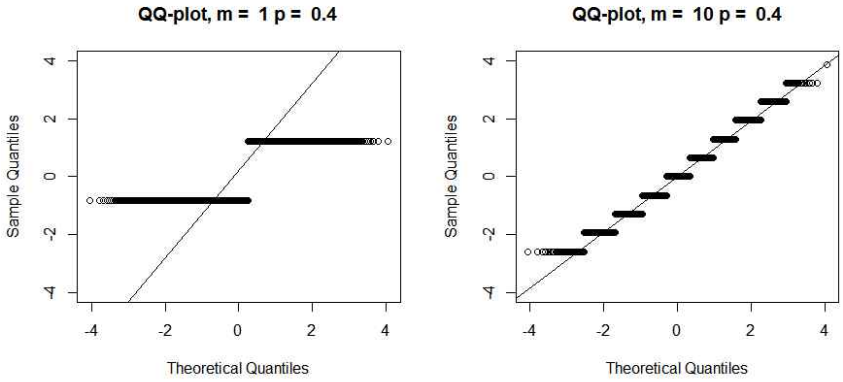
1. X가 모수 m, p인 이항분포를 따를 때 $Z = \frac{X-mp}{\sqrt{mp(1-p)}}$는 m이 커지면 근사적으로 표준정규분포를 따른다.

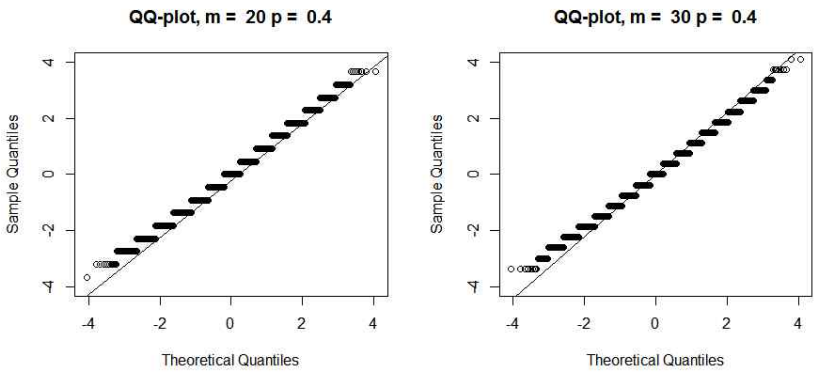
```
p = 0.4
for (m in 1:100)
{
    z = (rbinom(20000, size=m, prob=p) - m*p)/sqrt(m*p*(1-p))
    qqnorm(z, ylim=c(-4,4), main=paste("QQ-plot, m = ", m, "p = ", p))
    qqline(z)
}
```

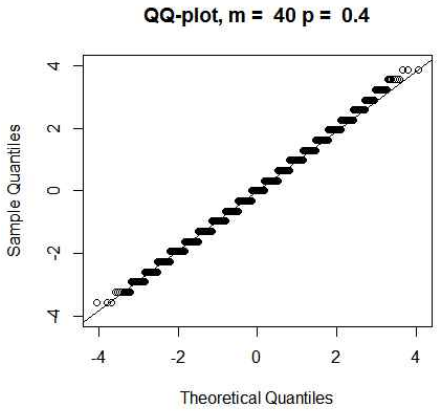
(a) 이 코드를 실행하여 m값이 커짐에 따라 Z의 분포가 어떻게 변하는지 설명하라.

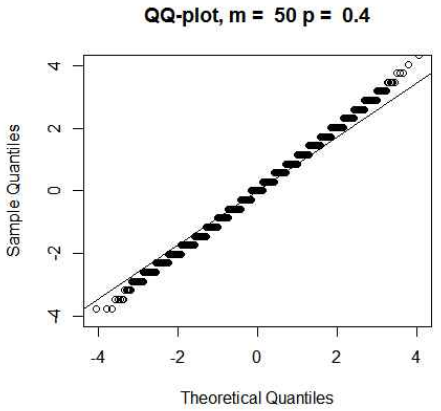
(b) $p = 0.2, 0.05$에 대하여 실행해보아라. m이 어느 정도 커지면 QQ-plot에서 거의 직선이 되는가? 모든 경우에 $m = 100$이면 충분한가?

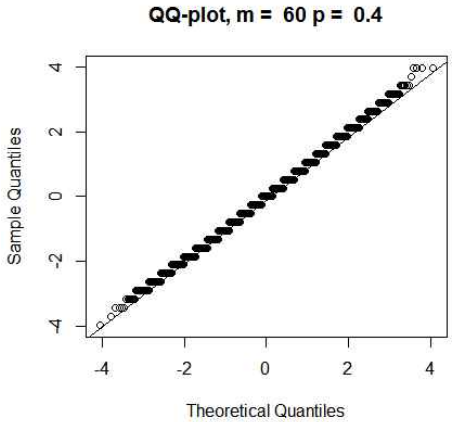
- m값이 커짐에 따라 QQ-Plot 가 직선에 가까워지는 것을 볼 수 있는데, 이를 통해서 m이 커지면 Z는 표준정규분포에 가까워짐을 알 수 있습니다.

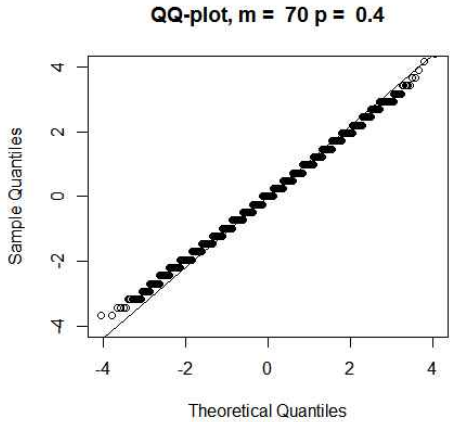


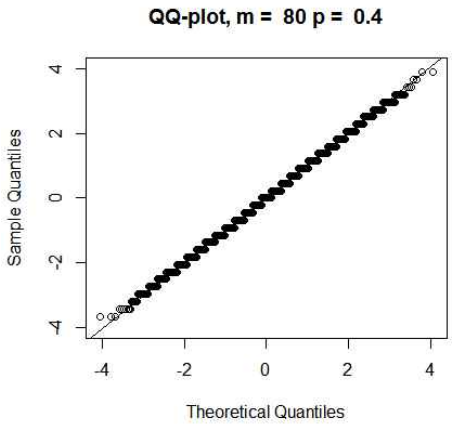


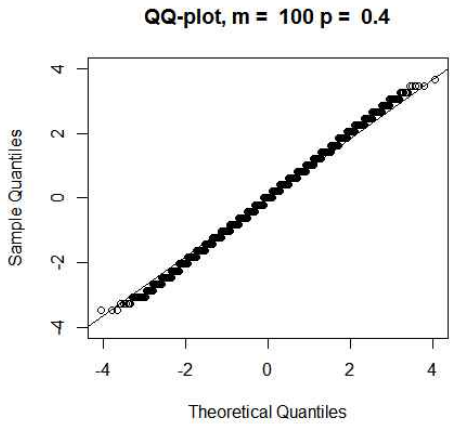




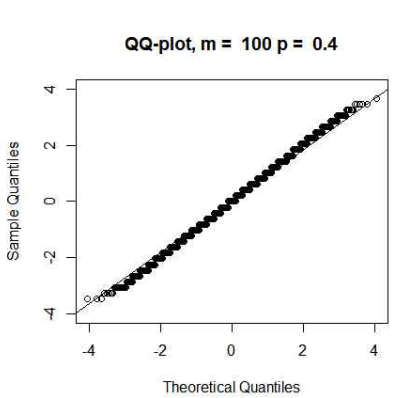


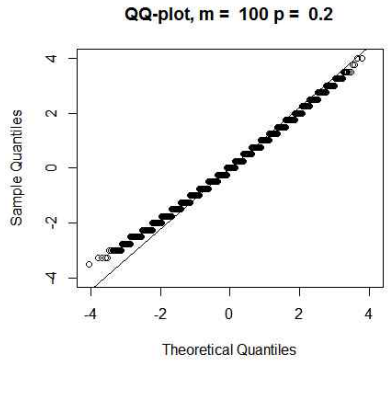


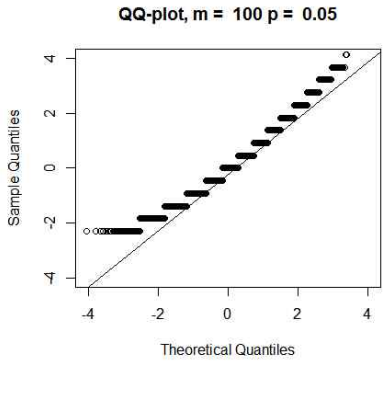




- 다음은 m = 100으로 동일하고 p = 0.2 일 때와 p = 0.05 일때의 QQ-Plot입니다.

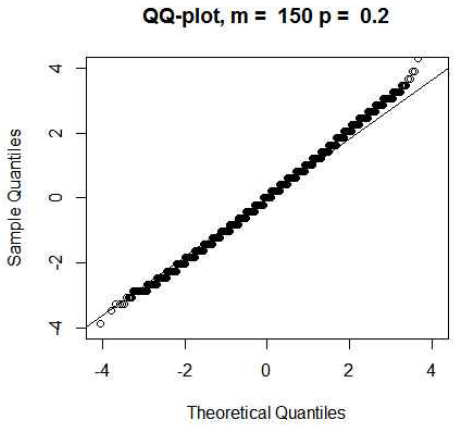


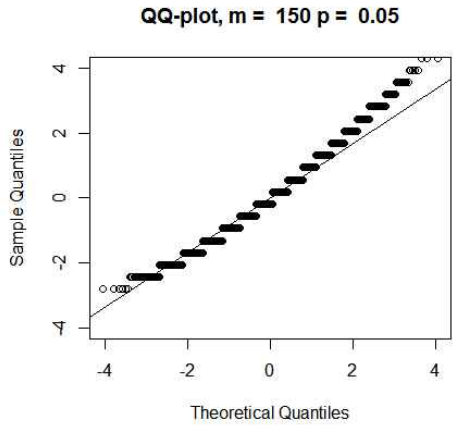




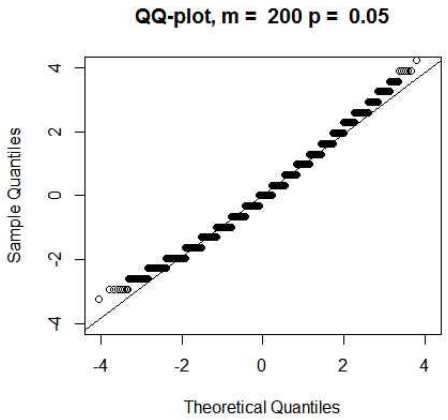
- 위의 그래프를 비교하면, p = 0.4일 때는 거의 직선처럼 보이고, p = 0.05일 때는 구조상 직선으로 보기 어

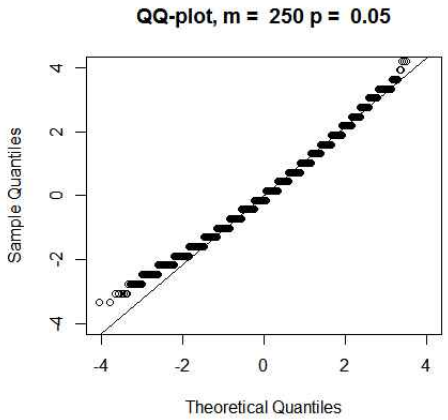
렵습니다. 그리고 p = 0.2 일 때는 직선에 가까운 정도가 두 경우의 중간 정도 되는 것으로 보입니다. 따라서, p의 값이 클수록 직선에 더 가깝고, p가 작을수록 끊어진 직선 형태를 보인다는 사실을 알 수 있습니다. p = 0.2 일 때와 0.05 일 때 모두 m의 값이 100으로는 충분하지 않았습니다. 그래서, p를 감소시켰을 때 m이 어느 정도 커져야 직선에 가까워지는지 m을 증가시켜 보았습니다.

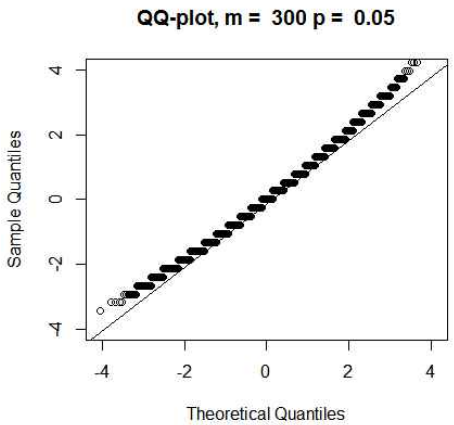


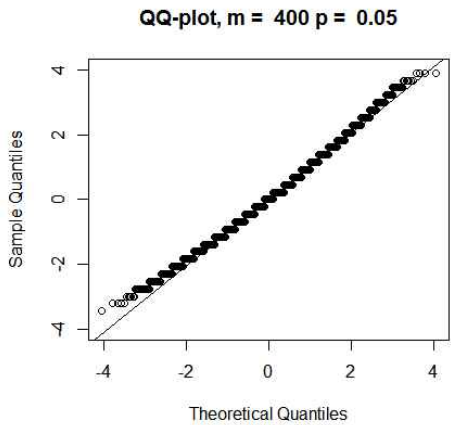


- p = 0.2 일 때는 m값이 150정도만 되어도 직선에 가까워지는 것을 확인할 수 있습니다. 따라서, p = 0.05 일 때만 m의 값을 증가시켜 보았습니다.

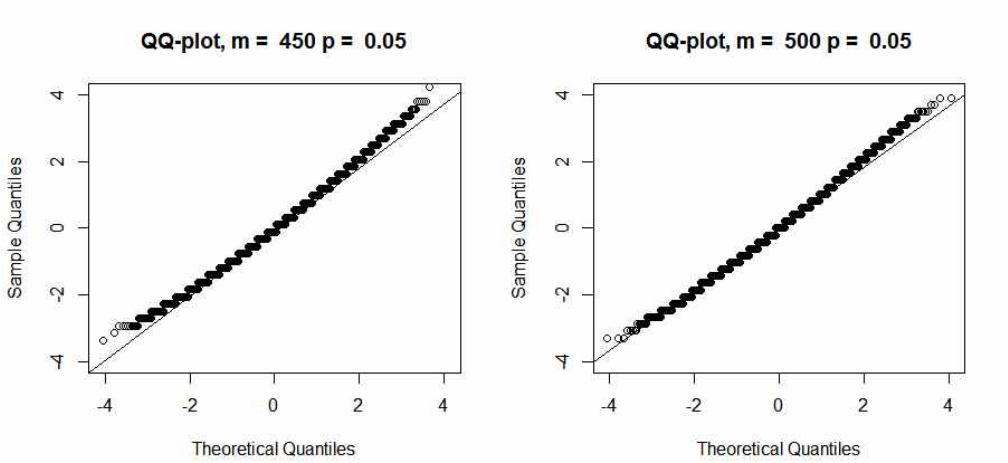








- p = 0.05 일 때 m의 값이 500정도만 되면 직선에 가까워짐을 확인할 수 있습니다. 따라서, p의 값이 작을



때는 QQ-Line가 직선에 가까워지려면 더 큰 값의 m이 필요함을 알 수 있습니다.

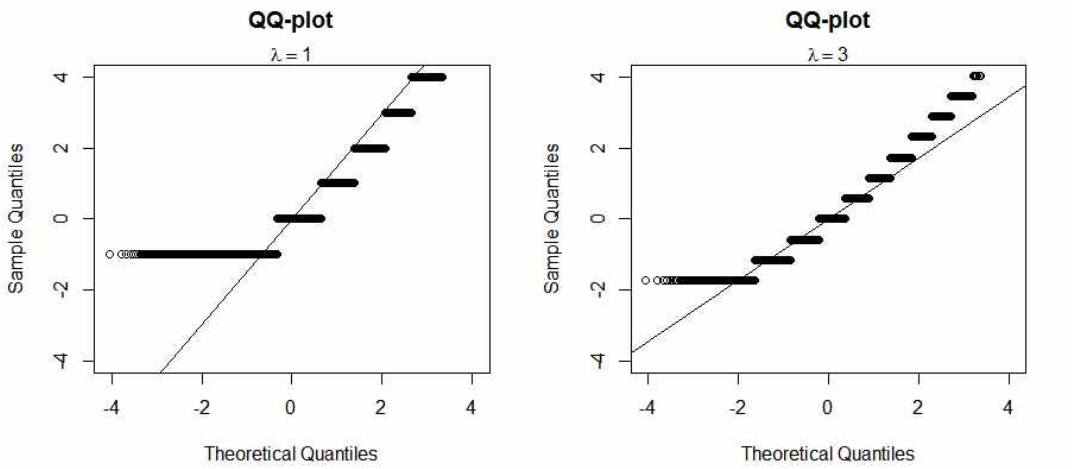
2. X가 모수 λ인 포아송 분포를 따를 때 $Z = \frac{X-\lambda}{\sqrt{\lambda}}$는 λ가 커지면 근사적으로 표준정규분포를 따른다.

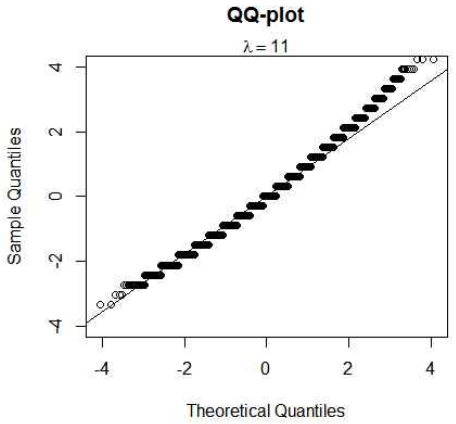
```
for (m in seq(1, 120, 2))
{
    z = (rpois(20000, lambda=m) - m)/sqrt(m)
    qqnorm(z, ylim=c(-4,4), main="QQ-plot")
    qqline(z)
    mtext(bquote(lambda == .(m)), 3)
}
```

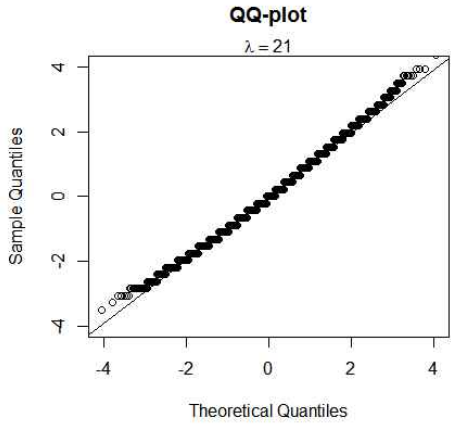
(a) 이 코드를 실행하여 λ값이 커짐에 따라 Z의 분포가 어떻게 변하는지 설명하라.

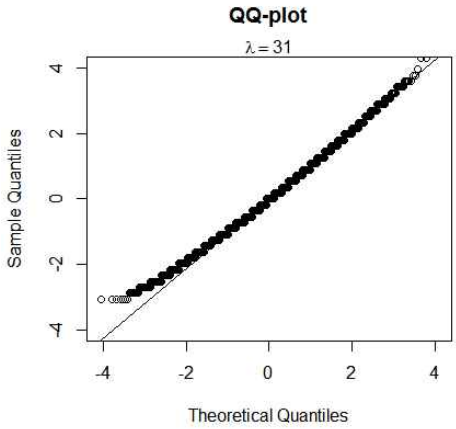
(b) λ가 어느 정도 커지면 QQ-plot에서 거의 직선이 되는가?

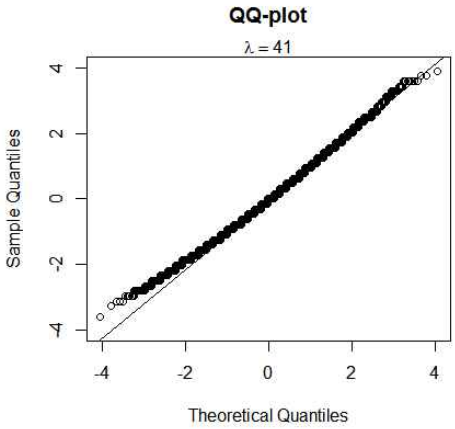
- 위의 코드를 1에서 120까지 실행한 결과, λ가 커질수록 Z는 근사적으로 표준정규분포를 따르는 것을 볼 수 있습니다.



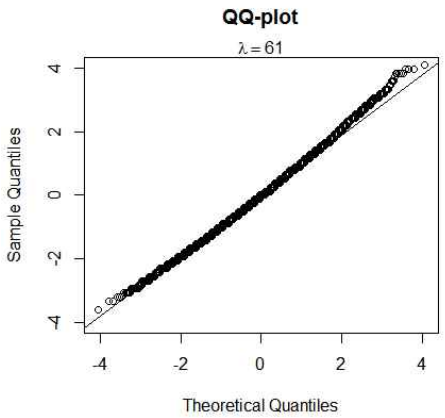


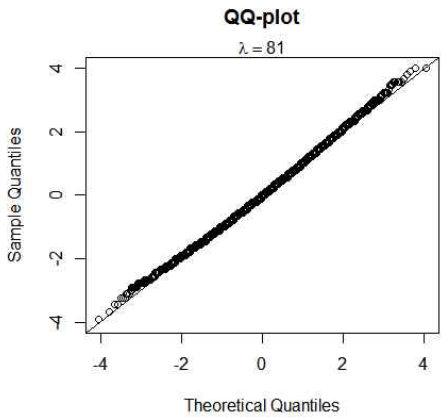


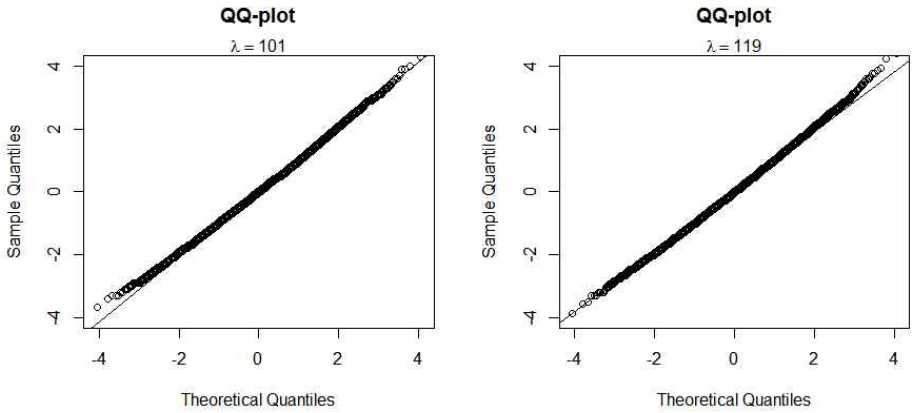




- 위의 그림을 통해서, λ가 41정도만 되어도 QQ-Line가 거의 직선에 가까워짐을 확인할 수 있습니다. λ가 41 이상일 때는 더 직선에 가까운 모습을 볼 수 있습니다.



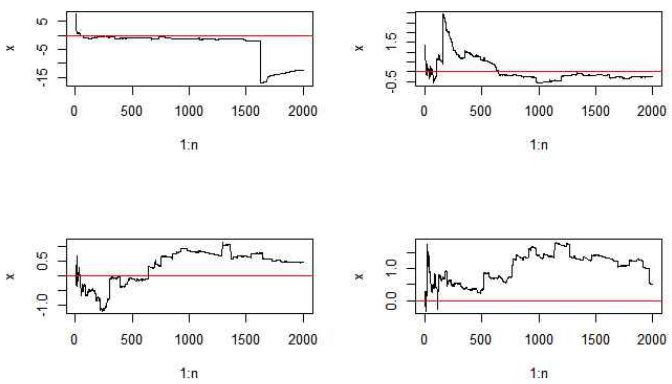




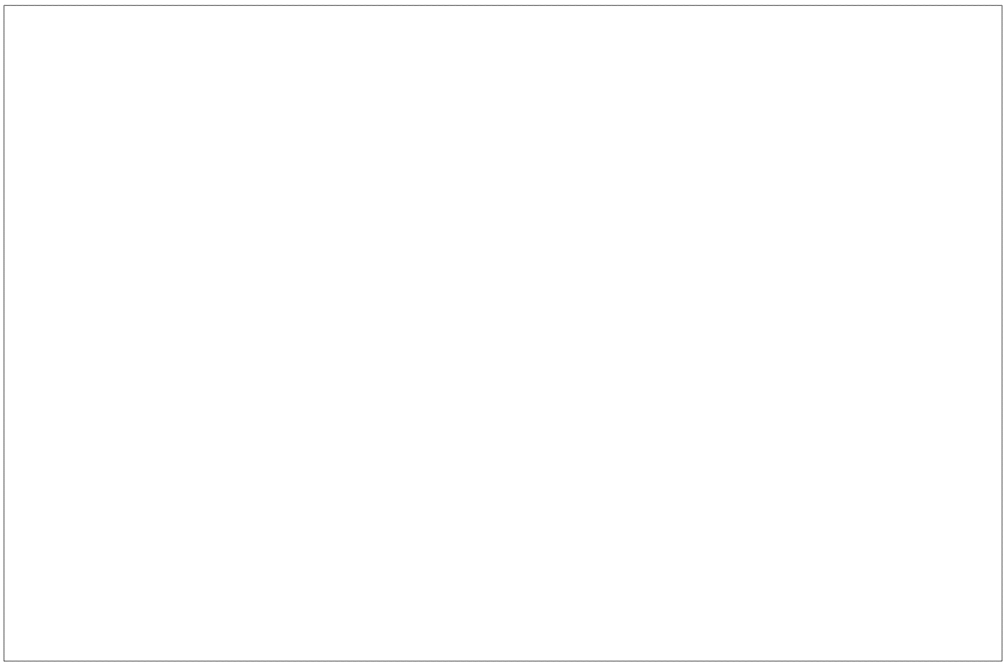
3. 코쉬분포에 대하여 대수의 법칙이 성립하는지 표본크기가 2000 이상인 경우에 대하여 모의실험을 해보고 [그림 8-3]과 비교해 보아라. 또한 중심극한정리가 성립하는지도 모의실험을 통해 확인하여라.

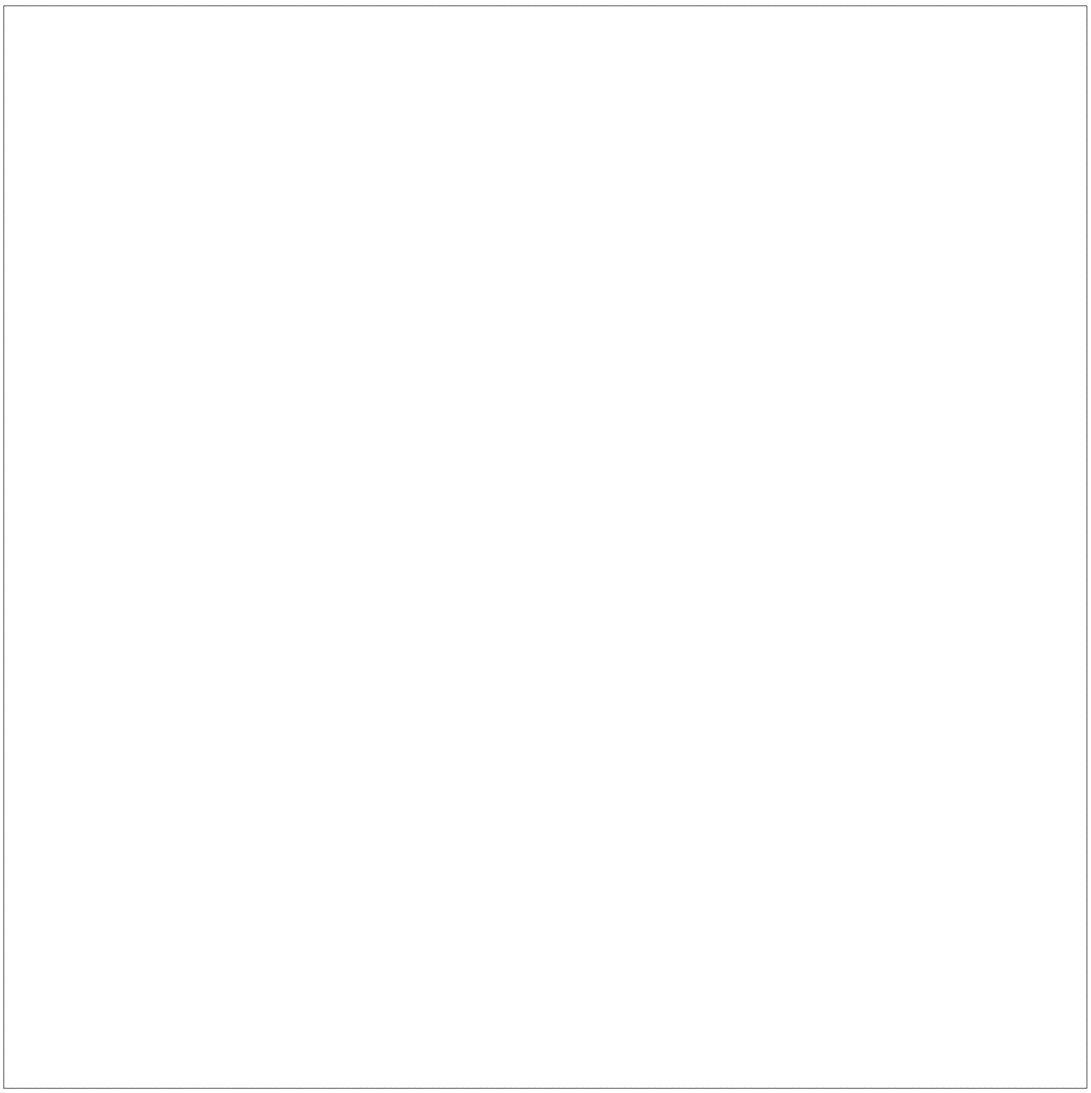
- 코쉬분포는 확률밀도함수가 $f(x) = \frac{1}{\pi\beta 1 + (x-\alpha)^2/\beta^2}, -\infty < x < \infty$와 같이 주어지는 분포로, 평균과 분산이 존재하지 않는 것으로 알려져 있습니다(증명 생략). 즉, 코쉬분포에서는 대수에 법칙에 따른 표본평균이 수렴하여야 할 값이 존재하지 않으며, 코쉬분포에서는 대수의 법칙이 성립하지 않습니다. (책 참고). 아래는 누적합을 계산하는 cumsum 함수를 이용한 강의 예제 코드입니다. 4개의 그림은 서로 다른 극한값을 가지는 것으로 추정됩니다. 이는 표본평균이 하나로 수렴하지 않는다는 것을 의미합니다.

```r
#Q3. RCauchy
n = 2000
par(mfrow = c(2,2))
for (i in 1:4)
{ x = rcauchy(n)
x = cumsum(x)/1:n
plot(1:n, x, type = "l")
abline(h = 0, col = "red")
}
```



- 코쉬분포에서는 대수의 법칙이 성립하지 않으므로 표본 크기가 커져도 평균인 μ는 수렴하지 않습니다.
아래 함수는 강의 예제 코드의 난수 생성을 코쉬분포로 바꾼 것입니다. 이를 통해서, 중심극한정리가 수렴하지 않을을 알 수 있습니다.





5. 아래에서 주어진 데이터에서 표본평균에 대한 붓스트랩과 잭나이프에 의한 95% 신뢰구간을 구하시오.

$$57, 60, 52, 49, 56, 46, 51, 63, 49, 57$$

- bootstrap 패키지를 이용해서 붓스트랩에 의한 신뢰구간을 구했습니다. 붓스트랩 함수의 결과 중에서 thetastar 가 붓스트랩 표본들로부터 구한 변동계수들의 추정치가 됩니다. 따라서, sdcvrats가 변동계수의 표준편차에 대한 붓스트랩 추정치이고 95%의 신뢰구간은 lowlimit를 하한값으로 그리고 upperlimit를 상한값으로 가집니다. (책 참고)

```
> #Q5. Bootstrap & Jackknife
> library(bootstrap)
> data = c(57, 60, 52, 49, 56, 46, 51, 63, 49, 57)
> theta1 = function(x) {mean(x)}
> results.bootstrap = bootstrap(data, 1000, theta = theta1)
> lowlimit = quantile(results.bootstrap$thetastar, 0.05)
> upperlimit = quantile(results.bootstrap$thetastar, 0.95)
> lowlimit
  5%
51.3
> upperlimit
 95%
56.8
```

- bootstrap 패키지의 jackknife 함수를 이용했습니다. 결과 리스트 중 jack.se 가 변동계수의 표준편차에 대한 잭나이프 추정치입니다. 잭나이프 방법에는 앞의 붓스트랩 방법에서와 같이 비모수적으로 신뢰구간을 구하는 방법은 없고 구해진 평균과 표준편차를 이용하여 대략적으로 표준편차의 2배를 전체 자료에 대한 변동계수에서 빼고 더한 구간을 이용할 수 있습니다. (책 참고) 아래의 lowlimit가 하한값, upperlimit가 상한구간입니다.

```
> data = c(57, 60, 52, 49, 56, 46, 51, 63, 49, 57)
> theta1 = function(x) {mean(x)}
> results.jack = jackknife(data, theta1)
> results.jack
$jack.se
[1] 1.719173

$jack.bias
[1] 0

$jack.values
 [1] 53.66667 53.33333 54.22222 54.55556 53.77778 54.88889 54.33333 53.00000 54.55556 53.66667

$call
jackknife(x = data, theta = theta1)

> results.jack$jack.se
[1] 1.719173
> lowlimit = mean(results.jack$jack.values) - 2 * results.jack$jack.se
> upperlimit = mean(results.jack$jack.values) + 2 * results.jack$jack.se
> lowlimit
[1] 50.56165
> upperlimit
[1] 57.43835
```

6. 파라미터가 1과 λ인 두 지수분포의 혼합분포

$$f(x) = \alpha e^{-x} + (1-\alpha)\lambda e^{-\lambda x}$$

에 α와 λ의 최대 가능도 추정량을 구하는 EM 알고리즘을 유도하고 R 함수로 작성하시오. $\alpha = 0.1, \lambda = 5$인 경우 혼합분포 $f(x)$에서 $N = 100$개의 난수를 발생하고 EM 알고리즘에 의해 모수 α와 λ를 추정하여 참값과 비교하시오.

지수분포의 확률밀도함수는 $\lambda e^{-\lambda x}$ 입니다.

따라서, $Likelihood$는 $L(\theta) = \prod_{i=1}^{n} f_{\lambda}(x_i) = \prod_{i=1}^{n} \lambda e^{-\lambda x}$

$Log-Likelihood$는 $L^{*}(\theta) = \sum_{i=1}^{n} \log(\lambda e^{-\lambda x_i}) = \sum_{i=1}^{n} (\log\lambda - \lambda x)$

$L^{*}(\theta)$를 λ에 대하여 미분하면 $\frac{\partial L^{*}(\theta)}{\partial \lambda} = \frac{1}{\lambda} - x = \frac{1-\lambda x}{\lambda} = 0$ 이 되는 $\lambda = \frac{1}{x}$ 입니다.

- 따라서, 이를 바탕으로 강의 예제 코드를 조금 수정하였습니다(dnorm을 dexp로, lambda 업데이트)

```
> #지수분포의 log-likelihood 함수값을 구하는 함수
> Log.lik = function(x, R=2, lambda, prior)
+ {
+   lik = 0
+   for (r in 1:R)
+     lik = lik + prior[r] * dexp(x, rate = lambda[r])
+   return(sum(log(lik)))
+ }
```

```
> #EM 알고리즘
> Exp.Mixture = function(X, R=2, maxiter=1000, eps=1e-5)
+ {
+
+   X = as.vector(X)
+   N = length(X)
+
+   lambda = prior = rep(0, R)
+
+   gama = matrix(0, R, N)
+   # 확률 초기화
+   prior = rep(1/R, R)
+   lambda = c(1,3)
+
+   old.lik = Log.lik(X, R, lambda, prior)
+   track.lik = as.vector(NULL)
+   track.lik = c(old.lik)
+   for (i in 1:maxiter)
+   {
+     for (r in 1:R)
+       gama[r, ] = prior[r] * dexp(X, rate = lambda[r])
+     denom = apply(gama, 2, sum)
+     for (r in 1:R)
+     {
+       gama[r, ] = gama[r, ] / denom
+       lambda[r] = t(gama[r, ]) %*% sqrt(1/X) /  sum(gama[r, ])
+     }
+     prior = apply(gama, 1, sum) / N
+     lambda[1] = 1
+     new.lik = Log.lik(X, R, lambda, prior)
+
+     if (abs(old.lik - new.lik) < eps * abs(old.lik))  break
+     old.lik = new.lik
+     track.lik = c(track.lik, old.lik)
+   }
+   return(list(lambda = lambda, prior = prior,
+               track = track.lik, resp = gama[r, ]))
+ }
```

```
> #혼합분포에서 난수 생성
> random.number = c()
> while(length(random.number)< 100){
+     v1 = runif(1)
+     if(v1 < 0.1){
+       rn = rexp(1, rate = 1)
+       random.number = c(random.number, rn)
+     }else{
+       rn = rexp(1, rate = 5)
+       random.number = c(random.number, rn)
+     }
+ }
> Exp.Mixture(random.number)
$lambda
[1] 1.000000 4.743209

$prior
[1] 0.1008393 0.8991607

$track
 [1]  9.614893 28.918219 33.270982 34.731548 35.302372 35.517549 35.588899 35.606074 35.605252
[10] 35.599966 35.594716 35.590643 35.587776 35.585852 35.584595 35.583787 35.583271

$resp
  [1] 0.9657206182 0.9646506597 0.9563366389 0.8316681707 0.8577719767 0.9726126758 0.9500274208
  [8] 0.9768493907 0.9421724732 0.9488526413 0.9739207115 0.7792188003 0.9329562203 0.9582534395
 [15] 0.9506532343 0.8170571126 0.9489723032 0.9356971398 0.9743199218 0.8859989071 0.9705982280
 [22] 0.9738565104 0.9723212636 0.9628557867 0.3630864478 0.9753563500 0.9736841065 0.7461690484
 [29] 0.9604375647 0.9768138842 0.7873160971 0.9488153448 0.9617952731 0.9561498831 0.9724115426
 [36] 0.8648953697 0.8560991766 0.9599559891 0.9744208205 0.9624037353 0.9692967907 0.9544947222
 [43] 0.9550125049 0.9461823041 0.0003865067 0.8150705231 0.9701955165 0.9654303328 0.9527805738
 [50] 0.9567686118 0.9762430951 0.9660177401 0.9684394466 0.9607198079 0.9748995540 0.9733795120
 [57] 0.9756584787 0.9671500634 0.8355748706 0.9611455565 0.9040043112 0.9760934072 0.9640654843
 [64] 0.9268709585 0.7192891460 0.9228838345 0.9767602684 0.9753391937 0.9653728127 0.9250372985
 [71] 0.9722336518 0.9641580254 0.1060645927 0.9726795391 0.9728617481 0.9291629704 0.9682020371
 [78] 0.9640709572 0.1099341192 0.9766822427 0.9760250744 0.8412743050 0.9598837811 0.9701192690
 [85] 0.9690291614 0.9488852401 0.9633202046 0.9709488321 0.9754561012 0.9713091537 0.9209566906
 [92] 0.9556339482 0.9725554778 0.9756457461 0.0004978920 0.8538404322 0.9671654617 0.9753575576
 [99] 0.9200288512 0.9483742786
```

- lambda 는 1과 4.7, 그리고 가중치는 0.1와 0.899로 아주 유사하게 나온 것을 알 수 있습니다.