3701 HW1

Dongnan Liu

2024-02-11

```r
#Question 1(b)
set.seed(3701)
n= 10
theta = 0.6
reps=1e4
ysum.list = numeric(reps)
for(r in 1:reps){
  #generate a binomial distribution
  y.list = rbinom(n=n,size=1,prob=theta)
  #use ysum.list to contain each sum of y.list
  ysum.list[r] = sum(y.list)
}
#take mean of realizations
mean(ysum.list)
```

```
## [1] 6.0098
```

```r
#take variance of realizations
var(ysum.list)
```

```
## [1] 2.417946
```

```r
theoretical_mean = n*theta
theoretical_var = n*theta*(1-theta)
#compare the simulation-based estimated mean and variance and correspond
ing values from the formulas
c(mean(ysum.list),theoretical_mean)
```

```
## [1] 6.0098 6.0000
```

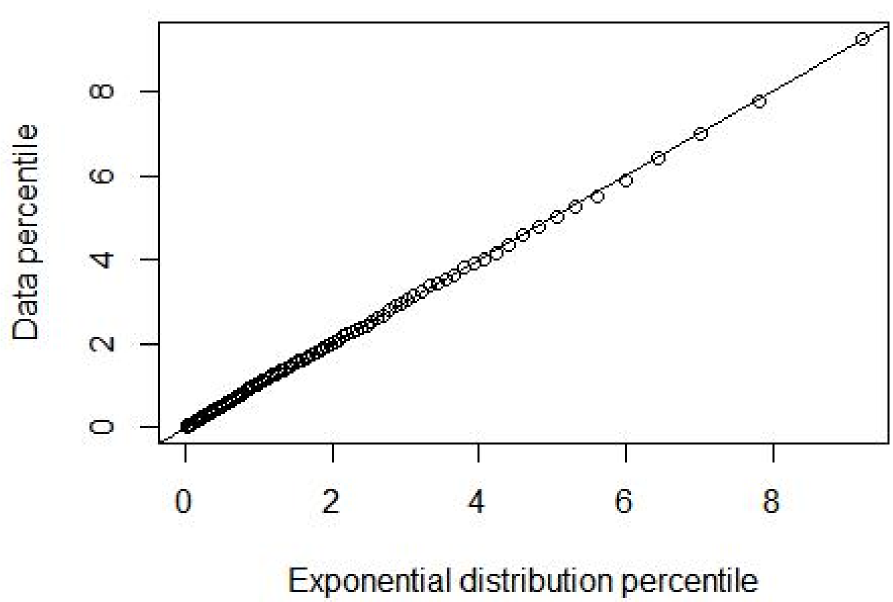
```r
c(var(ysum.list),theoretical_var)
```

```
## [1] 2.417946 2.400000
```

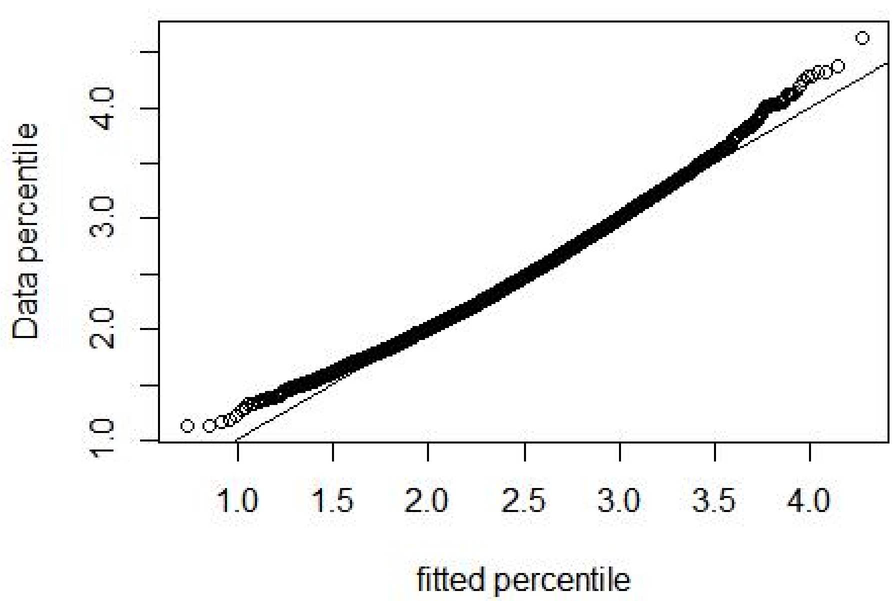
```r
#Question 1(e)
set.seed(3701)
reps=1e4
n=10
theta=0.6
ybar.list = numeric(reps)
for(r in 1:reps){
#generate a binomial distribution
y.list = rbinom(n=n,size=1,prob=theta)
```

```
#use ybar.list to contain each mean of y.list
ybar.list[r] = mean(y.list)
}
#take mean of realizations
mean(ybar.list)

## [1] 0.60098

#take variance of realizations
var(ybar.list)

## [1] 0.02417946

theoretical_mean2 = theta
theoretical_var2 = theta*(1-theta)/n
#compare the simulation-based estimated mean and variance and correspond
ing values from the formulas
c(mean(ybar.list),theoretical_mean2)

## [1] 0.60098 0.60000

c(var(ybar.list),theoretical_var2)

## [1] 0.02417946 0.02400000

#Question 1(f)
set.seed(3701)
n.list = c(35, 40, 55, 100, 200)
reps = 1e4
theta = 0.6
mu = theta
sigma = sqrt(theta*(1-theta))
#Allocate memory for simulation results
sim_results<-matrix(data=NA,nrow=reps,ncol=length(n.list))
for(i in 1:length(n.list)){
  n = n.list[i]
  for(j in 1:reps){
    #get realization of binomial random variables
    x.list = rbinom(n=n,size=1,prob=theta)
    #calculate margin of error
    moe=1/sqrt(n)
    #mean of realizations
    xbar = mean(x.list)
    #save simulation results
    sim_results[j,i]=((xbar+moe>mu & (xbar-moe<=mu)))
  }
}
cbind(n.list,colMeans(sim_results))

##      n.list
## [1,]     35 0.9456
## [2,]     40 0.9622
```

```
## [3,]     55 0.9636
## [4,]    100 0.9595
## [5,]    200 0.9652

#We can see the result that the coverage probability of this random inte
rval approximately equal to 0.95 for these values of n and ϑ=0.6.

#Question 2(a)
#n is the sample size, mu is the mean of the exponential distribution
myrexp = function(n,mu)
{u.list = runif(n=n)
#generate a realization of exponential distribution
x.list = -mu*log(1-u.list)
return(x.list)
}

#Question 2(b)
n=2e3
mu = 2
x.list=myrexp(n=n,mu=mu)
probs = seq(from=0.01,to =0.99,by = 0.01)
dataperc = quantile(x.list,probs)
#make a QQ plot
plot(-mu*log(1-probs),dataperc,xlab="Exponential distribution percentil
e",ylab="Data percentile")
abline(0,1)
```



```r
#The QQ plot does not show any problems with our myrexp function from R

#Question 2(c)
run.exp.sim = function(n,mu,reps)
{
  #generate a realization of reps independent copies of Xbar
  x.mat=matrix(myrexp(n=(n*reps),mu=mu),nrow=reps,ncol=n)
  xbar.list=apply(x.mat,1,mean)
  probs=ppoints(reps)
  #data percentiles of the entries in this vector
  dataperc=quantile(xbar.list,probs)
  #comparing the data percentiles of the entries in this vector to the p
ercentiles of the fitted Normal distribution
  plot(qnorm(probs,mean=mean(xbar.list),sd=sd(xbar.list)),dataperc,xlab
="fitted percentile",ylab="Data percentile")
  abline(0,1)
  return(xbar.list)
}

#Question 2(d)
x.list=run.exp.sim(n=30, mu=2.5, reps=1e4)
```




```r
# We can see in the QQ plot that sample size of 30 is not large enough f
or xbar to be a realization of random variable with a distribution well
approximated by the Normal distribution with sample size of 30 when μ =
2.5 since it is not exactly comply with the line y=x.
```