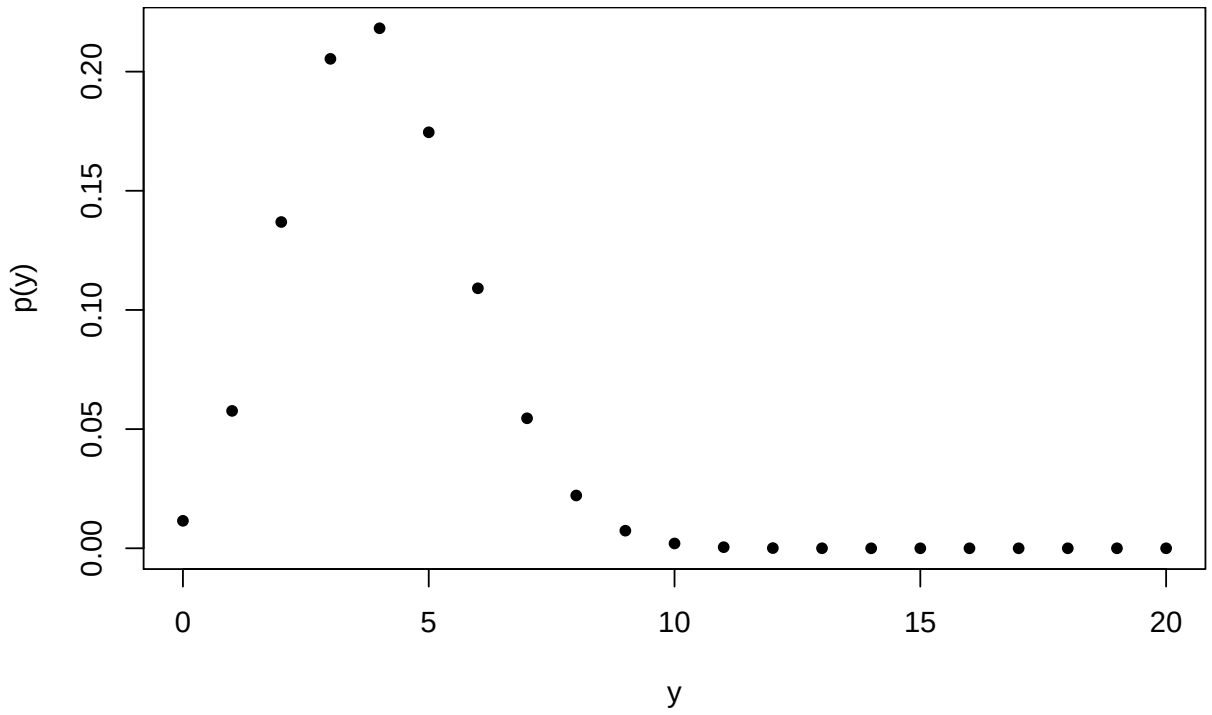
DISCRETE DISTRIBUTIONS: EXAMPLES IN R

1. **The Binomial distribution.** The binomial pmf $p(y)$ takes the form

$$p(y) = \binom{n}{y} p^y (1-p)^{n-y} \qquad y = 0, 1, 2, \ldots, n$$

with $p(y) = 0$ for other values of y. This pmf can be computed using the `dbinom` function in R.

```
n<-20;p<-0.2;y<-0:n                          #Set the parameters
p.y<-dbinom(y,n,p)                           #Compute the pmf
par(mar=c(4,4,0,2))                          #Set up the plot
plot(y,p.y,pch=19,ylab='p(y)',cex=0.75)      #Produce the plot
```




```
print(cbind(y,p.y))                          #Print the numerical values of y and p(y)
```

```
+         y                    p.y
+   [1,]  0 0.0115292150460684812 7
+   [2,]  1 0.0576460752303423335 0
+   [3,]  2 0.1369094286720630993 0
+   [4,]  3 0.2053641430080946073 1
+   [5,]  4 0.2181994019461005185 3
+   [6,]  5 0.1745595215568804536 9
+   [7,]  6 0.1090997009730503147 8
+   [8,]  7 0.0545498504865251296 3
+   [9,]  8 0.0221608767601508133 1
+  [10,]  9 0.0073869589200502789 1
+  [11,] 10 0.0020314137030138243 8
+  [12,] 11 0.0004616849325031422 7
+  [13,] 12 0.0000865659248443391 7
+  [14,] 13 0.0000133178345914367 6
+  [15,] 14 0.0000016647293239296 0
+  [16,] 15 0.0000001664729323929 6
+  [17,] 16 0.0000000130056978432 0
+  [18,] 17 0.0000000007650410496 0
```

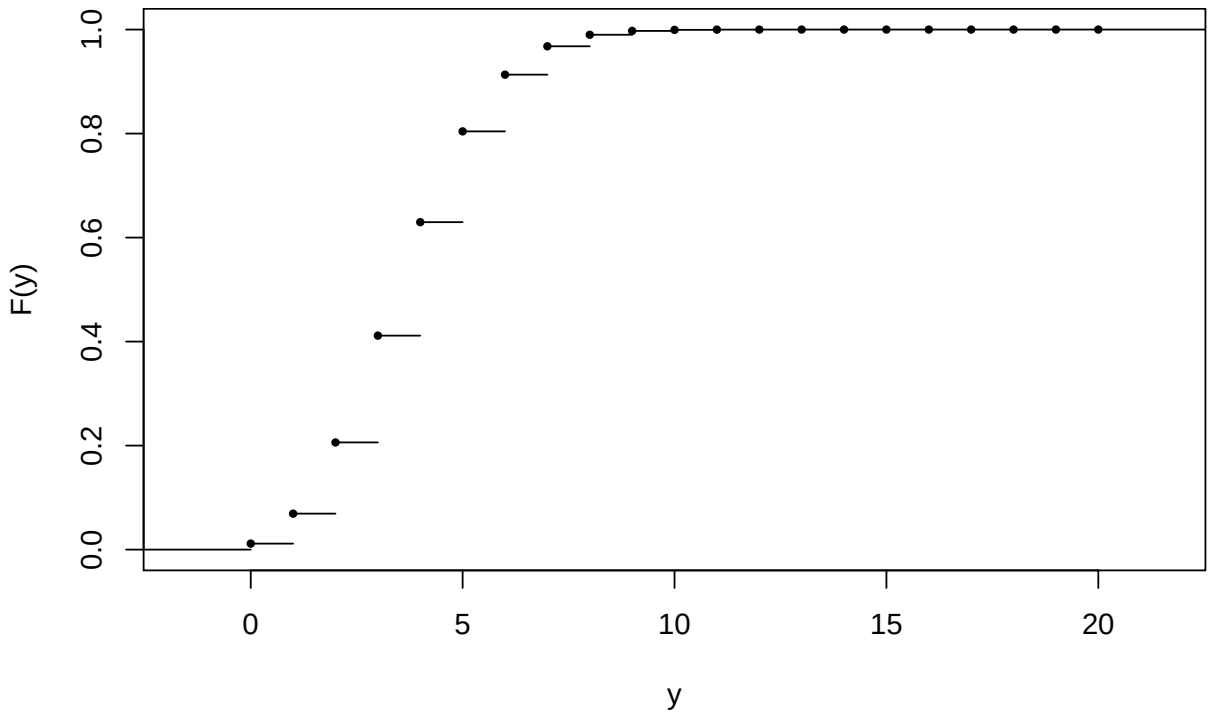
```
+ [19,] 18 0.00000000003187671040
+ [20,] 19 0.00000000000083886080
+ [21,] 20 0.00000000000001048576
```

We can use the function `pbinom` to compute the cdf $F(y)$

$$F(y) = P(Y \leq y) = \sum_{j=0}^{\lfloor y \rfloor} p(j) = \sum_{j=0}^{\lfloor y \rfloor} \binom{n}{j} p^j (1-p)^{n-j} \qquad y > 0$$

where $\lfloor y \rfloor$ is the largest integer no greater than y.

```
F.y<-pbinom(y,n,p)                          #Compute the cdf
par(mar=c(4,4,0,2))
plot(stepfun(y,c(0,F.y)),xlab='y',ylab='F(y)',pch=19,cex=0.5,vertical=FALSE,main='')
```



We can compute the expectation and variance by using the `sum` function

```
Expectation<-sum(y*p.y)                     #Compute the expectation
Variance<-sum((y-Expectation)^2*p.y)        #Compute the variance
c(Expectation,Variance)                     #Values are np and npq

+ [1] 4.0 3.2
```

Finally, we can simulate independent random values from this distribution using `rbinom`. To simulate $N = 50$ random, Binomial random variables:

```
N<-50
Ysamp<-rbinom(N,n,p)
Ysamp

+  [1] 5 7 3 6 5 1 7 6 3 4 2 5 3 6 6 7 6 2 5 6 5 2 4 3 5 1 5 6 4 4 3 4 7 5 2
+ [36] 1 5 3 7 1 6 7 7 5 4 3 3 6 7 3
```

2. **The Geometric distribution.** The geometric pmf $p(y)$ takes the form

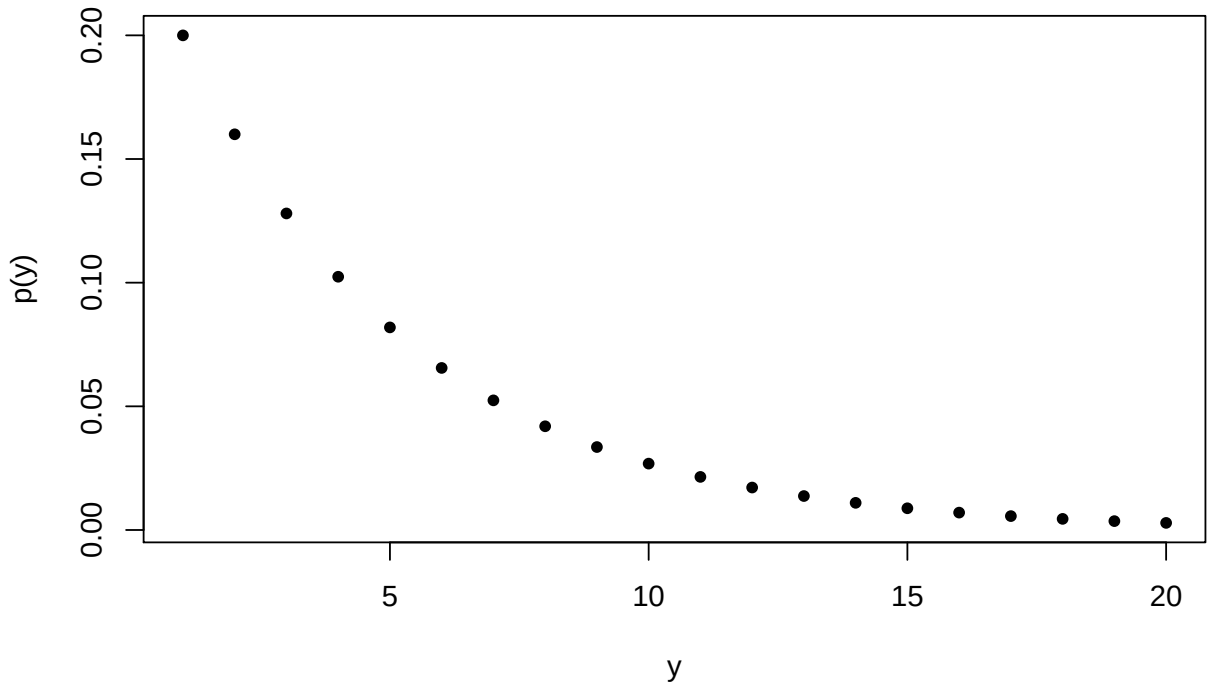
$$p(y) = (1-p)^{y-1}p \qquad y = 1, 2, \ldots,$$

with $p(y) = 0$ for other values of y. This pmf can be computed using the `dgeom` function in `R`, however, `R` uses the alternative form of the pmf

$$p(y) = (1-p)^{y}p \qquad y = 0, 1, 2, \ldots,$$

so we need to subtract one from the argument of `dgeom`

```
p<-0.2;y<-1:20                          #Set the parameters
p.y<-dgeom(y-1,p)                       #Compute the pmf
par(mar=c(4,4,1,2))                     #Set up the plot
plot(y,p.y,pch=19,ylab='p(y)',cex=0.75) #Produce the plot
```



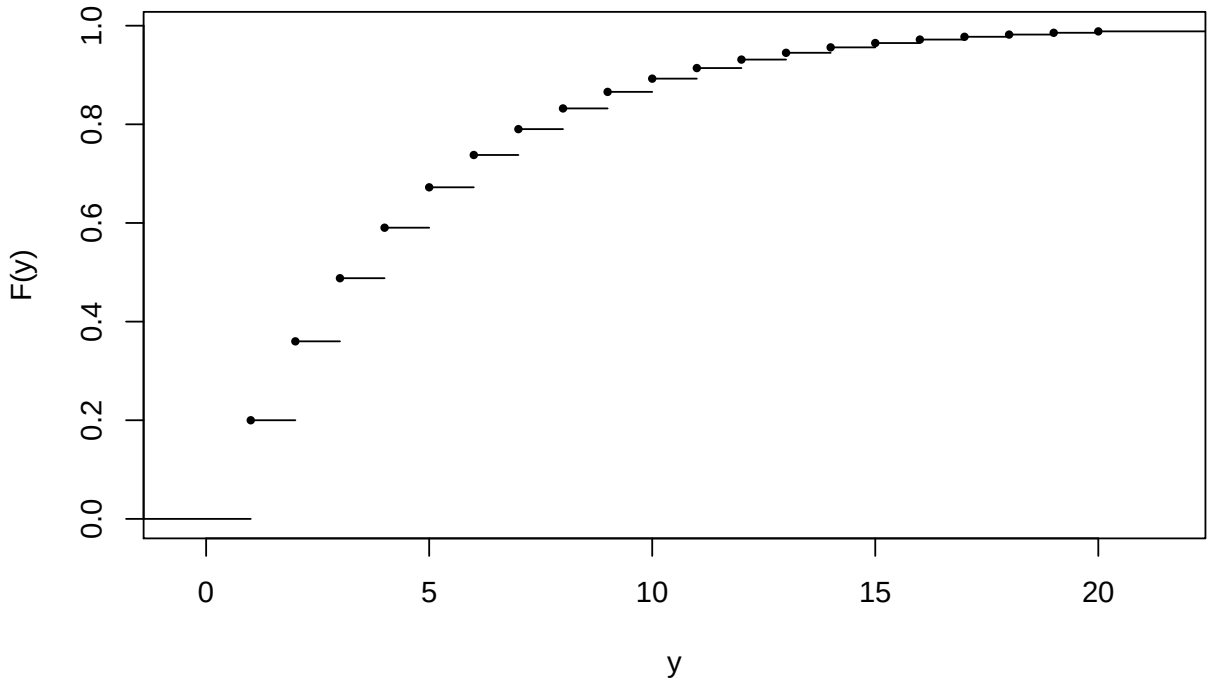

```
print(cbind(y,p.y))                     #Print the numerical values of y and p(y)

+         y         p.y
+   [1,]  1 0.200000000
+   [2,]  2 0.160000000
+   [3,]  3 0.128000000
+   [4,]  4 0.102400000
+   [5,]  5 0.081920000
+   [6,]  6 0.065536000
+   [7,]  7 0.052428800
+   [8,]  8 0.041943040
+   [9,]  9 0.033554432
+  [10,] 10 0.026843546
+  [11,] 11 0.021474836
+  [12,] 12 0.017179869
+  [13,] 13 0.013743895
+  [14,] 14 0.010995116
+  [15,] 15 0.008796093
+  [16,] 16 0.007036874
+  [17,] 17 0.005629500
+  [18,] 18 0.004503600
+  [19,] 19 0.003602880
+  [20,] 20 0.002882304
```

We can use the function `pgeom` to compute the cdf $F(y)$: for $y = 1, 2, 3, \ldots$,

$$F(y) = P(Y \leq y) = \sum_{j=1}^{y} p(j) = \sum_{j=1}^{y} (1-p)^{y-1} p = 1 - (1-p)^y$$

```
F.y<-pgeom(y-1,p)                          #Compute the cdf
par(mar=c(4,4,1,2))
plot(stepfun(y,c(0,F.y)),xlab='y',ylab='F(y)',pch=19,cex=0.5,vertical=FALSE,main='')
```



We can compute the expectation and variance by using the `sum` function. Here, we can only approximate the infinite sum by taking the first 10000 (say) terms

```
y<-1:10000                                 #Take the first 10000 y values
p.y<-dgeom(y-1,p)
Expectation<-sum(y*p.y)                    #Compute the expectation
Variance<-sum((y-Expectation)^2*p.y)       #Compute the variance
c(Expectation,Variance)                    #Values are 1/p and (1-p)/p^2

+ [1]  5 20
```

Finally, we can simulate independent random values from this distribution using `rgeom`. To simulate $N = 50$ random, Geometric random variables: we must remember to add one to the generated values due to the alternative pmf that `R` uses.

```
N<-50
Ysamp<-rgeom(N,p)+1
Ysamp

+  [1]  1  6  6  5  1  1  1  2  3  2  4 11 10  2  7  3  3 10  5  1 14  5 15
+ [24]  4  5  5 19 26  1  4  2  8  5  8  4  2  1  2  2  9  3  4  2  1  5  1
+ [47]  4  5  6  1
```

3. **The Negative Binomial distribution.** The negative binomial pmf $p(y)$ takes the form

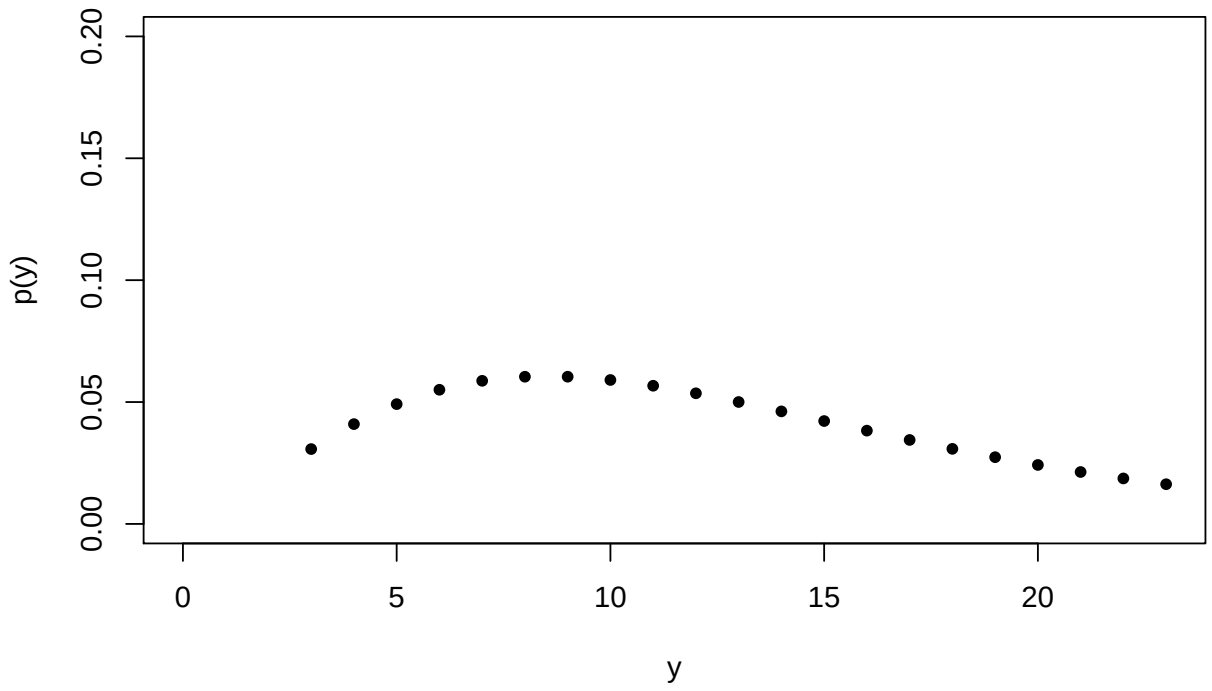
$$p(y) = \binom{y-1}{r-1} p^r (1-p)^{y-r} \qquad y = r, r+1, r+2, \ldots,$$

with $p(y) = 0$ for other values of y, for some fixed positive integer r. This pmf can be computed using the `dnbinom` function in `R`, however, `R` uses the alternative form of the pmf

$$p(y) = \binom{y+r-1}{r-1} p^r (1-p)^{y} \qquad y = 0, 1, 2, \ldots,$$

so we need to subtract r from the argument of `dnbinom`. Below, we use $r = 3$;

```
p<-0.2;r<-3;y<-r+c(0:20)                          #Set the parameters
p.y<-dnbinom(y-1,r,p)                             #Compute the pmf
par(mar=c(4,4,1,2))                               #Set up the plot
plot(y,p.y,pch=19,ylab='p(y)',cex=0.75,xlim=range(0,max(y)),ylim=range(0,0.2))
```



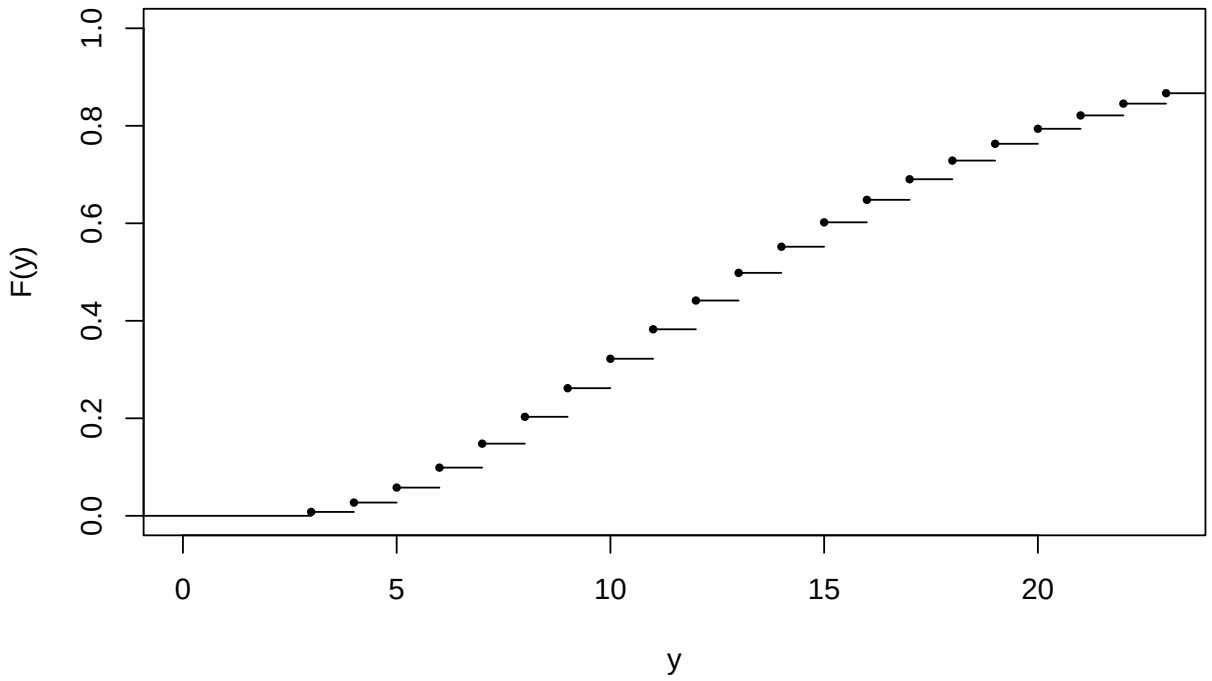

```
print(cbind(y,p.y))                               #Print the numerical values of y and p(y)
```

```
+         y       p.y
+   [1,]  3 0.03072000
+   [2,]  4 0.04096000
+   [3,]  5 0.04915200
+   [4,]  6 0.05505024
+   [5,]  7 0.05872026
+   [6,]  8 0.06039798
+   [7,]  9 0.06039798
+   [8,] 10 0.05905580
+   [9,] 11 0.05669357
+  [10,] 12 0.05360119
+  [11,] 13 0.05002778
+  [12,] 14 0.04617949
+  [13,] 15 0.04222125
+  [14,] 16 0.03828060
```

```
+ [15,] 17 0.03445254
+ [16,] 18 0.03080462
+ [17,] 19 0.02738189
+ [18,] 20 0.02421135
+ [19,] 21 0.02130599
+ [20,] 22 0.01866811
+ [21,] 23 0.01629216
```

We can use the function `pnbinom` to compute the cdf $F(y)$:

```
F.y<-pnbinom(y-r,r,p)                          #Compute the cdf
par(mar=c(4,4,1,2))
plot(stepfun(y,c(0,F.y)),xlab='y',ylab='F(y)',xlim=range(0,max(y)),ylim=range(0,1),
       pch=19,cex=0.5,vertical=FALSE,main='')
```



We can compute the expectation and variance by using the `sum` function. Here, we can only approximate the infinite sum by taking the first 10000 (say) terms

```
y<-r+c(0:10000)                                #Take the first 10000 y values
p.y<-dnbinom(y-r,r,p)
Expectation<-sum(y*p.y)                        #Compute the expectation
Variance<-sum((y-Expectation)^2*p.y)           #Compute the variance
c(Expectation,Variance)                        #Values are r/p and r(1-p)/p^2

+ [1] 15 60
```

Finally, we can simulate independent random values from this distribution using `rnbinom`. To simulate $N = 50$ random, negative binomial random variables: we must remember to add r to the generated values due to the alternative pmf that `R` uses.

```
N<-50;Ysamp<-rnbinom(N,r,p)+r; Ysamp

+  [1] 47 10 12 20  9 14  7 17 12 16 25 13  4 11  4 43 14  9  7 19  5 12 23
+ [24] 12 35 19  4 14  8 20 24 11  4  9 35  9 19 15  9 12 18 10 14 25 15 35
+ [47] 13 10 20 13
```

The experimental context that defines the use of the negative binomial distribution is that the random variable records the number of independent binary trials required to obtain the rth 'success'. Therefore, we can regard Y as being the sum of independent geometric random variables

$$Y = \sum_{i=1}^{r} Y_i$$

where Y_i is the number of trials between the $(i-1)$st and the ith success. For example, we can regard the sequence

$$001000001001$$

by breaking it down as the three sub-sequences

$$001 \quad + \quad 000001 \quad + \quad 001.$$

We verify this in the following simulation of $N = 100000$ negative binomial random variables with $r = 3$ and $p = 0.3$:

```
N<-100000
Yi.mat<-matrix(rgeom(N*3,p),ncol=3)      #Simulate an N x 3 matrix of Geometric(0.2) r.v.s
Y<-rowSums(Yi.mat)+3                     #Form the vector of row sums of the matrix.
Sim.tab<-cbind(1:25,dnbinom(1:25-r,r,p),tabulate(Y,nbins=25)/N)
colnames(Sim.tab)<-c('y','p(y)','Rel. Freq.')
Sim.tab
```

```
+        y       p(y) Rel. Freq.
+   [1,]  1 0.00000000    0.00000
+   [2,]  2 0.00000000    0.00000
+   [3,]  3 0.00800000    0.00744
+   [4,]  4 0.01920000    0.01992
+   [5,]  5 0.03072000    0.03159
+   [6,]  6 0.04096000    0.04237
+   [7,]  7 0.04915200    0.04899
+   [8,]  8 0.05505024    0.05600
+   [9,]  9 0.05872026    0.05806
+  [10,] 10 0.06039798    0.06093
+  [11,] 11 0.06039798    0.05977
+  [12,] 12 0.05905580    0.05811
+  [13,] 13 0.05669357    0.05661
+  [14,] 14 0.05360119    0.05360
+  [15,] 15 0.05002778    0.05007
+  [16,] 16 0.04617949    0.04691
+  [17,] 17 0.04222125    0.04173
+  [18,] 18 0.03828060    0.03845
+  [19,] 19 0.03445254    0.03406
+  [20,] 20 0.03080462    0.03106
+  [21,] 21 0.02738189    0.02682
+  [22,] 22 0.02421135    0.02360
+  [23,] 23 0.02130599    0.02081
+  [24,] 24 0.01866811    0.01918
+  [25,] 25 0.01629216    0.01588
```

In this table, the column `Rel. Freq.` is the relative frequency of times (out of 1000000 simulated values) that that value of y was observed. We could expect roughly a proportion $p(y)$ of simulated values to take the value y: that is, the second and third columns should match.