Lab 1

Björn Edblom (bjoed735) och Eskil Brännerud (eskbr129)

2023-09-21

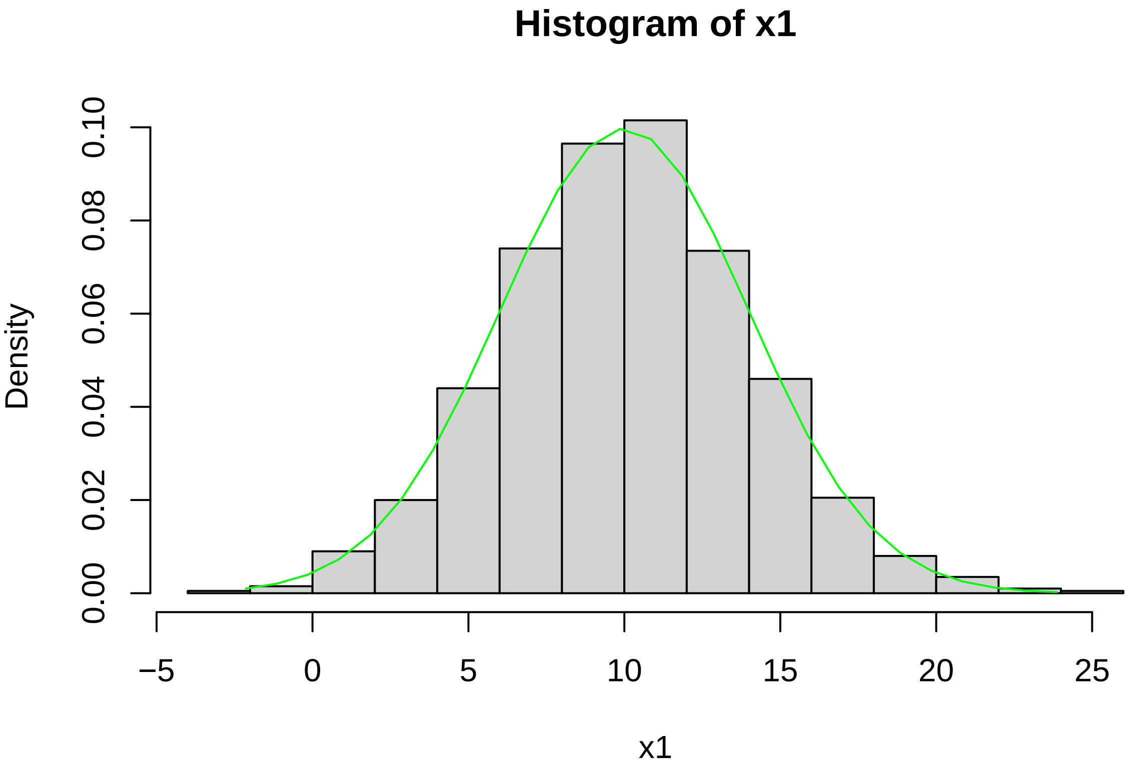
Uppgift 3.1.1

Simulering av normalfördelning. Simulera 100 och 10000 dragningar från en normalfördelning med u = 10 och o = 4.

1) Visualisera fördelningarna i två histogram. Visualisera fördelningens pdf i samma graf. Nedan simuleras normalfördelningen med olika antalet dragningar, samt ett histogram av dragningarna från normal-fördelningen.

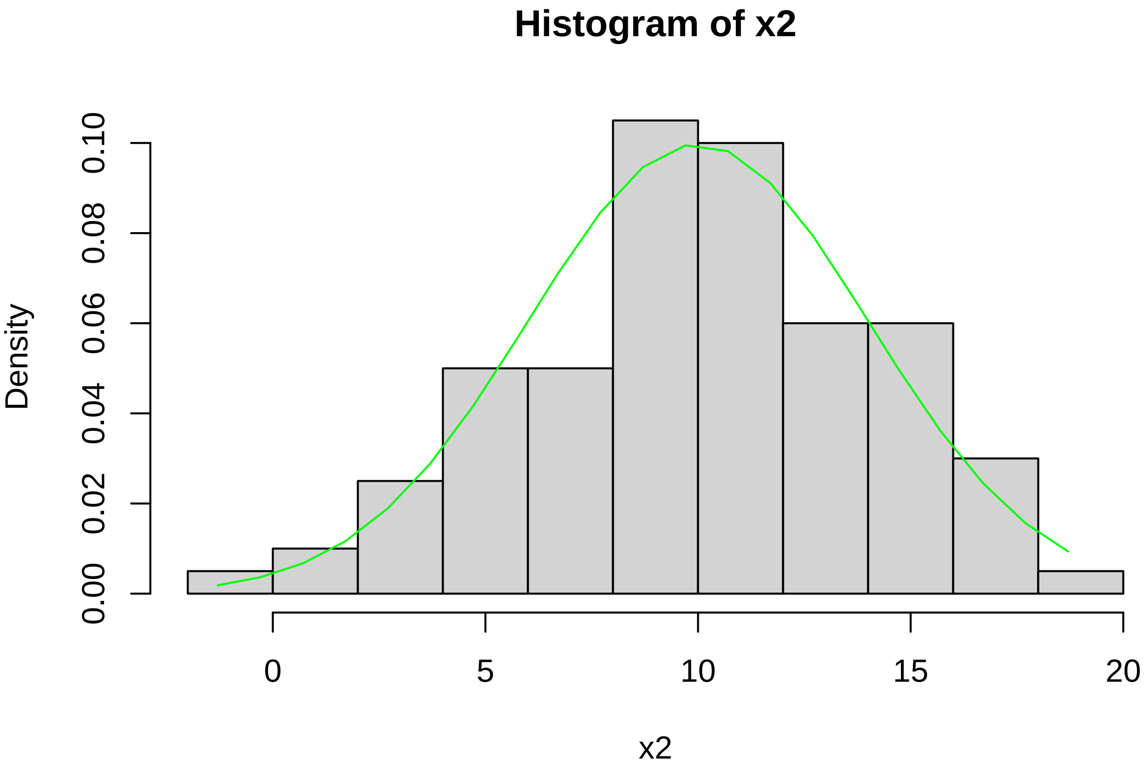
```
x1 <- rnorm(1000, mean = 10, sd = 4)

hist(x1, probability = TRUE)
xfit <- seq(min(x1), max(x1), 1)
yfit <- dnorm(xfit, mean = 10, sd = 4 )
lines(xfit, yfit, col="green")
```



```
x2 <- rnorm(100, mean = 10, sd = 4)
hist(x2, probability = TRUE)
xfit <- seq(min(x2), max(x2), 1)
yfit <- dnorm(xfit, mean = 10, sd = 4 )
```

```
lines(xfit, yfit, col="green")
```



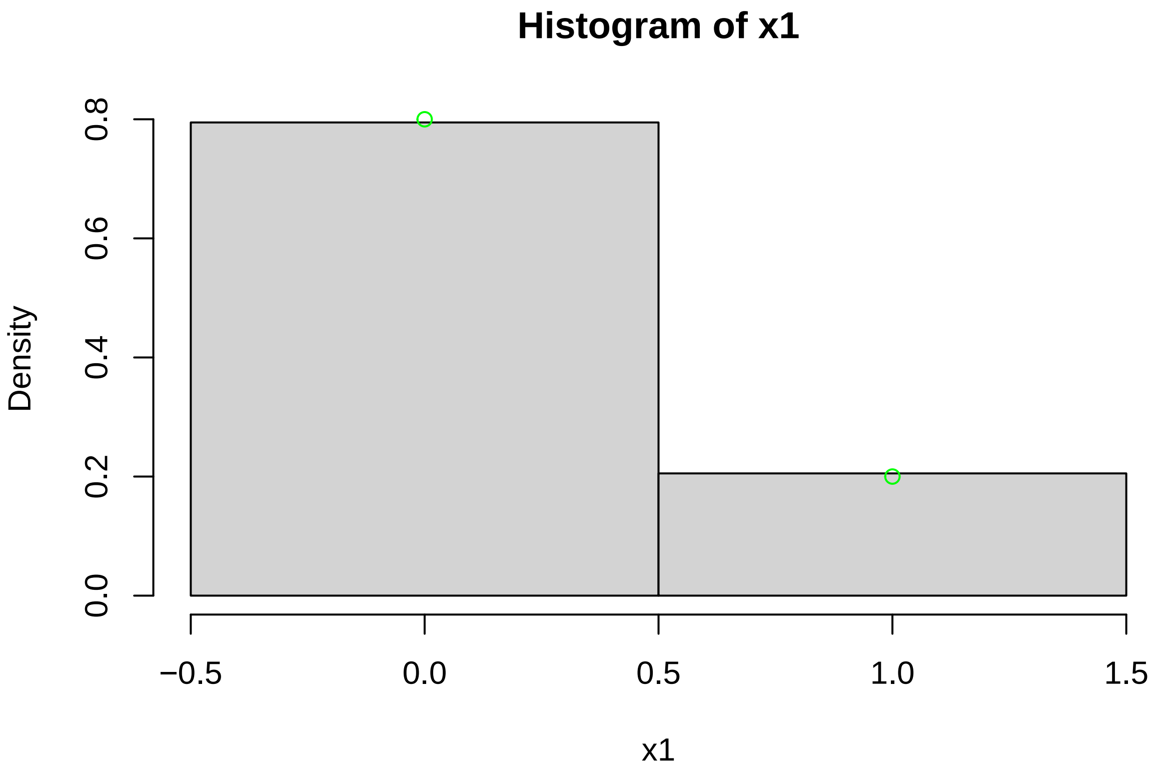
2) Beskriv skillnaden mellan de olika graferna. I graf har fler dragningar, vilket gör att den är mer stabil mellan körningar. Den närmar sig mer normalfördelningen.

Uppgift 3.1.2

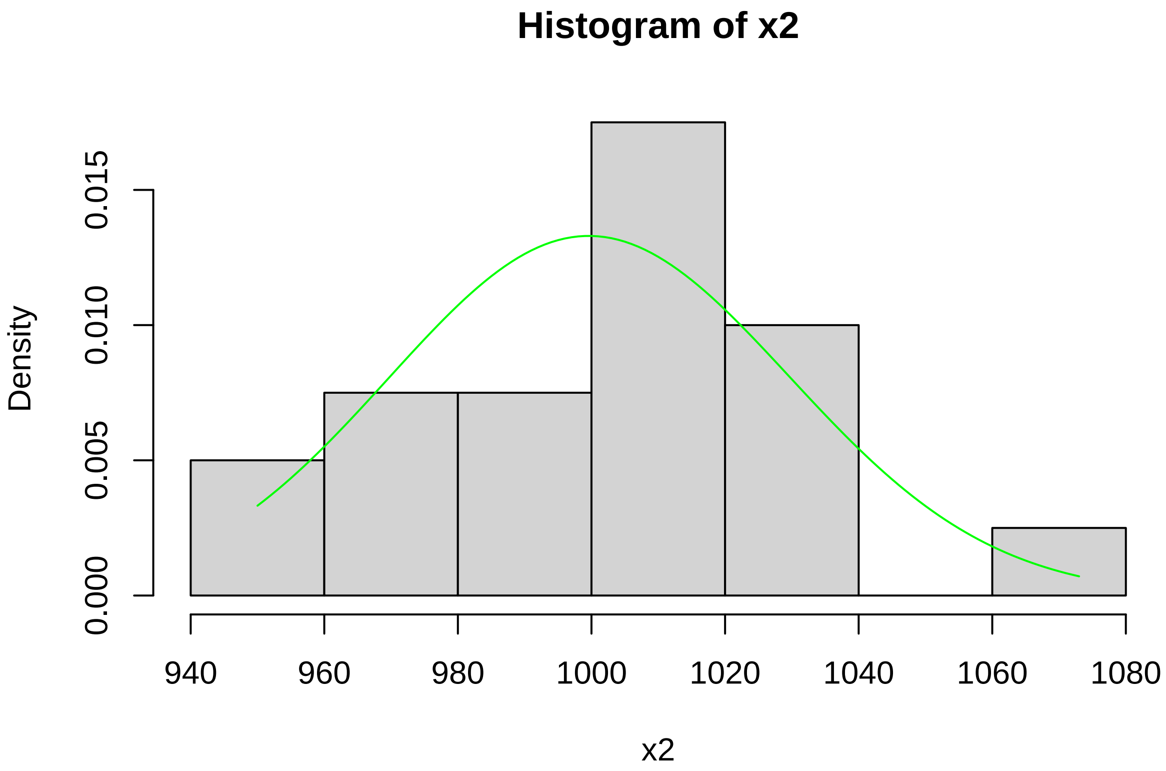
Simulera och visualisera andra fördelningar.

1) Simulera och visualisera följande fördelningar med 10000 dragningar från varje fördelningens täthetsfunktioner Diskreta fördelningar

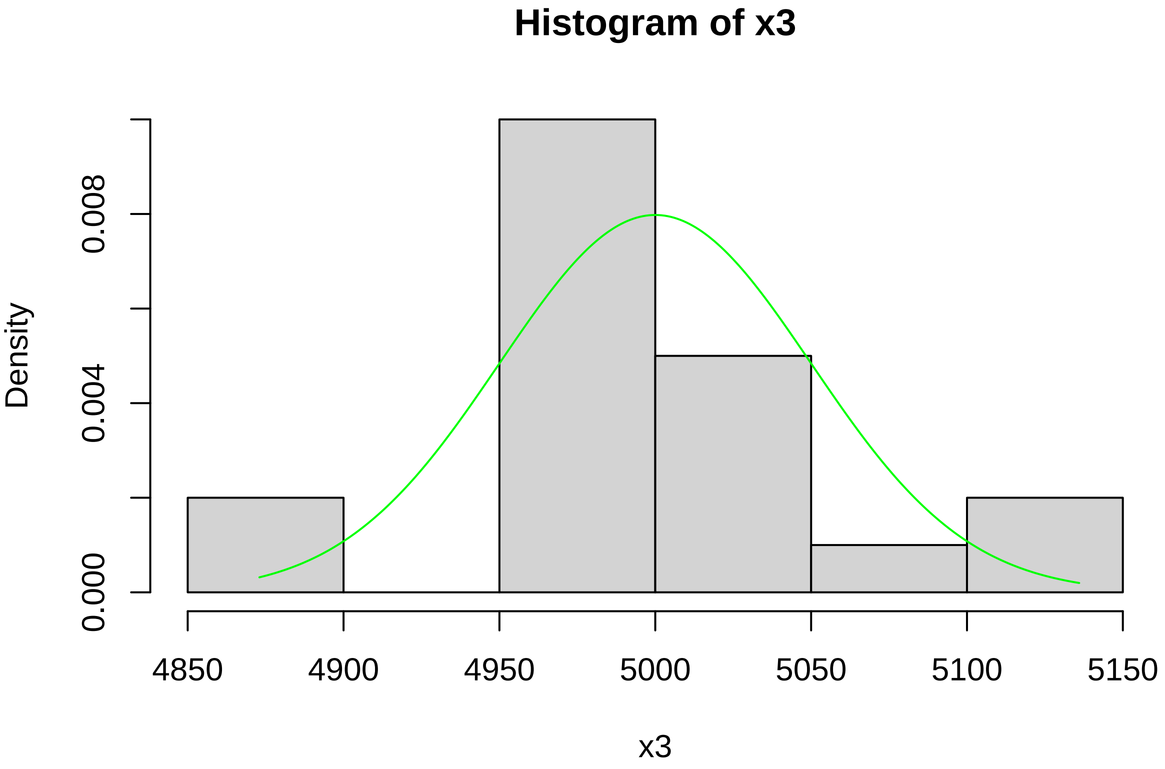
```
x1 <- rbern(10000, 0.2)
hist(x1, probability = TRUE, breaks=c(-0.5,0.5,1.5))
xfit <- seq(min(x1), max(x1), 1)
yfit <- dbern(xfit, prob = 0.2)
points(xfit, yfit, col="green")
```



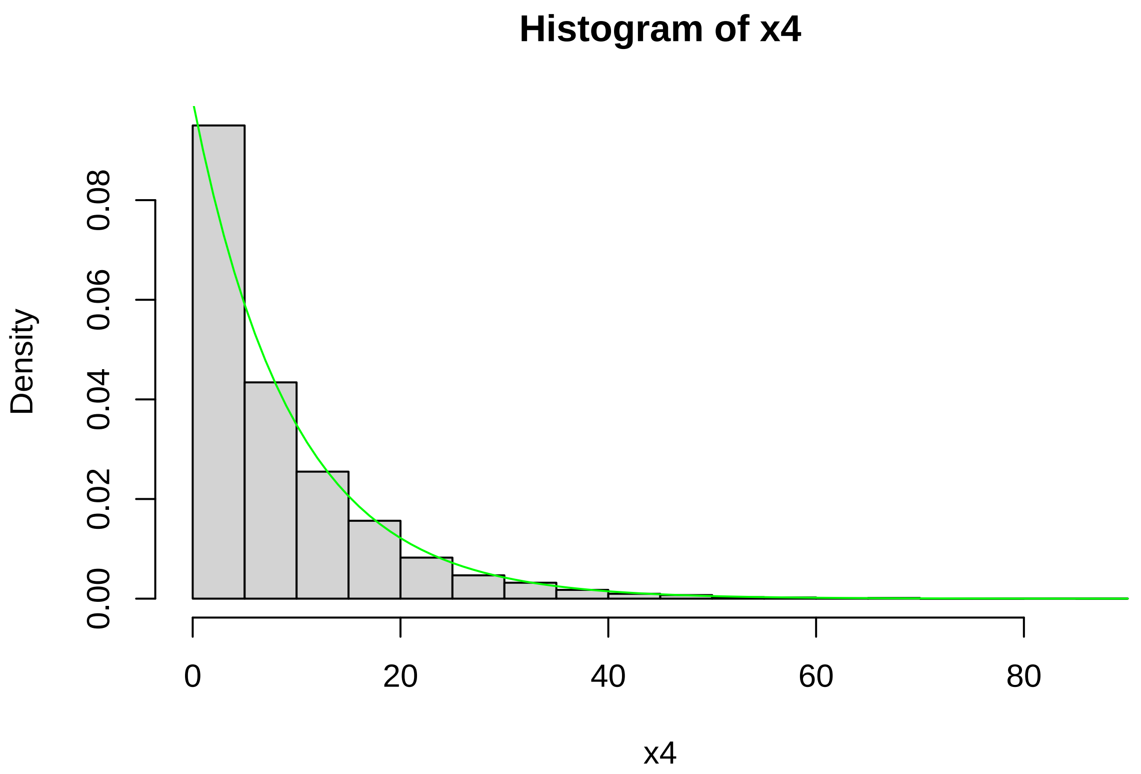
```r
x2 <- rbinom(20,10000,0.1)
hist(x2, probability = TRUE)
xfit <- seq(min(x2), max(x2), 1)
yfit <- dbinom(xfit, size = 10000, prob = 0.1)
lines(xfit, yfit, col="green")
```



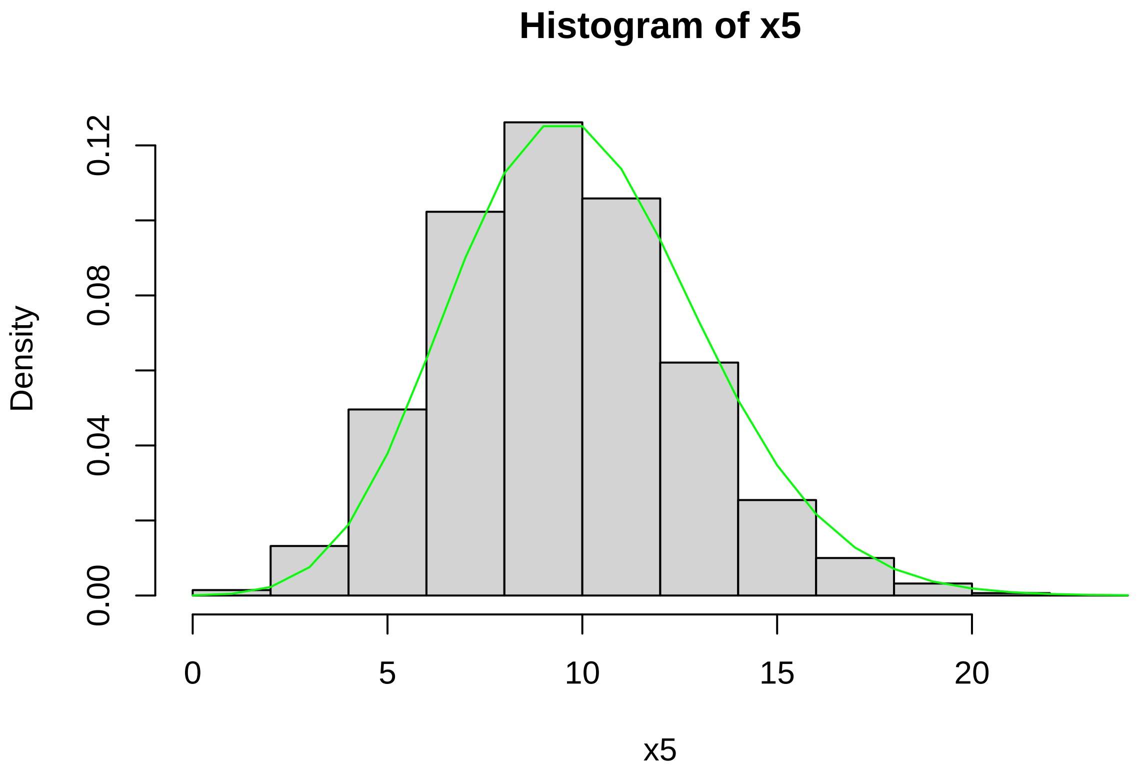
```
x3 <- rbinom(20,10000,0.5)
hist(x3, probability = TRUE)
xfit <- seq(min(x3), max(x3), 1)
yfit <- dbinom(xfit, size = 10000, prob = 0.5)
lines(xfit, yfit, col="green")
```



```
x4 <- rgeom(10000, 0.1)
hist(x4, probability = TRUE)
xfit <- seq(min(x4), max(x4), 1)
yfit <- dgeom(xfit, prob = 0.1 )
lines(xfit, yfit, col="green")
```

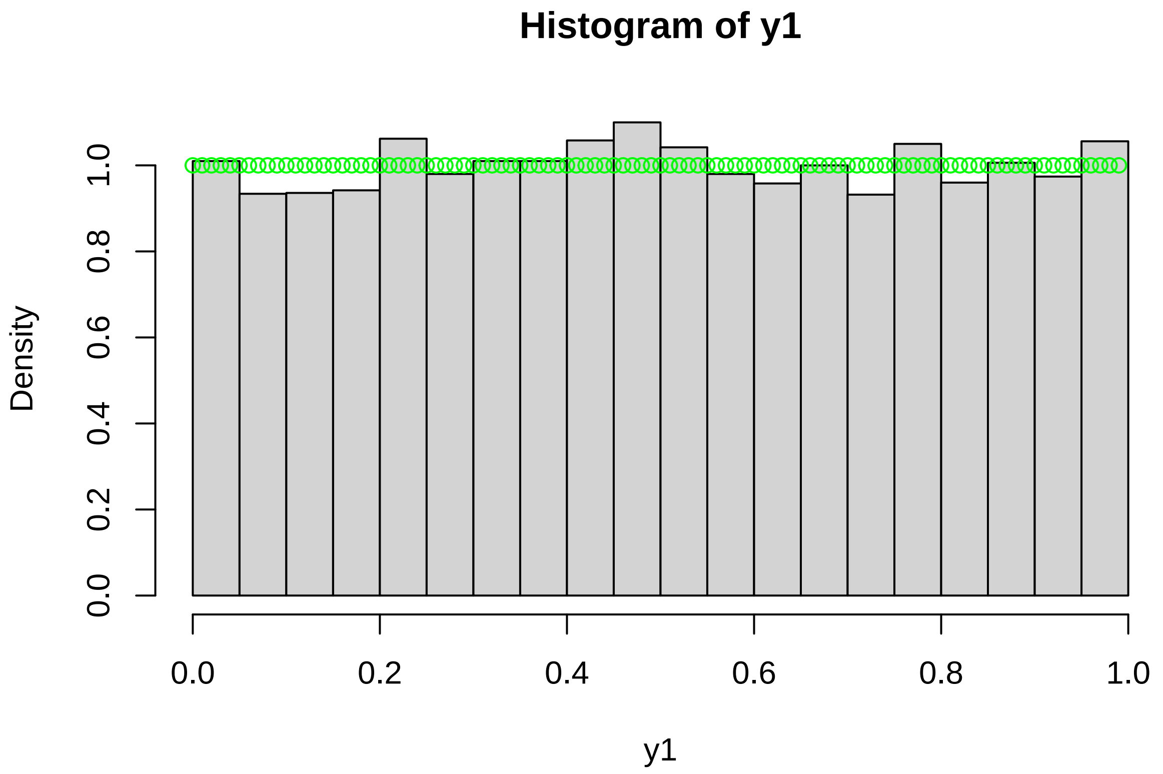


```
x5 <- rpois(10000,10)
hist(x5, probability = TRUE)
xfit <- seq(min(x5), max(x5), 1)
yfit <- dpois(xfit, lambda= 10)
lines(xfit, yfit, col="green")
```

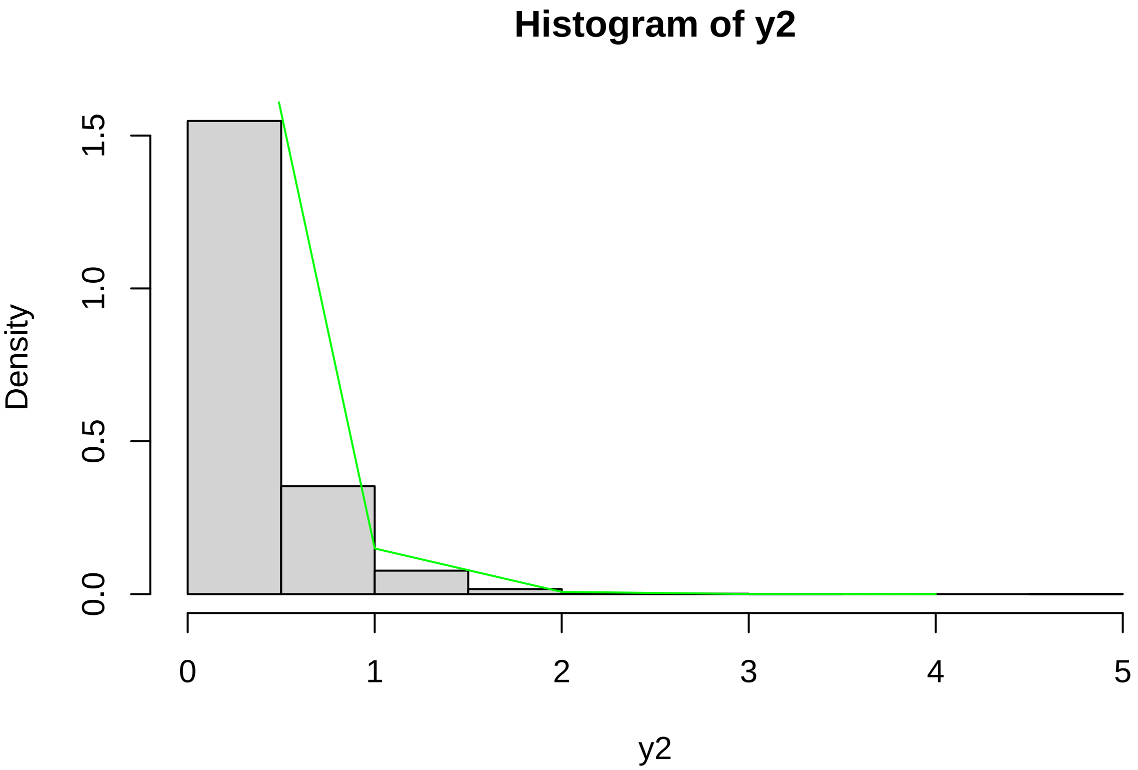


Kontinuerliga fördelningar

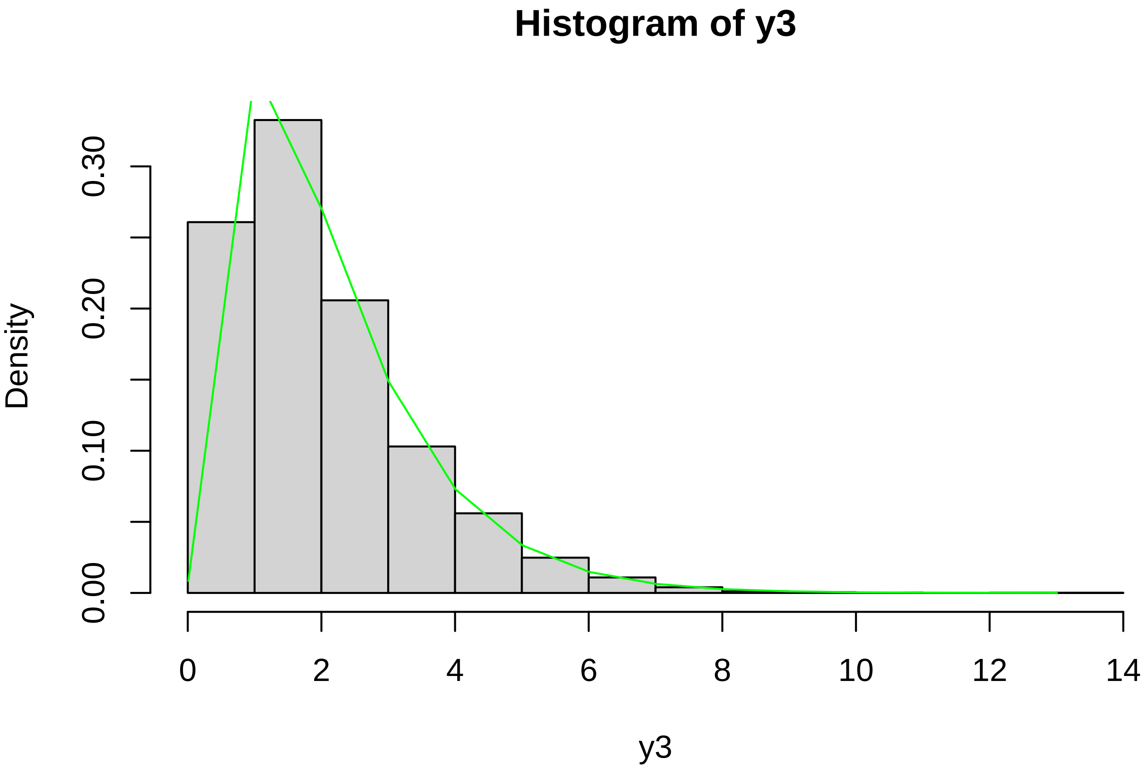
```
y1 <- runif(10000,0,1)
hist(y1, probability = TRUE)
xfit <- seq(min(y1), max(y1), 0.01)
yfit <- dunif(xfit, min = 0, max = 1 )
points(xfit, yfit, col="green")
```



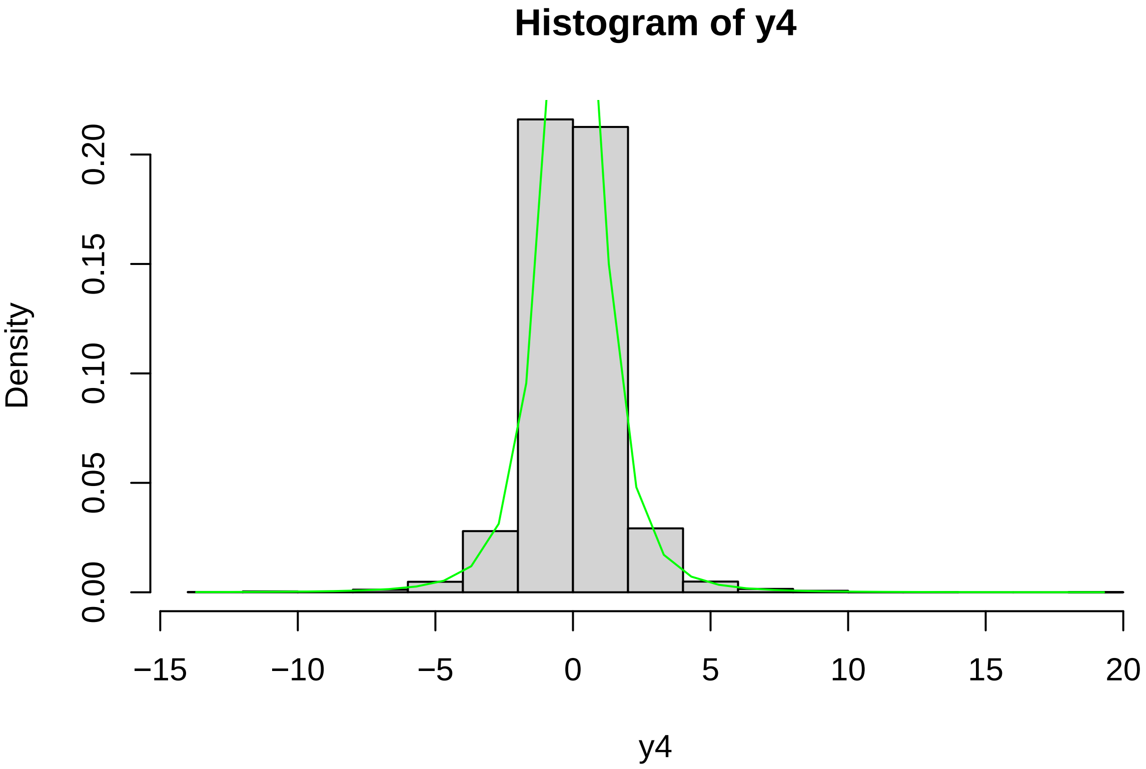
```
y2 <- rexp(10000,3)
hist(y2, probability = TRUE)
xfit <- seq(min(y2), max(y2), 1)
yfit <- dexp(xfit, rate = 3)
lines(xfit, yfit, col="green")
```



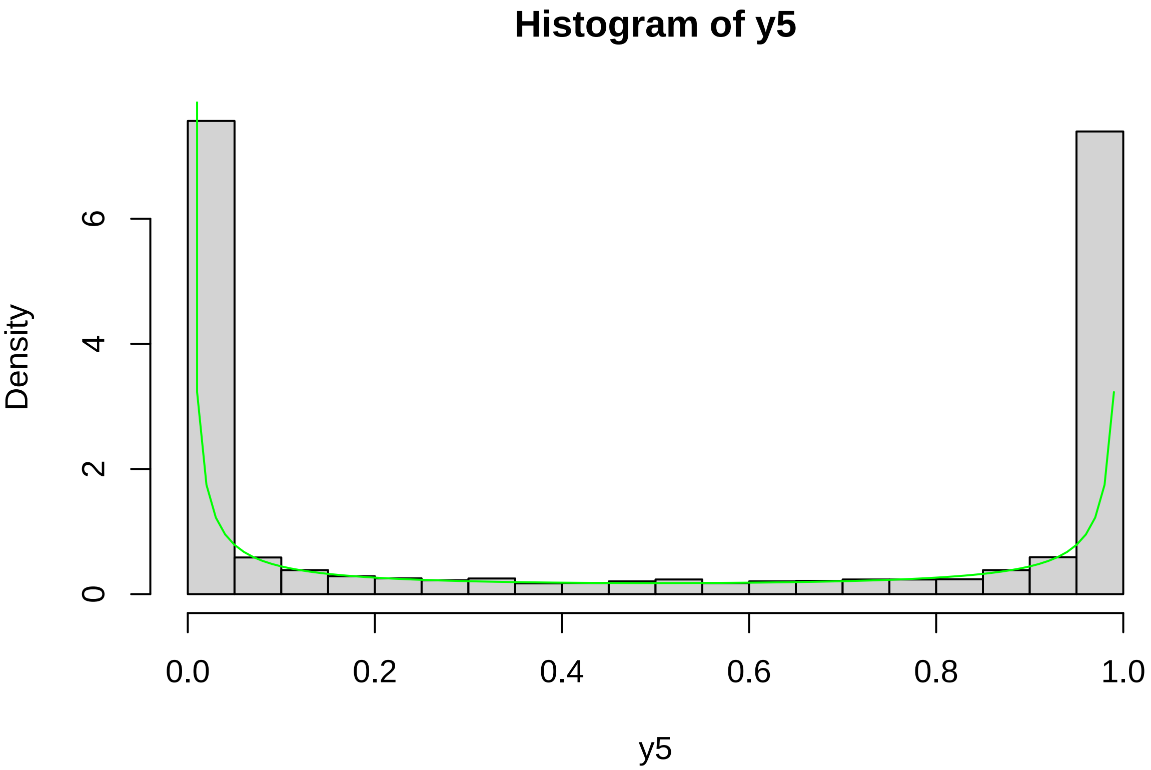
```
y3 <- rgamma(10000,2,1)
hist(y3, probability = TRUE)
xfit <- seq(min(y3), max(y3), 1)
yfit <- dgamma(xfit, shape = 2, rate = 1 )
lines(xfit, yfit, col="green")
```



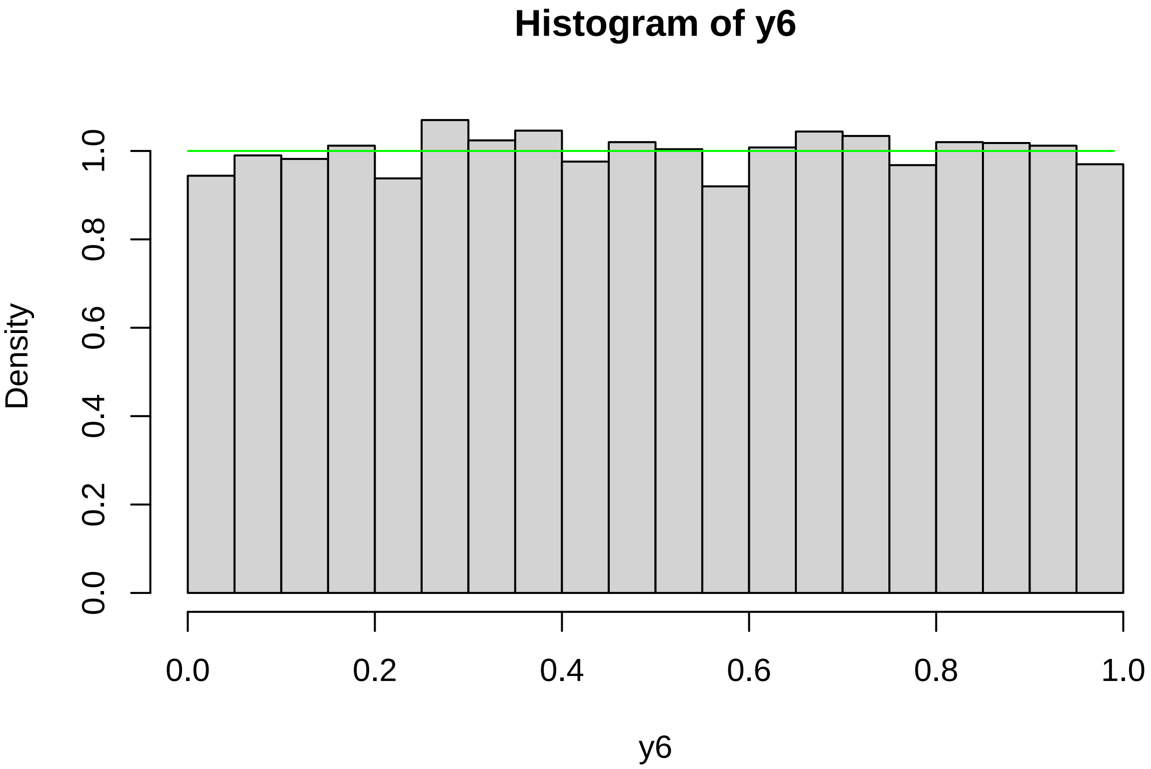
```
y4 <- rt(10000,3)
hist(y4, probability = TRUE)
xfit <- seq(min(y4), max(y4), 1)
yfit <- dt(xfit, df = 3)
lines(xfit, yfit, col="green")
```



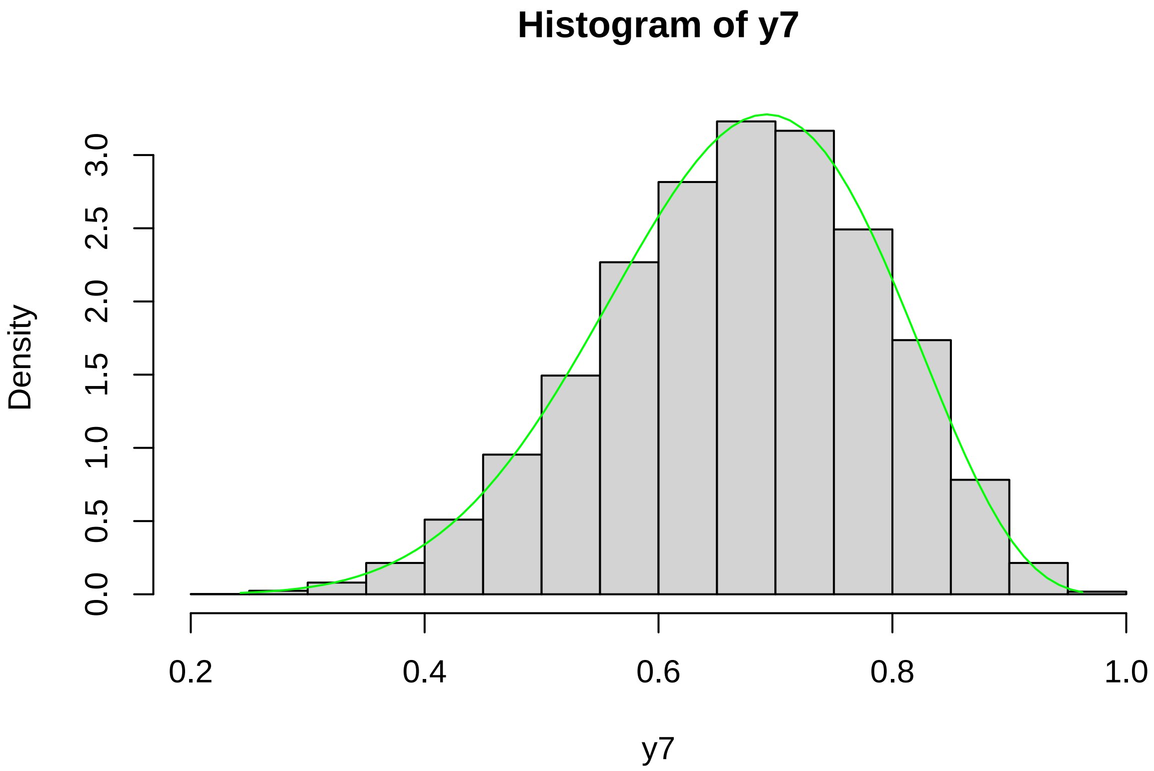
```
y5 <- rbeta(10000,shape1=0.1,shape2=0.1)
hist(y5, probability = TRUE)
xfit <- seq(min(y5), max(y5), 0.01)
yfit <- dbeta(xfit,shape1=0.1,shape2=0.1)
lines(xfit, yfit, col="green")
```



```
y6 <- rbeta(10000,1,1)
hist(y6, probability = TRUE)
xfit <- seq(min(y6), max(y6), 0.01)
yfit <- dbeta(xfit,shape1 = 1,shape2 =1)
lines(xfit, yfit, col="green")
```

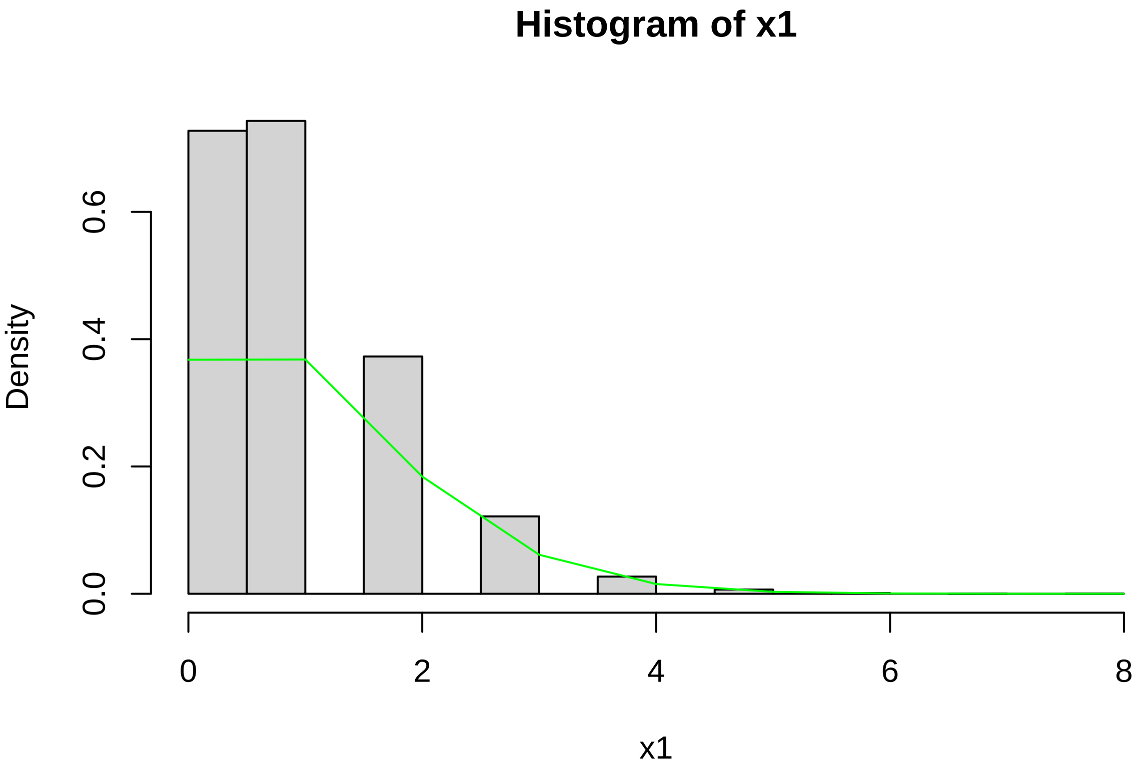


```
y7 <- rbeta(10000,10,5)
hist(y7, probability = TRUE)
xfit <- seq(min(y7), max(y7), 0.01)
yfit <- dbeta(xfit,shape1 = 10,shape2 =5)
lines(xfit, yfit, col="green")
```



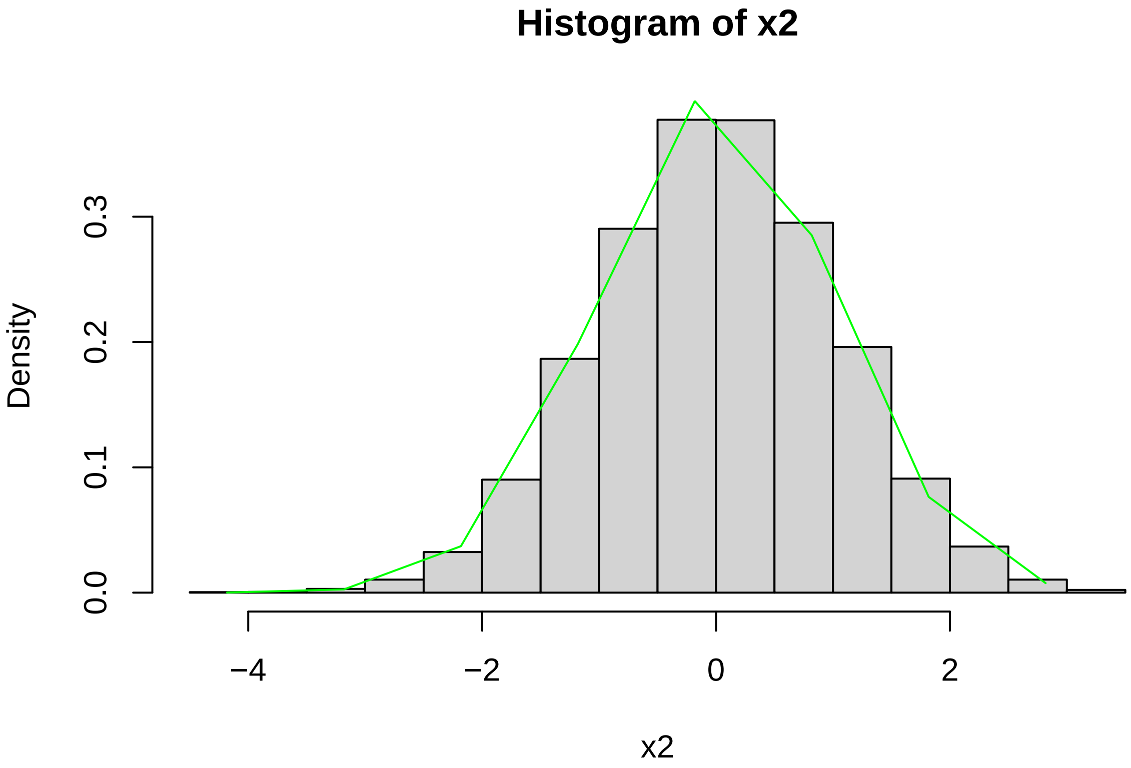
Uppgift 3.1.3

```
x1 <- rbinom(n= 10000, size = 1000 ,prob = 0.001 )
hist(x1, probability = TRUE)
xfit <- seq(min(x1), max(x1))
yfit <- dbinom(xfit, size= 1000, prob = 0.001)
lines(xfit, yfit, col="green")
```



(1)

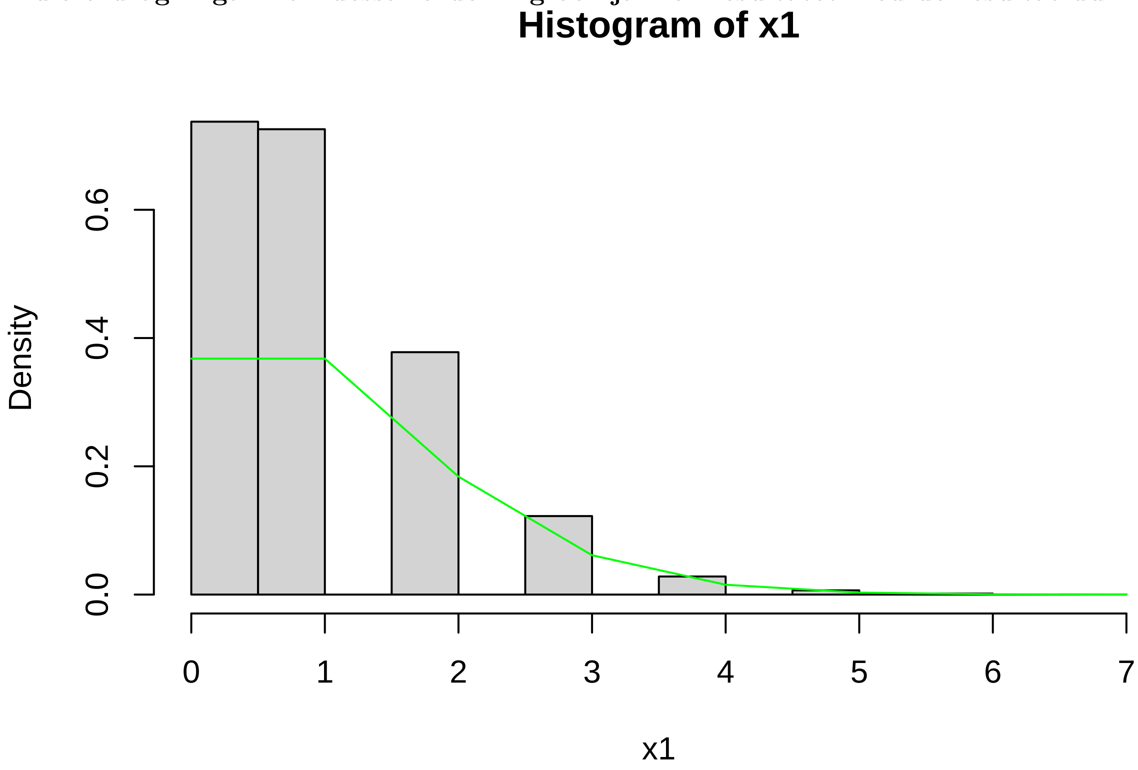
```r
x2 <- rt(n=10000, df=1000)
hist(x2, probability = TRUE)
xfit <- seq(min(x2), max(x2), 1)
yfit <- dt(xfit, df = 1000)
lines(xfit, yfit, col="green")
```



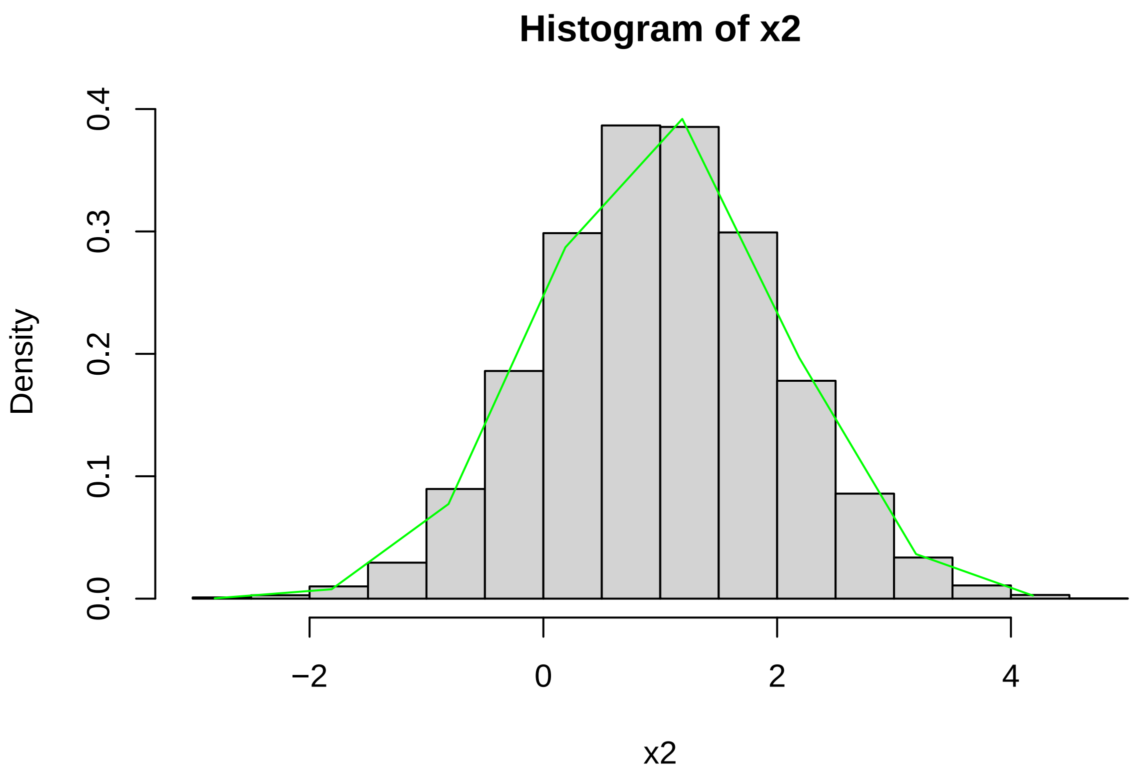
(2) Ta reda på (t ex via Wikipedia eller föreläsningarnas slides) vilken annan fördelning som respektive fördelning börjar konvergera mot. Binomial fördelning konvergerar mot Poisson fördelning Student t fördelningen kovergerar mot Standard Normal fördelning

```
x1 <- rpois(10000, 1)
hist(x1, probability = TRUE)
xfit <- seq(min(x1), max(x1))
yfit <- dpois(xfit, 1)
lines(xfit, yfit, col="green")
```

(3) Simulera dragningar från dessa fördelning och jämför resultatet med de resultat du fick i (1).



```
x2 <- rnorm(10000, 1)
hist(x2, probability = TRUE)
xfit <- seq(min(x2), max(x2))
yfit <- dnorm(xfit, 1)
lines(xfit, yfit, col="green")
```



Vi simulerade, och det är tydligt att de är väldigt lika.

Uppgift 3.1.4

```
x <- dbinom(0, size=10, prob = 0.1)
y <- rbinom(10000, size = 10, prob = 0.1)
print(paste("P(Y=0)", x))
```

1)

```
## [1] "P(Y=0) 0.3486784401"
```

```
print(paste("Simulated 10000 times", mean(y==0)))
```

```
## [1] "Simulated 10000 times 0.3508"
```

```
#P(X<0)
a <- pnorm(0, mean = 0, sd = 1)
print(a)
```

2)

```
## [1] 0.5
```

```
#P(X<0)
b <- pnorm(1, mean = 0, sd = 1)-(pnorm(-1, mean = 0, sd = 1))
print(b)
```

```
## [1] 0.6826895
```

```
#P(1.096<X)
c <- 1-pnorm(1.96, mean = 0, sd = 1)
print(c)
```

```
## [1] 0.0249979
```

```
#P(X<0)
d <- pbinom(10, 10, 0.1) - pbinom(0, 10,  0.1)
print(d)
```

```
## [1] 0.6513216
```

```
#P(X=0)
e <- pbinom(-0.0001,10, 0.1) + pbinom(0.0001, 10,  0.1)
print(e)
```

```
## [1] 0.3486784
```

```
#P(1.096<X)
f <- e + d
print(f)
```

```
## [1] 1
```

```

x <- rnorm(10000, 0, 1)
y <- rbinom(10000, 10, 0.1)

p1 <- sum(x < 0) / 10000
print(p1)
```

3) Beräkna samma sannolikheter som i (2) men genom att simulera dragningar från X och Y i R.

```
## [1] 0.4974
```

```
p2 <- (sum(x < 1) - sum(x <= -1)) / 10000
print(p2)
```

```
## [1] 0.6777
```

```
p3 <- sum(x > 1.96) / 10000
print(p3)
```

```
## [1] 0.0274
```

```
p4 <- (sum(y < 10) - sum(y <= 0)) / 10000
print(p4)
```

```
## [1] 0.6507
```

```
p5 <- sum(y == 0) / 10000
print(p5)
```

```
## [1] 0.3493
```

```
p6 <- (sum(y <= 10) - sum(y < 0)) / 10000
print(p6)
```

```
## [1] 1
```

3.1.5 Berakna (icke-triviala) sannolikheter.

(1)

```
# Old system
x_old <- rbinom(10000, 337, 0.1)
print(sum(x_old)/10000)
```

```
## [1] 33.7945
```

```
# New system
p <- sum(runif(10000, 0.02, 0.16)) / 10000
x_new <- rbinom(10000, 337, p)
print(sum(x_new) / 10000)
```

```
## [1] 30.3367
```

(2)

```
print(sum(x_old > x_new)/10000)
```

```
## [1] 0.6514
```

(3)

```
print(sum(x_old > 50)/10000)
```

```
## [1] 0.0017
```

```
print(sum(x_new > 50)/10000)
```

```
## [1] 2e-04
```

3.2.1 Stora talens lag

(1)

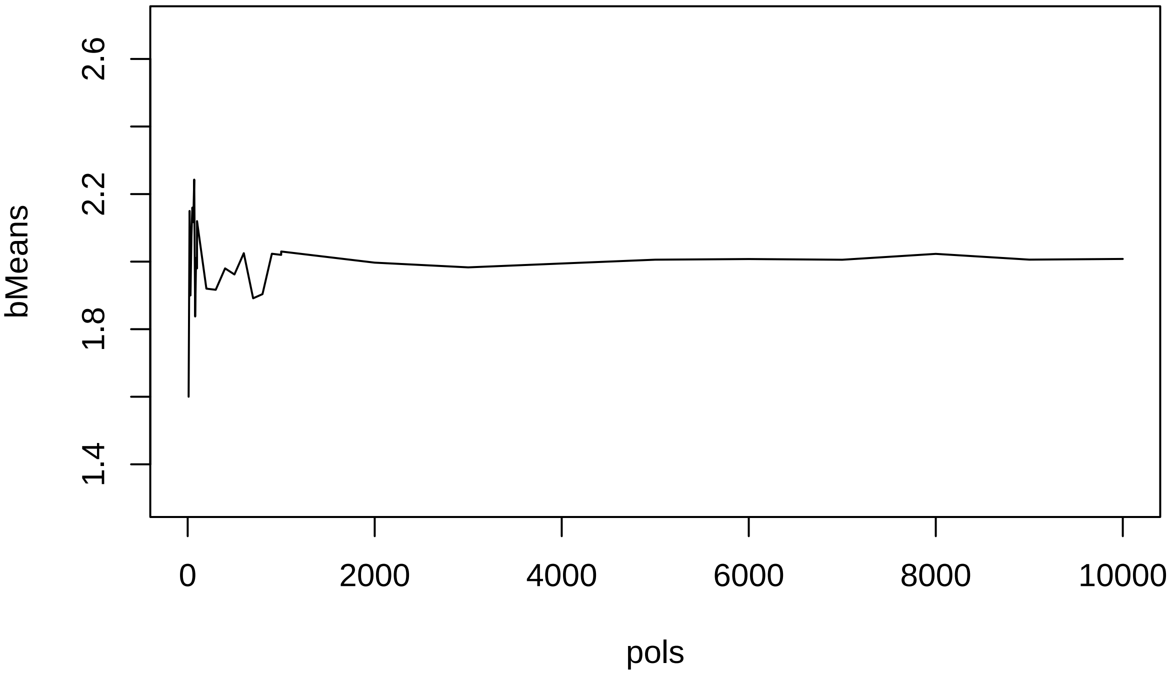
E(x) = N*P* = *10* 0.2 = 2

E(y) = a/b = 2/2 = 1

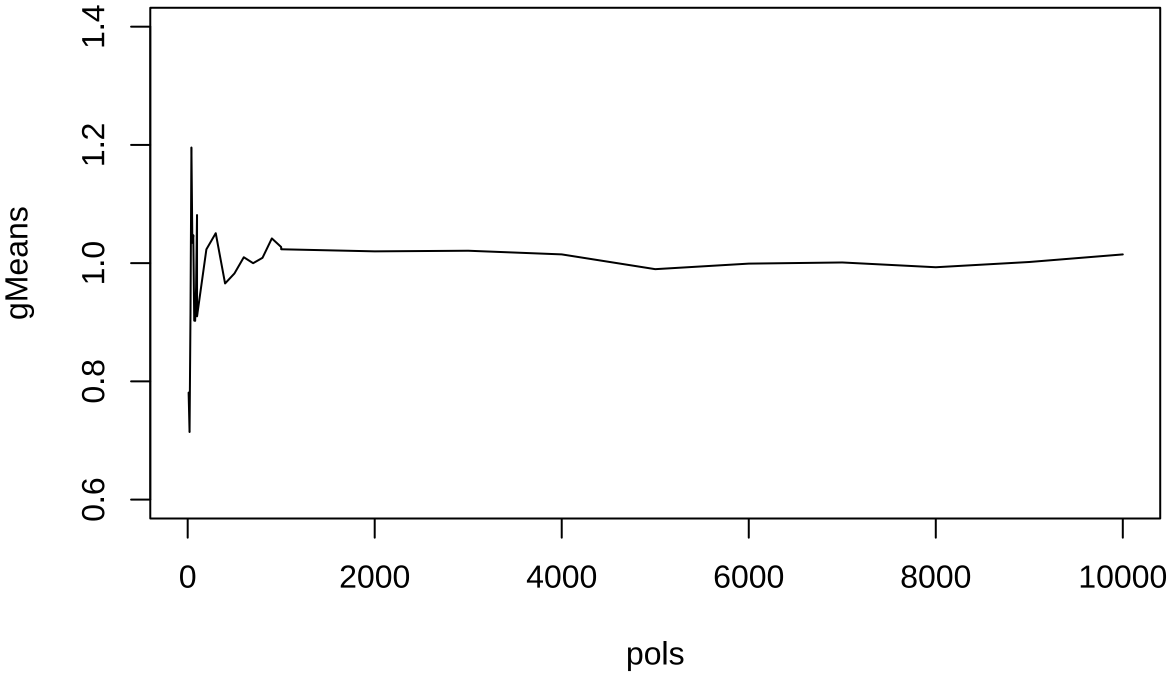
(2)

```
# Sequence of draws
pols <- c(seq(10,100,10), seq(100,1000,100), seq(1000,10000,1000))
#  List of means
bMeans <- numeric(length(pols))
gMeans <- numeric(length(pols))

for ( x in 1:length(pols)){
  n <- pols[x]
  gMeans[x] <- mean(rgamma(n,2,2))
  bMeans[x] <- mean(rbinom(n,10,0.2))
}
plot(x=pols,y=bMeans,  xlim=c(0,10000) , ylim = c(1.3,2.7) ,type="l")
```



```
plot(x=pols,y=gMeans,  xlim=c(0,10000) , ylim = c(0.6,1.4) ,type="l")
```



3.3.1

(1)

E(X) = 1 / 10 = 0.1 Var(X) = 1 / (10^2) = 0.01 E(Y) = 3 Var(Y) = 3

(2) Simulera 10 000 varden

```
x <- rexp(10000, 10)
print(mean(x))
```

```
## [1] 0.1007093
```

```
print(var(x))
```

```
## [1] 0.01028045
```

```
y <- rpois(10000,3)
print(mean(y))
```

```
## [1] 2.9822
```

```
print(var(y))
```

```
## [1] 2.933176
```

(3)

E(3) = 3

E(3$X + 2) = E(3x) + E(2) = 3$ E(x) + 2 = 2.3

E(x+y) = 0.1 + 3 = 3.1

E(x$y) = 0.1$ 3 = 0.3

E(3$x + 2$y - 3) = 3 * 0.1 + 2 * 3 - 3 = 3.3

Var(2 * x -5) = 2^2 * Var(x) = 0.01 * 4 = 0.01

Var(x+y) = 0.01 + 3 = 3.01

```
print(mean(3))
```

```
## [1] 3
```

```
print(mean(3*x +2))
```

```
## [1] 2.302128
```

```
print(mean(x + y))
```

```
## [1] 3.082909
```

```
print(mean(x*y))
```

```
## [1] 0.2990195
```

```
print(mean(3*x +2*y -3))
```

```
## [1] 3.266528
```

```
print(var(2*x -5))
```

```
## [1] 0.04112178
```

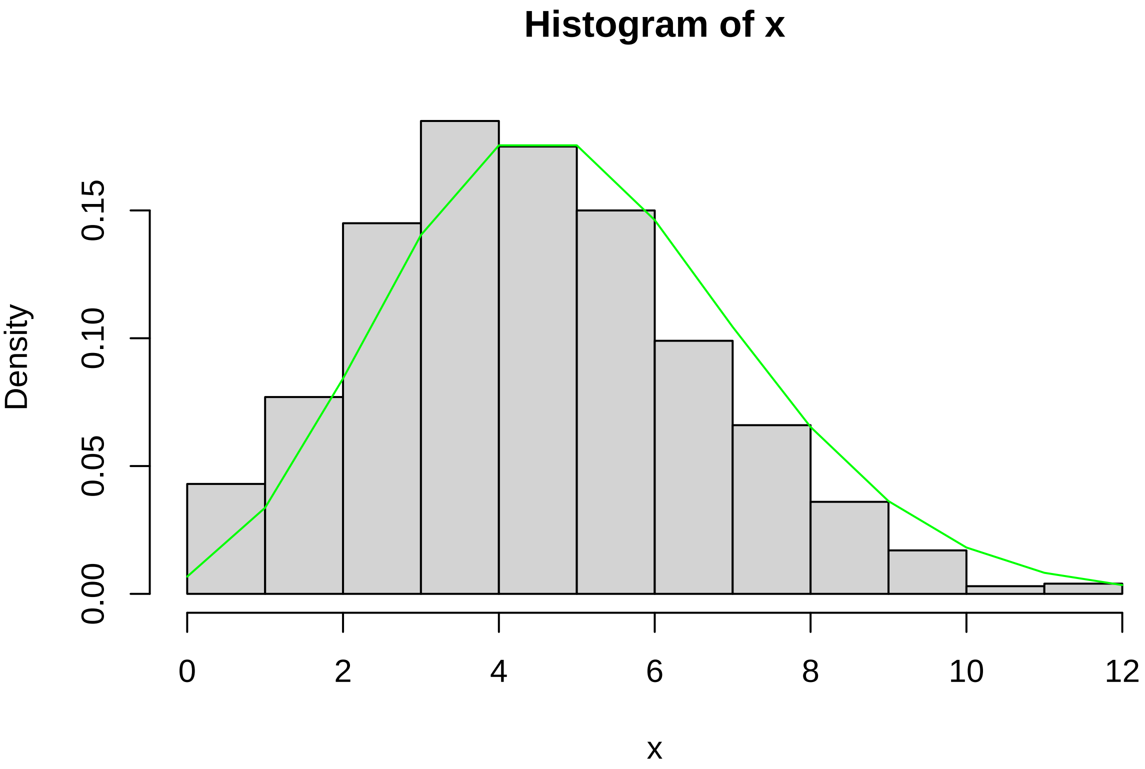
```
print(var(x + y))
```

```
## [1] 2.940825
```

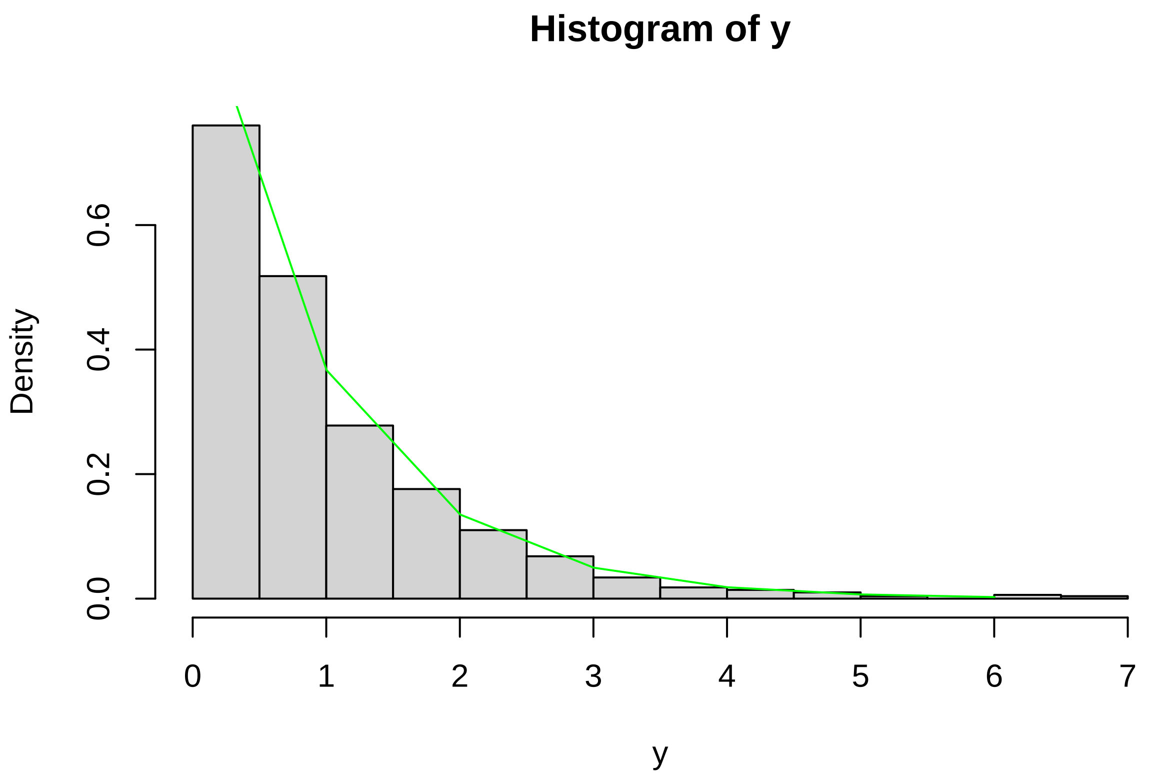
3.4.1

(1)

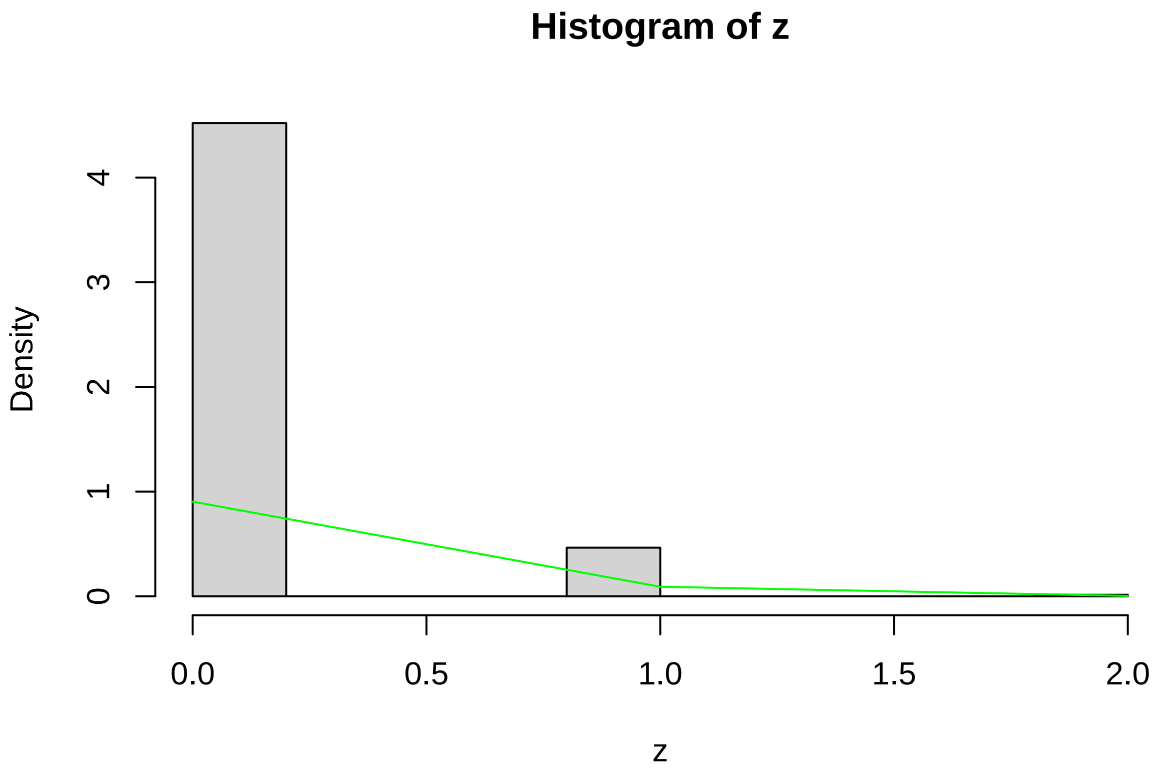
```
x <- rpois(1000, 5)
hist(x, probability = TRUE)
xfit <- seq(min(x), max(x), 1)
yfit <- dpois(xfit, 5)
lines(xfit, yfit, col="green")
```



```
y <- rexp(1000, 1)
hist(y, probability = TRUE)
xfit <- seq(min(y), max(y), 1)
yfit <- dexp(xfit, 1)
lines(xfit, yfit, col="green")
```



```
z <- rbinom(1000, 10 ,0.01)
hist(z, probability = TRUE)
xfit <- seq(min(z), max(z), 1)
yfit <- dbinom(xfit, 10, 0.01)
lines(xfit, yfit, col="green")
```



```
print(var(x))
```

```
## [1] 4.654559
```

```
print(var(y))
```

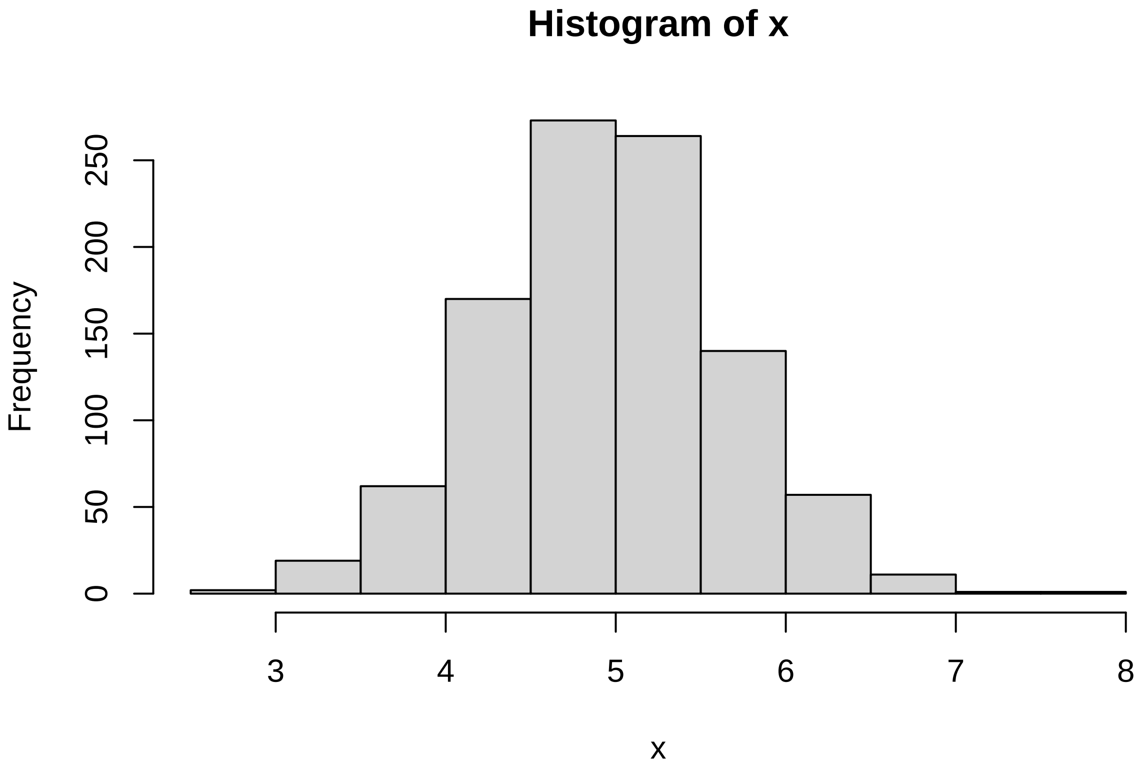
```
## [1] 0.9796795
```

```
print(var(z))
```

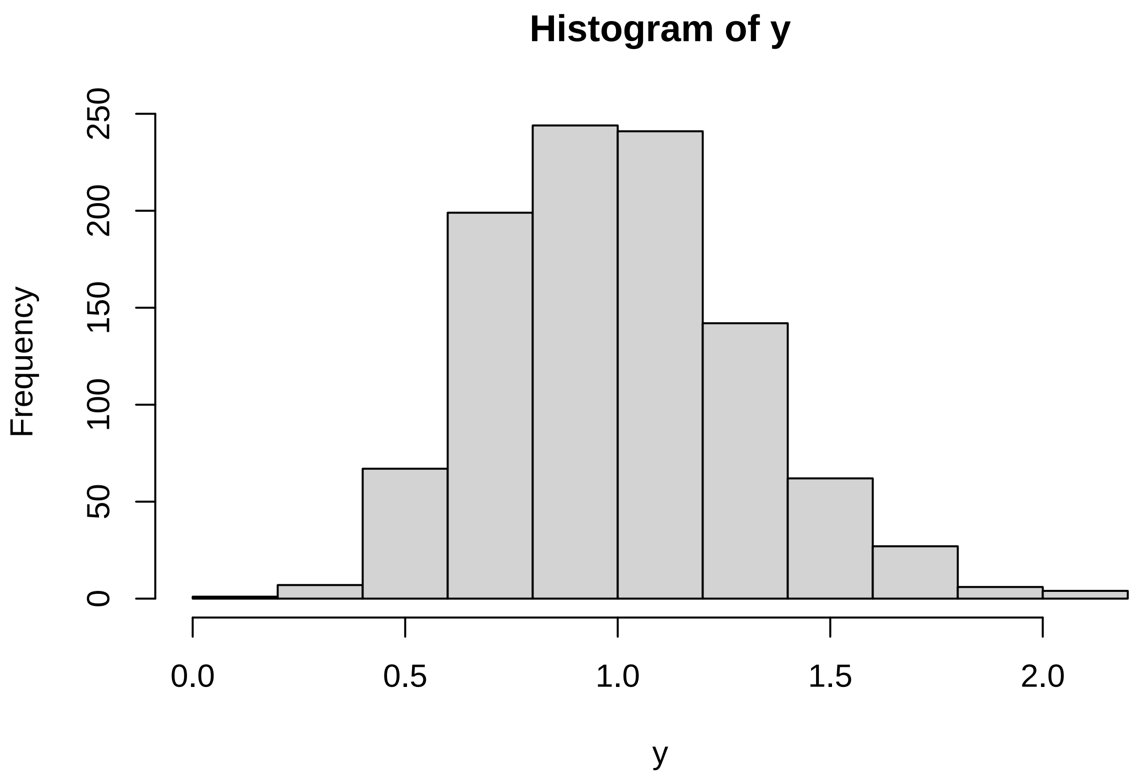
```
## [1] 0.09529429
```

(2) Skriv en loop/funktion

```
x <- numeric(0)
y <- numeric(0)
for(i in 1:1000){
  x <- c(x, mean(rpois(10,5)))
  y <- c(y, mean(rexp(10,1)))
}
hist(x)
```

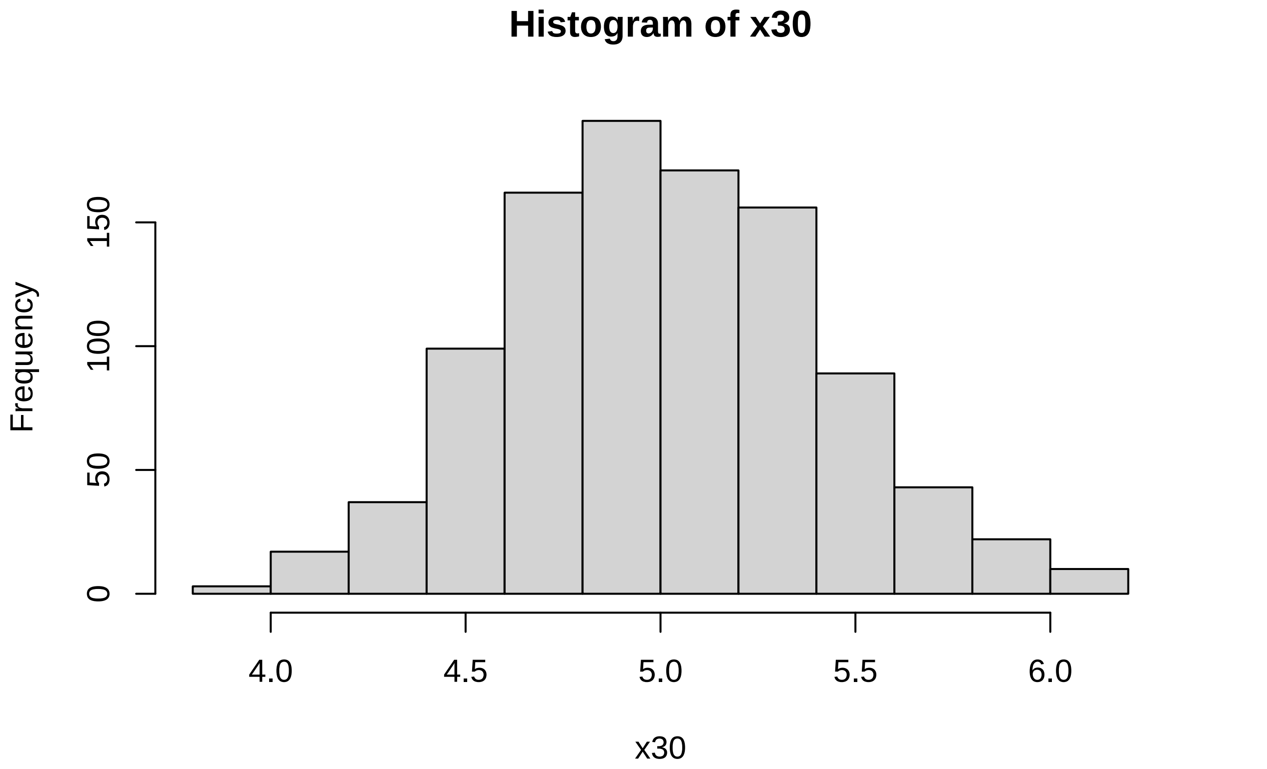


```
hist(y)
```

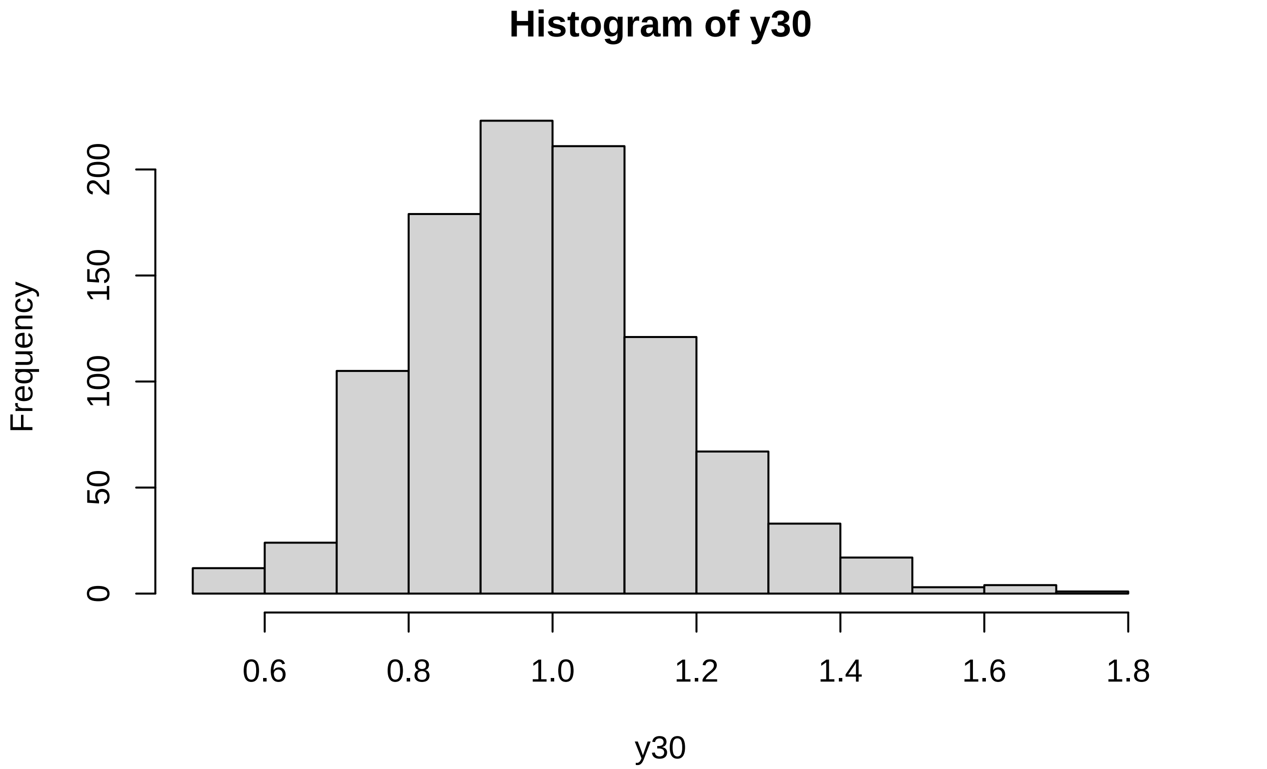


(3)

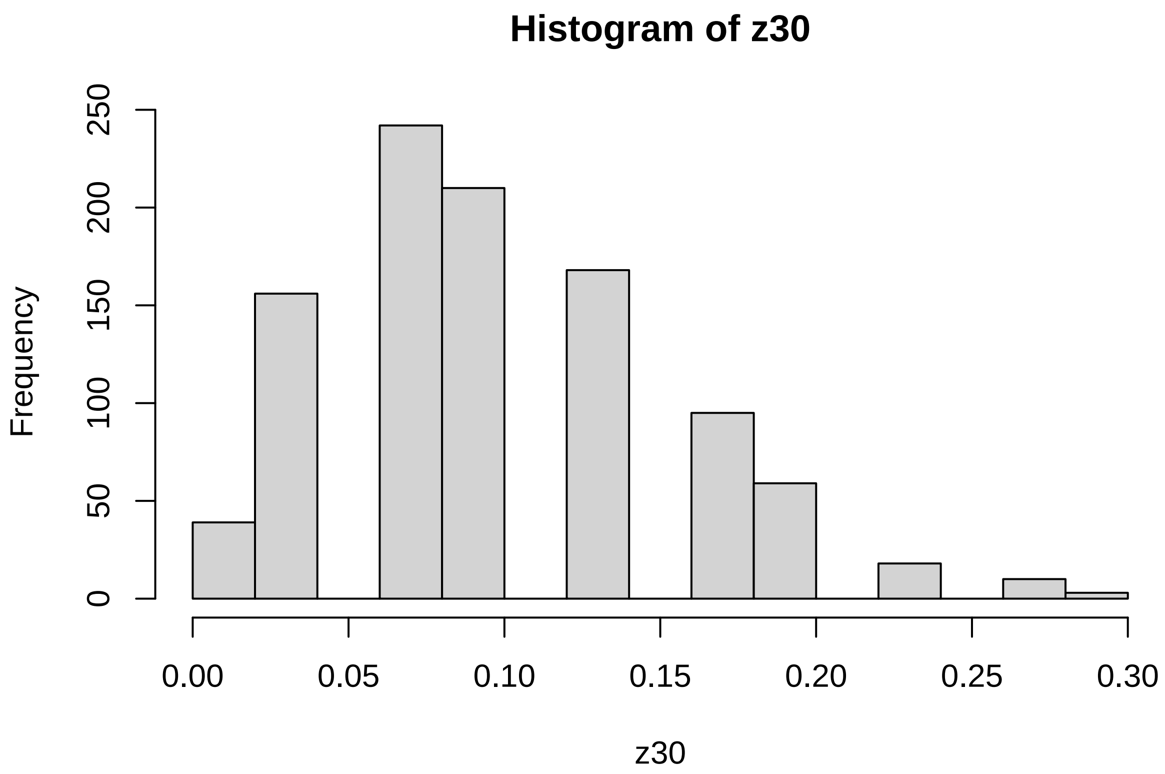
```
# 30 values
x30 <- numeric(0)
y30 <- numeric(0)
z30 <- numeric(0)
for(i in 1:1000){
  x30 <- c(x30, mean(rpois(30,5)))
  y30 <- c(y30, mean(rexp(30,1)))
  z30 <- c(z30, mean(rbinom(30,10,0.01)))
}
hist(x30)
```



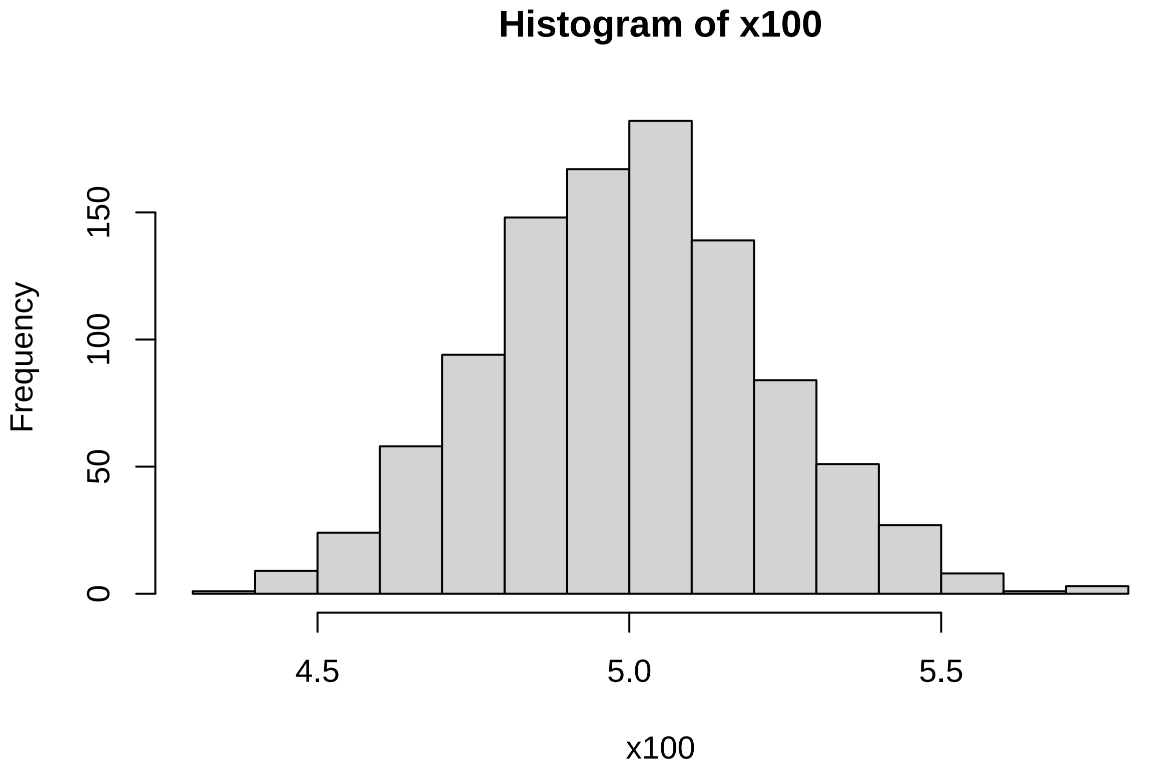
```
hist(y30)
```



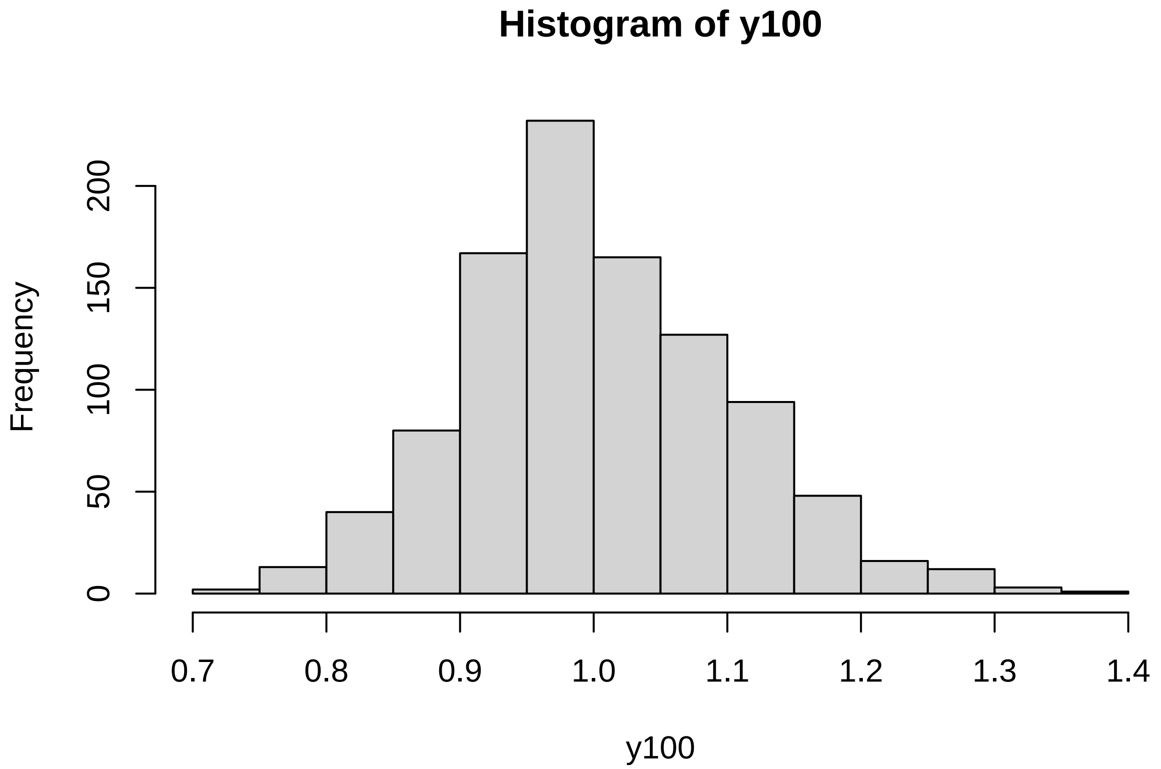
```
hist(z30)
```



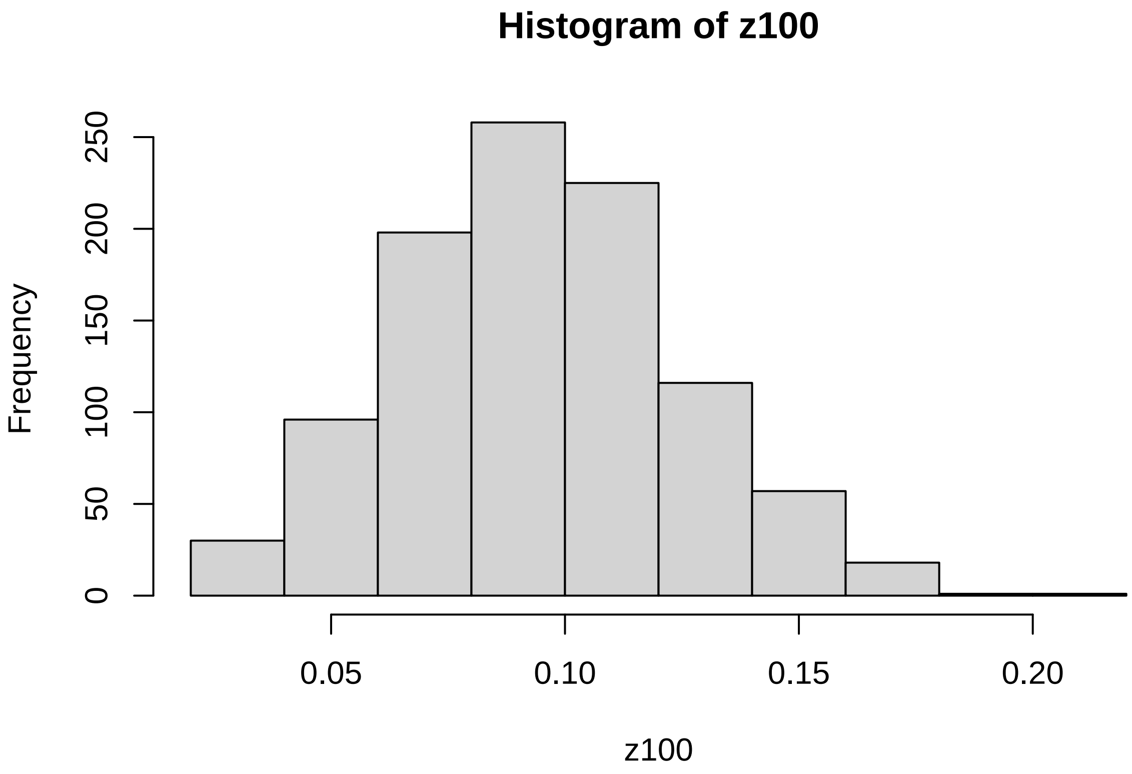
```
# 100 values
x100 <- numeric(0)
y100 <- numeric(0)
z100 <- numeric(0)
for(i in 1:1000){
  x100 <- c(x100, mean(rpois(100,5)))
  y100 <- c(y100, mean(rexp(100,1)))
  z100 <- c(z100, mean(rbinom(100,10,0.01)))
}
hist(x100)
```



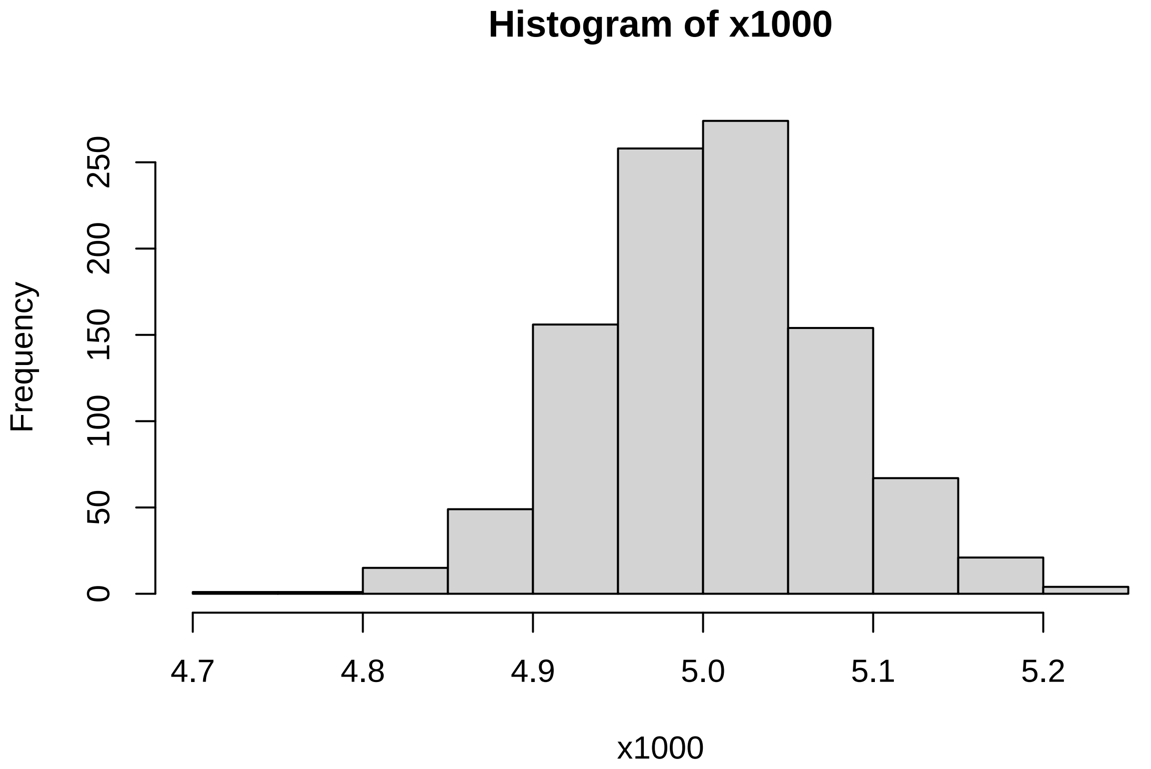
```
hist(y100)
```



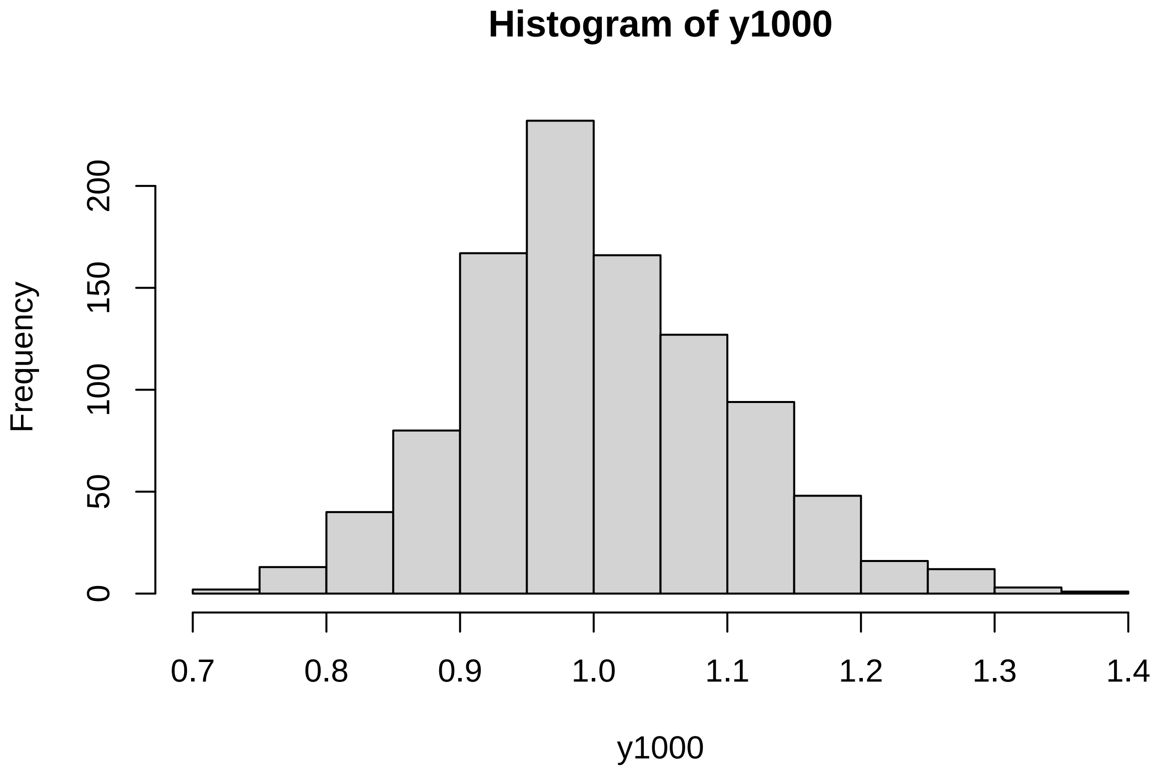
```
hist(z100)
```



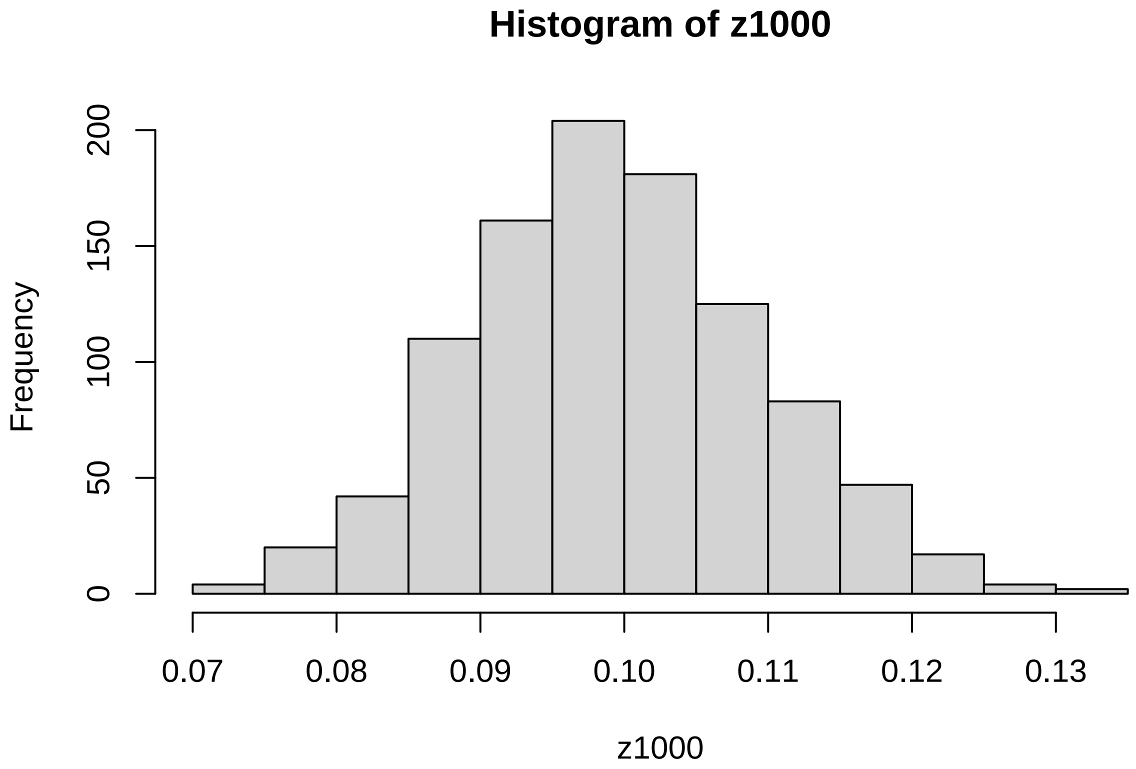
```
# 1000 value
x1000 <- numeric(0)
y1000 <- numeric(0)
z1000 <- numeric(0)
for(i in 1:1000){
  x1000 <- c(x1000, mean(rpois(1000,5)))
  y1000 <- c(y100, mean(rexp(1000,1)))
  z1000 <- c(z1000, mean(rbinom(1000,10,0.01)))
}
hist(x1000)
```



```
hist(y1000)
```



```
hist(z1000)
```



Vi kan se att Z har lägst varians. Detta gör att kurvan blir smalare (pga mindre spridning), och därav så konvergerar Z snabbare mot en normalfördelning

Var(X) = 5

Var(Y) = 1/(1^2) = 1

Var(Z) = 10 * 0.01 = 0.1