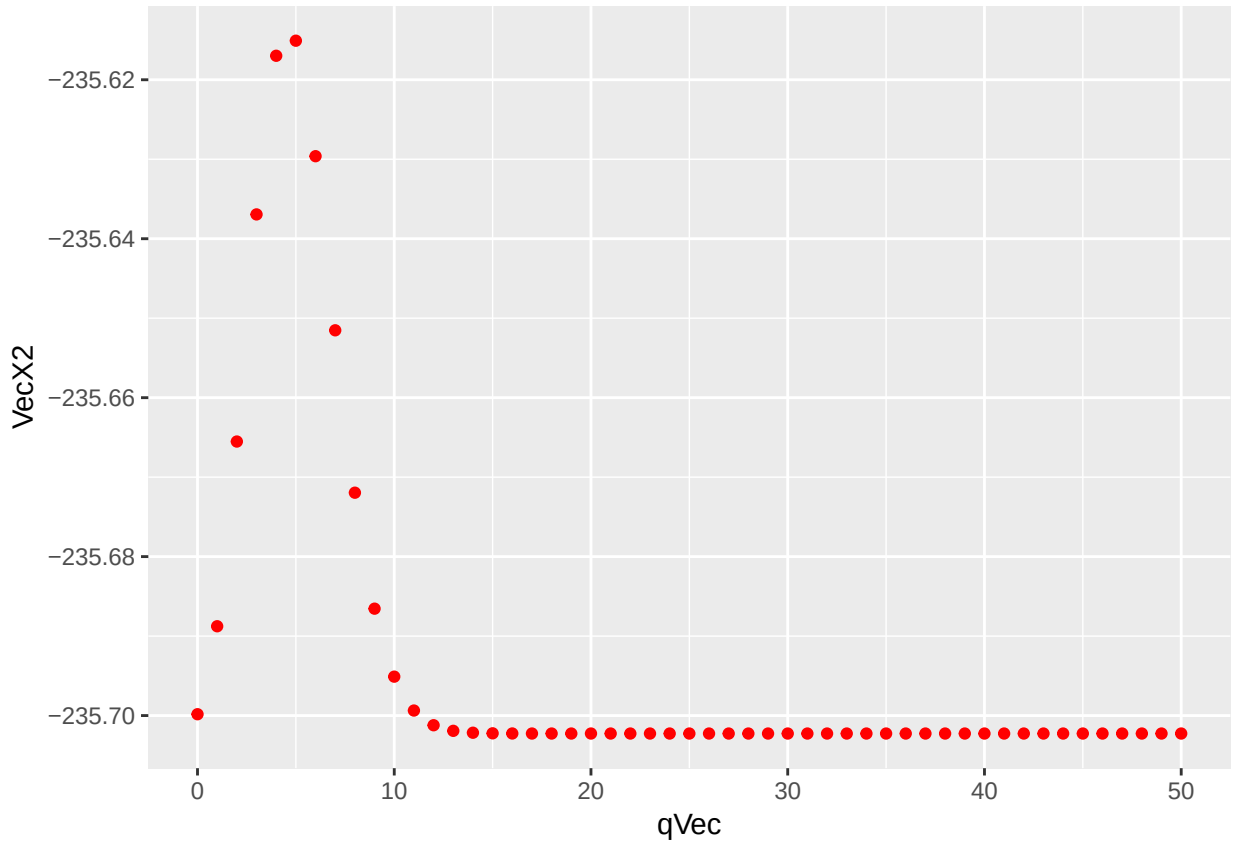
The first pair seems to approximate the normal distribution more than the others. Increasing S decreased the vertical scale, while adjusting p skewed it and gave the distributions a tail.

```
X2<-function(y, s, p){(y-s/p)/sqrt(s*p*(1-p))}
#first pair
s<-50
p<-0.5
qVec<-0:s
var3<-dbinom(qVec, s, p)

#apply to binomial distribution
VecX2<-X2(var3, s, p)
dNormDat<-data.frame(value=qVec, prob=var3)
g2<-ggplot(dNormDat, aes(x=qVec, y = VecX2)) + geom_point(colour = "blue") #+ stat_function(fun = dnorm

g2
```

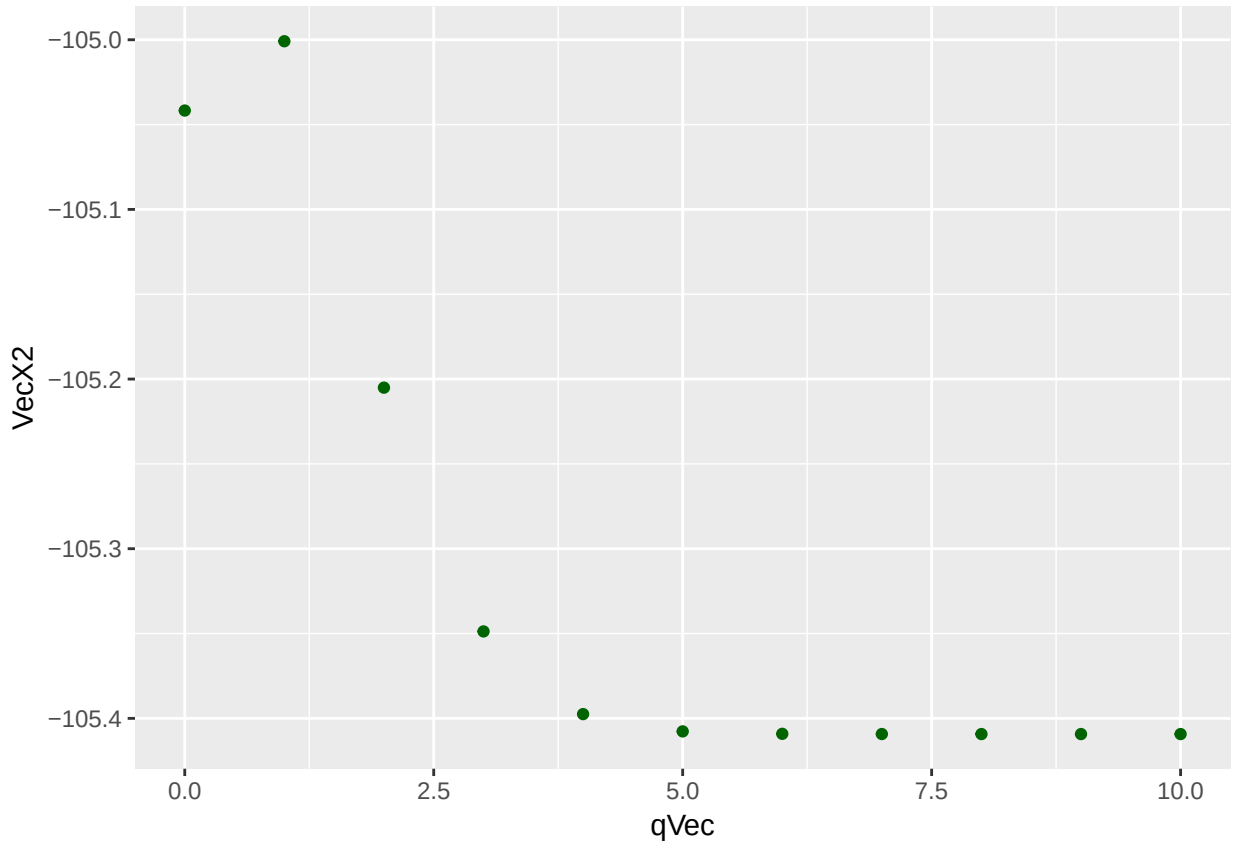



```
#compare to standard normal

#second pair
s<-50
p<-0.1
qVec<-0:s
var3<-dbinom(qVec, s, p)
#third pair
```

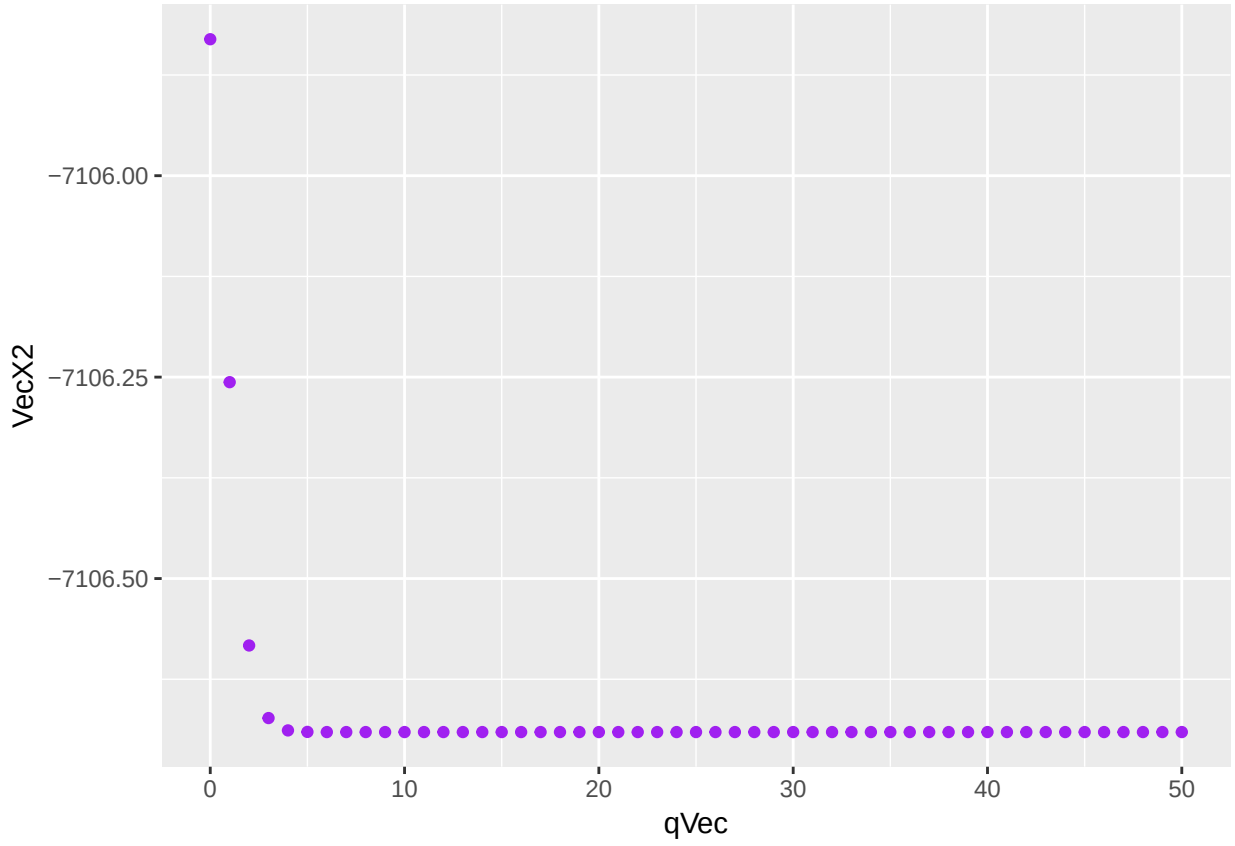
```r
#apply to binomial distribution
VecX2<-X2(var3, s, p)
dNormDat<-data.frame(value=qVec, prob=var3)
g3<-ggplot(dNormDat, aes(x=qVec, y = VecX2)) + geom_point(colour = "red")
g3
```



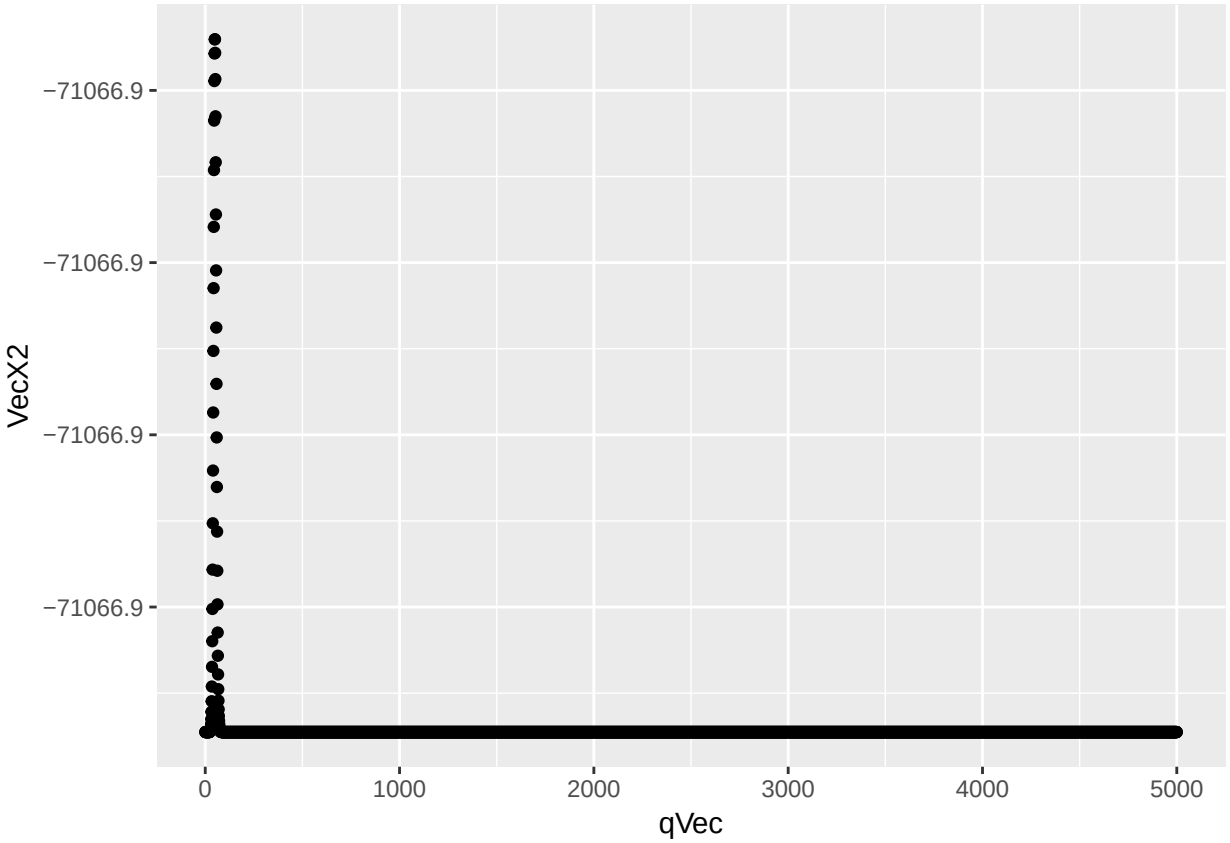

```r
s<-10
p<-0.1
qVec<-0:s
var3<-dbinom(qVec, s, p)

#apply to binomial distribution
VecX2<-X2(var3, s, p)
dNormDat<-data.frame(value=qVec, prob=var3)
g4<-ggplot(dNormDat, aes(x=qVec, y = VecX2)) + geom_point(colour = "dark green")
g4
```




```
#fourth pair
s<-50
p<-0.01
qVec<-0:s
var3<-dbinom(qVec, s, p)

#apply to binomial distribution
VecX2<-X2(var3, s, p)
dNormDat<-data.frame(value=qVec, prob=var3)
g5<-ggplot(dNormDat, aes(x=qVec, y = VecX2)) + geom_point(colour = "purple")
g5
```




```
#fifth pair
s<-5000
p<-0.01
qVec<-0:s
var3<-dbinom(qVec, s, p)

#apply to binomial distribution
VecX2<-X2(var3, s, p)
dNormDat<-data.frame(value=qVec, prob=var3)
g6<-ggplot(dNormDat, aes(x=qVec, y = VecX2)) + geom_point()
g6
```



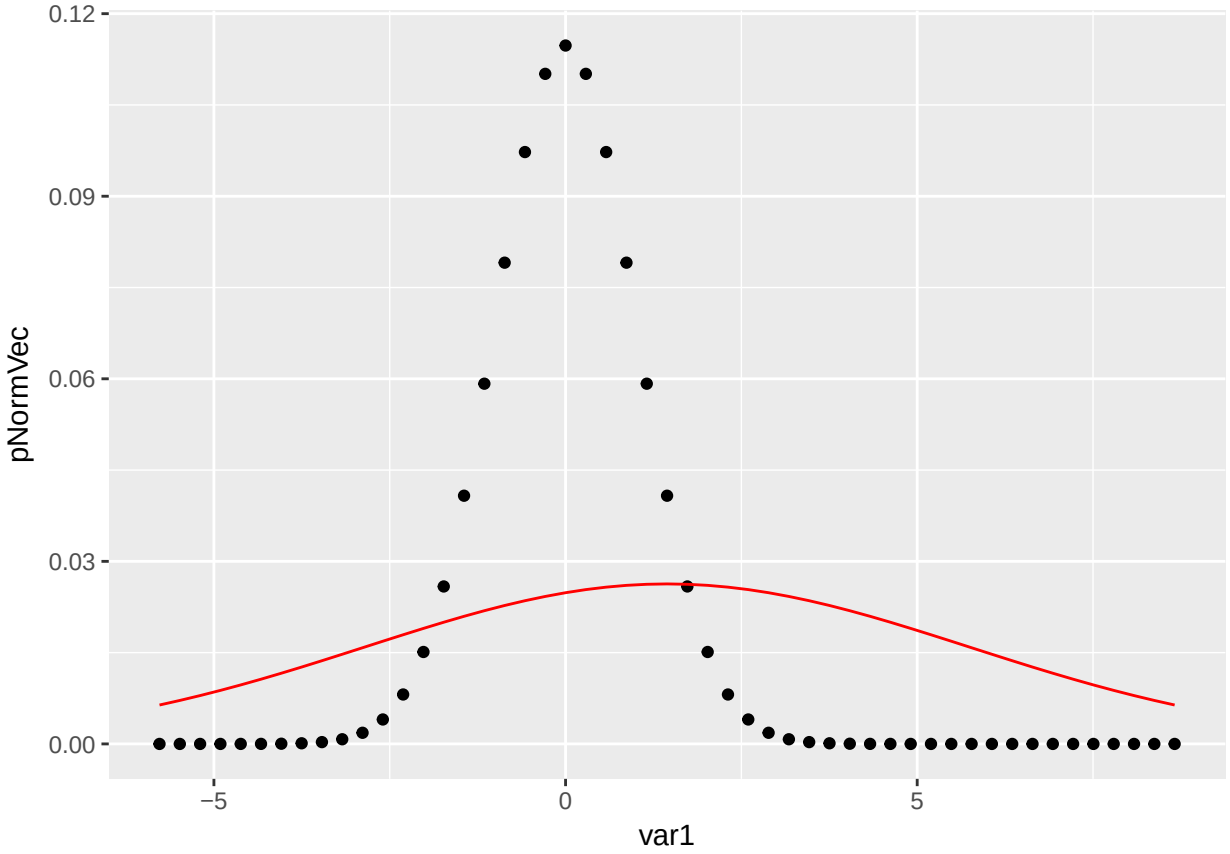
6. (4441 only) Recall the normal probabilities from question 4: the x values are (z-50*.4)/sqrt(50*.4*(1-.4)) for the integers z in {0,1,2,3,...50}. The value d is defined by d=1/sqrt(50*.4*(1-.4)), For each x value, the point (x,y) where y is probability that a draw from the standard normal distribution lies between x-d/2 and x+d/2 was plotted. If you plot the density of the standard normal distribution on the same plot, it appears to be of a similar shape but different vertical scale. What can you multiply the standard normal density function by to make the scales comparable? Please give a formula in terms of s and p and explain your reasoning.(10 points)

Response

In order to make the scales comparable, you can multiply the standard normal distribution by the multiplicative inverse of d. This was done in order to "undo" the effects that d had on the graph in problem #4.The formula is SPform, which returns sqrt(s*p*(1-p)).

```
#set d
d=1/sqrt(50*0.4*(1-0.4))

#random variable

X<-function(x){(x-50*0.4)/(sqrt(50*0.4*(1-0.4)))}
pNormfun<-function(x, d){
  abs(pnorm(x+d/2) - pnorm(x-d/2))
}
var1 <- X(0:50)
probVar1 <- dbinom(0:50, 50, .4)
var1DF <- data.frame(value = var1, prob = probVar1)
```

```
pNormVec<-sapply(var1, pNormfun, d=d)
pNormDat<-data.frame(pNormVec)




#apply pnorm to each value in var1 +- d/2
#loop through in vector form, apply pnorm to a vector
#feed pnorm a vector as an x argument, plot points

pNormVec<-sapply(var1, pNormfun, d=d)
pNormDat<-data.frame(pNormVec)

s <- 50
p <- .5

SPform <- function(s, p){sqrt(s*p*(1-p))}
newD <- SPform(s, p)

distVec <- newD*var1
distFrame <- data.frame(distVec)
dcurve <- dnorm(distVec, mean(distVec), sd(distVec))


g7<-ggplot(pNormDat, aes(x=var1, y=pNormVec)) + geom_point()+ geom_line(aes(y = dcurve), data = distFra
g7
```


0.12
0.09
0.06
0.03
0.00
pNormVec
−5
0
5
var1