통계계산 과제

Seongmin Ji (student id: 2021710322)
2021-05-22

Bradley Eron's paper

1. Show $(4) = (2)$

다음과 같이 정의하자.

$$\bar{x}_{(i)} = \frac{n\bar{x} - x_i}{n-1} = \frac{1}{n-1}\sum_{j\neq i} x_j$$

$$\bar{x}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n} \bar{x}_{(i)} = \sum_{i=1}^{n} \frac{\bar{x} - x_i/n}{n-1} = \frac{(n-1)\bar{x}}{n-1} = \bar{x}$$

$$\hat{\sigma_J^2} = \frac{n-1}{n}\sum_{i=1}^{n} (\bar{x}_{(i)} - \bar{x}_{(\cdot)})^2$$

$$= \frac{n-1}{n}\sum_{i=1}^{n} \Big(\frac{1}{n-1}\sum_{j\neq i}^{n} x_j - \bar{x}\Big)^2$$

$$= \frac{1}{n(n-1)}\sum_{i=1}^{n}\Big(\sum_{j\neq i}^{n} x_j - \frac{n-1}{n}\sum_{j=1}^{n} x_j\Big)^2$$

$$= \frac{1}{n(n-1)}\sum_{i=1}^{n}\Big(-x_i + \frac{1}{n}\sum_{j=1}^{n} x_j\Big)^2 = \frac{1}{n(n-1)}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\therefore\ (4) = \hat{\sigma}_J = \sqrt{\frac{1}{n(n-1)}\sum_{i=1}^{n} (x_i - \bar{x})^2} = \hat{\sigma}_n = (2)$$

2. Show $(9) = (8) \times \frac{n^2}{(n^2-1)}$

다음으로부터

$$\bar{x}_{(i)} = \frac{n\bar{x} - x_i}{n-1} = \frac{1}{n-1}\sum_{j\neq i} x_j$$

아래식이 성립한다.

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}_{(i)})^2 &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \frac{n\bar{x} - x_i}{n-1})^2 \\ &= \frac{n}{(n-1)^2}\sum_{i=1}^{n}(x_i - \bar{x})^2\end{aligned}$$

한편 다음이 성립하므로

$$(n+1)\hat{\sigma}_n^2 = \frac{n+1}{n(n-1)}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

$$\therefore\ (9) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}_{(i)})^2 = \frac{n}{(n-1)^2}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{n^2}{n^2-1}\frac{n+1}{n(n-1)}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

$$= \frac{n^2}{n^2-1}(n+1)\hat{\sigma}_n^2 = \frac{n^2}{n^2-1}(8)$$

Chapter 7

7.A

In [1]:

```r
library(boot)

B <- 3000
N <- 200
mu <- 0
sigma <- 1

s_set <- rnorm(N, mu, sigma)

theta.boot <- function(s_set, booted){
  mean(s_set[booted])
}

boot_obj <- boot(s_set, statistic = theta.boot, R = B)
c_i <- boot.ci(boot_obj, type = c("basic", "norm", "perc"))

### matrix of confidence interval
c_i_mat <- matrix(c(c_i$normal[2:3], c_i$basic[4:5], c_i$percent[4:5]), 3, 2, byrow = T)
miss_mat <- matrix(0, 3, 2, byrow=T)

colnames(miss_mat) <- c("left", "right")
rownames(miss_mat) <- c("basic", "norm", "perc")
```

In [2]:

```
### find the ratio of missing value
for(i in 1:3){

  samp_mean <- vector("numeric", 1000)
  for(irit in 1: 1000){
    sampled <- sample(1:N, N, replace = T)
    samp_mean[irit] <- theta.boot(s_set, sampled)
  }
  miss_mat[i, 1] <- sum(samp_mean < c_i_mat[i, 1])
  miss_mat[i, 2] <- sum(samp_mean > c_i_mat[i, 2])
}

print(c_i)

print(miss_mat/1000)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 3000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_obj, type = c("basic", "norm", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   (-0.1247,  0.1525 )   (-0.1262,  0.1534 )   (-0.1245,  0.1550 )
Calculations and Intervals on Original Scale
       left right
basic 0.026 0.017
norm  0.025 0.026
perc  0.027 0.019
```

In [3]:

```
### Test for the independent sampling
miss_mat <- matrix(0, 3, 2, byrow=T)

colnames(miss_mat) <- c("left", "right")
rownames(miss_mat) <- c("basic", "norm", "perc")


samp_mean <- vector("numeric", 1000)
for(irit in 1: 1000){
  sampled <- sample(1:N, N, replace = T)
  samp_mean[irit] <- theta.boot(rnorm(N, mu, sigma), sampled)
}

for(i in 1:3){
  miss_mat[i, 1] <- sum(samp_mean < c_i_mat[i, 1])
  miss_mat[i, 2] <- sum(samp_mean > c_i_mat[i, 2])
}

print(c_i)

print(miss_mat/1000)
```

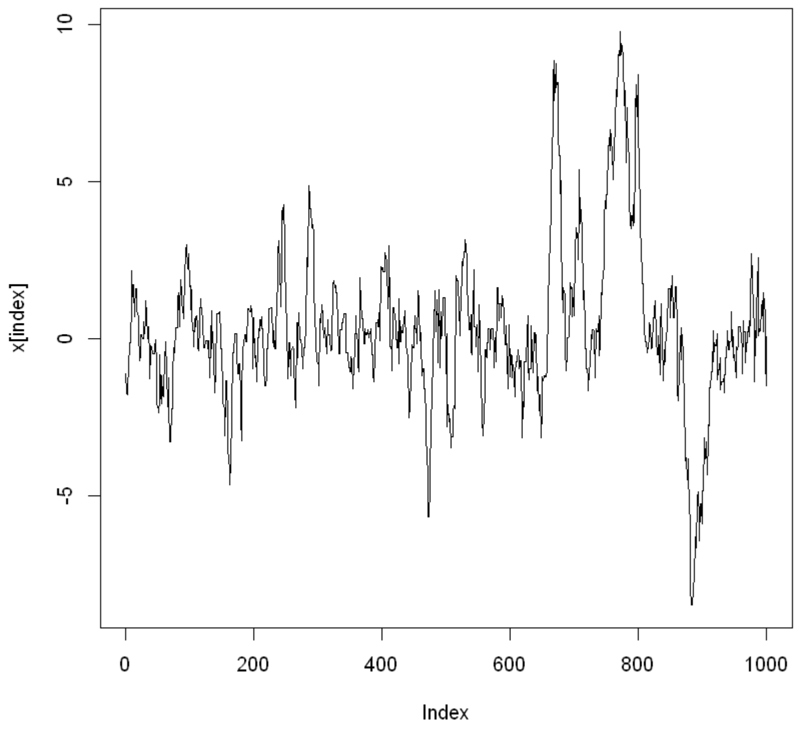
```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 3000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_obj, type = c("basic", "norm", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   (-0.1247,  0.1525 )   (-0.1262,  0.1534 )   (-0.1245,  0.1550 )
Calculations and Intervals on Original Scale
       left right
basic 0.114 0.059
norm  0.110 0.059
perc  0.115 0.053
```

Chapter 9

9.3

In [4]:

```
### Metropolis-Hostings sampler with target dist: Cauchy(0,1)

n <- 10000
sigma <- 1
x <- vector("numeric", n)


# set a proposal distribution as Normal(0, sigma)
x[1] <- rnorm(1, 0, sigma)
k <- 0


for (i in 2:n){
  xt <- x[i-1]
  y <- rnorm(1, xt, sigma)
  u <- runif(1, 0, 1)


  nu <- dnorm(xt, y, sigma)*dcauchy(y, 0, 1)
  den <- dnorm(y, xt, sigma)*dcauchy(xt, 0, 1)
  r <- nu/den

    # acceptance-rejection method to make it a reversible MC
  if(u <= min(r, 1)) x[i] <- y
  else{
    x[i] <- xt
    k <- k + 1 # y is rejected
  }
}

index <- 9001:10000

plot(x[index], type = "l")
```



In [5]:

```
# the proportion of rejecting y in n
print(k/n)
```

```
[1] 0.224
```

In [6]:

```
# compare with theoretical quantile
q1 <- seq(0.1 , 0.5, 0.1)
q2 <- seq(0.6 , 0.9, 0.1)

compare1 <- rbind(quantile(x[index], probs = q1), qcauchy(q1, location = 0, scale = sigma))

rownames(compare1) <- c("quantiles form the sampler", "Theoretical value")
compare1

compare2 <- rbind(quantile(x[index], probs = q2), qcauchy(q2, location = 0, scale = sigma))
rownames(compare2) <- c("quantiles form the sampler", "Theoretical value")
compare2
```

A matrix: 2 × 5 of type dbl

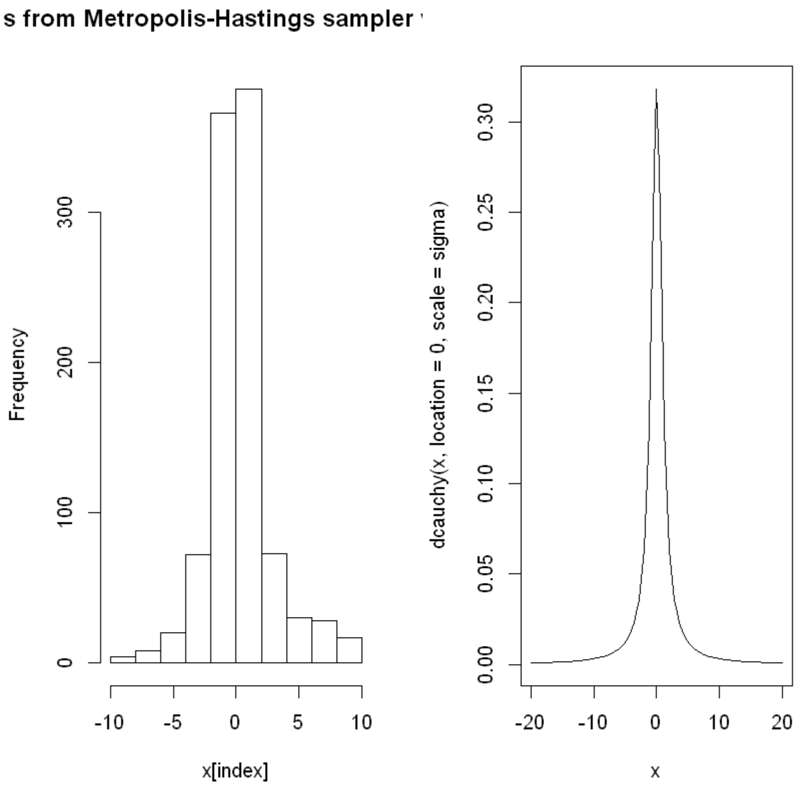
	10%	20%	30%	40%	50%
quantiles form the sampler	-2.095014	-1.052469	-0.5367557	-0.2315536	0.1121612
Theoretical value	-3.077684	-1.376382	-0.7265425	-0.3249197	0.0000000

A matrix: 2 × 4 of type dbl

	60%	70%	80%	90%
quantiles form the sampler	0.3730953	0.8234607	1.505810	3.040734
Theoretical value	0.3249197	0.7265425	1.376382	3.077684

In [7]:

```
# Draw a histogram of 9001th~10000th samples with the theoretical curve
par(mfrow = c(1,2))

hist(x[index], main = "1000 samples from Metropolis-Hastings sampler with Normal dist")
curve(dcauchy(x, location = 0, scale = sigma), from = -20, to = 20)
```



9.8

Gibbs sampler 를 이용하여 다음 결합확률분포의 표본을 구하자.

$$f(x,y),\ x \in \mathbb{N} \cup \{0\},\ y \in [0,1]$$

이때 조건부확률분포는 다음과 같이 주어진다.

$$f(x|y) \sim B(n,y),\ \ f(y|x) \sim Beta(x+a, n-x+b)$$

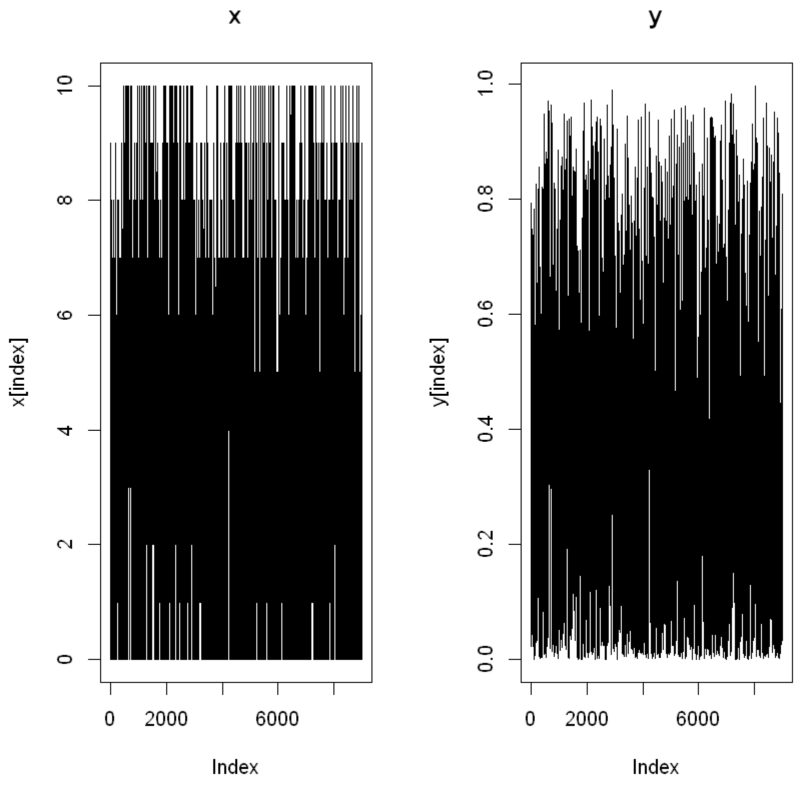
Let $a = 1,\ b = 2$

In [8]:

```
### Gibbs sampler

m = 10000
x <- vector("numeric", m)
y <- vector("numeric", m)
x[1] <- NA
y[1] <- 0.5

n <- 10
a <- 1
b <- 2
for(i in 2:m){
  yt <- y[i-1]
  x[i] <- rbinom(1, n, p = yt)
  y[i] <- rbeta(1, shape1 = x[i]+a, shape2  = n-x[i]+b)
}
```

In [9]:

```
# Get samples with ruling out the 1:1000 sample
index <- 1001:10000

par(mfrow = c(1,2))
plot(x[index], type = "l", main = "x")
plot(y[index], type = "l", main = "y")
```



Bayes' rule 에 의해

$$f(y|x) = \frac{f(x|y) * f(y)}{\int_{[0,1]} f(x,y)dy}$$

따라서 $f(x|y) \sim B(n, y),\ f(y) \sim Beta(a, b)$ 일 때

$f(y|x) \sim Beta(x + a, n - x + b)$ 이므로

$$f(y) \sim Beta(a, b)$$

In [10]:

```
# Compare the theoretical quantile of y
q1 <- seq(0.1 , 0.5, 0.1)
q2 <- seq(0.6 , 0.9, 0.1)

compare1 <- rbind(quantile(y[index], probs = q1), qbeta(q1, shape1= a, shape2 = b))

rownames(compare1) <- c("quantiles form the sampler", "Theoretical value")
compare1

compare2 <- rbind(quantile(y[index], probs = q2), qbeta(q2, shape1= a, shape2 = b))
rownames(compare2) <- c("quantiles form the sampler", "Theoretical value")
compare2
```

A matrix: 2 × 5 of type dbl

	10%	20%	30%	40%	50%
quantiles form the sampler	0.05908062	0.1171050	0.1787504	0.2444443	0.3117735
Theoretical value	0.05131670	0.1055728	0.1633400	0.2254033	0.2928932

A matrix: 2 × 4 of type dbl

	60%	70%	80%	90%
quantiles form the sampler	0.3854428	0.4673391	0.5730118	0.6945162
Theoretical value	0.3675445	0.4522774	0.5527864	0.6837722

Chapter 11

11.5

a 에 대한 근을 찾아라.

$$\frac{2\Gamma(\frac{k}{2})}{\sqrt{\pi(k-1)}\Gamma(\frac{(k-1)}{2})} \int_0^{c_{k-1}} \left(1 + \frac{u^2}{k-1}\right)^{-k/2} = \frac{2\Gamma(\frac{k+1}{2})}{\sqrt{\pi k}\Gamma(\frac{k}{2})} \int_0^{c_k} \left(1 + \frac{u^2}{k}\right)^{-(k+1)/2}$$

Where

$$c_k = \sqrt{\frac{a^2 k}{k + 1 - a^2}}$$

이 때 a 는 $(0, \sqrt{k})$ 사이에 있는 다음의 교점이다.

$$P\Big(t(k-1) > \sqrt{\frac{a^2(k-1)}{k-a^2}}\Big)$$
$$P\Big(t(k) > \sqrt{\frac{a^2k}{k+1-a^2}}\Big)$$

In [11]:

```
f11.5 <- function(k){
  fint <- function(u,n){
    (1+ u^2/(n-1))^{-n/2}
  }
  ck <- function(n, a){
    sqrt(a^2*n/(n+1 - a^2))
  }

  ## left or right term
  expre <- function(n, a){
    g <- function(u){
      fint(u, n)
    }
    c <- ck(n-1, a)

    2*gamma(n/2)/(sqrt(pi*(n-1))*gamma((n-1)/2)) * integrate(g, lower = 0, upper = c)$value
  }

  eq <- function(a){
    left <- expre(k, a)
    right <- expre(k+1, a)
    return(left - right)
  }

  eps <- 0.01
  testvalue <- eq(eps) * eq(sqrt(k)-eps)
  if(is.nan(testvalue)){
    solution <- NA
  }else if(testvalue < 0){
    solution <- uniroot(eq, interval = c(eps, sqrt(k)-eps))$root
  }else{
    solution <- NA
  }
  return(solution)
}
```

In [12]:

```
valuek <- c(4:25, 100, 500, 1000)
result <- sapply(valuek, function(k) f11.5(k))
result
```

```
1.49207328373192 · 1.53355476613729 · 1.56274494704044 · 1.58442931421668 ·
1.60118445644418 · 1.61451636648543 · 1.62538941289325 · 1.63441842875999 ·
1.64202600904932 · 1.64855232674654 · 1.65417450443119 · 1.65909949192816 ·
1.66345352992923 · 1.66728747635259 · <NA> · 1.67383235080696 ·
1.67660540992662 · 1.67914788527931 · 1.68146833623951 · 1.68359336240274 ·
<NA> · 1.68736566329355 · <NA> · <NA> · <NA>
```

따라서 $k \to \infty$ 에 따라 대략 1.70 에 수렴하는 것으로 보인다.

11.6

Cauchy(η, θ), $\theta > 0$, $\eta \in \mathbb{R}$ 분포의 확률밀도함수가 다음과 같다.

$$\frac{1}{\theta\pi\left(1 + \{(x-\eta)/\theta\}^2\right)}, \quad -\infty < x < \infty$$

수치적인 방법으로 cdf 를 계산하고 함수 pcauchy 의 결과와 비교하자.

In [13]:

```
# pdf of cauchy distribution
cauchy <- function(x, nu=0, theta=1){
  1/(theta*pi*(1 + ((x - nu)/theta)^2) )
}

# values from pcauchy
values <- seq(-10, 10, length = 10)
theorical_values <- pcauchy(values, 0, 1)
```

In [14]:

```
# calculate values from 'integrate' with 'cauchy'
numerical_values <- numeric(length(values))

for(i in 1:length(values)){
    numerical_values[i] <- integrate(cauchy, -Inf, values[i])$value
}
```

In [15]:

```
round(rbind(numerical_values, theorical_values), 3)
```

A matrix: 2 × 10 of type dbl

numerical_values	0.032	0.041	0.057	0.093	0.233	0.767	0.907	0.943	0.959	0.968
theorical_values	0.032	0.041	0.057	0.093	0.233	0.767	0.907	0.943	0.959	0.968

In [16]:

```
numerical_values - theorical_values
```

```
1.51314724639029e-11 ·  6.06688310700321e-13 ·  1.67059768796385e-11 ·
-1.18655085756814e-14 ·  -2.77555756156289e-17 ·  3.93018950717305e-14 ·
-5.9094507065538e-11 ·  2.9561908476694e-12 ·  -9.20041820506867e-13 ·
1.46177514537271e-11
```

이번에는 다양한 수치적인 방법으로 함수를 만들고 계산해보자.

In [17]:

```
# Other numerical methods

## Trapizoid method
Trap <- function(f, low, up, n = 1000){
  h <- abs(up-low) / n
  integral <- (f(low) + f(up)) / 2
  x <- low
  for(i in 1:(n-1)){
    x <- x + h
    integral <- integral + f(x)
  }
  integral <- integral * h
  return(integral)
}
## Rectangle method

Rectan <- function(f, low, up, n = 1000){
  sum <- 0
  h <- abs(up-low)/n
  for (i in 1:n) sum = sum + h*f(low + i*h)
  return(sum)
}

## Simpson's fomula
Sim <- function(f, low, up, n=1000){
  h <- abs(up-low) / n
  x <- low
  integral <- 0.5*f(up)
  for(i in 1:n){
    integral <- integral + (0.5*f(x) + 2*f((2*x+h)/2) + 0.5*f(x+h))
    x <- x + h
  }
  integral <- integral * h / 3
  return(integral)
}
```

위의 방법들은 $x \to -\infty$ 일 때 값이 계산되지 않는다.

따라서 $P(-100 < X \leq 10) = cdf(10) - cdf(-100)$ 을 계산해보자.

In [18]:

```
# Integrate the pdf with numerical methods

nu = 0; theta = 1

rfunction <- pcauchy(10, location = nu, scale = theta) - pcauchy(-100, location = nu, scale = theta)

trapizoid <- Trap(f = cauchy, -100, 10, n=1000)
rectangle <- Rectan(f = cauchy, -100, 10, n=1000)
simpson <- Sim(f = cauchy, -100, 10, n=1000)

simpson2 <- (2 * rectangle + trapizoid)/3

result <- c(rfunction, trapizoid, rectangle, simpson, simpson2)

names(result) <- c("pcauchy", "trapizoid", "rectangle", "Simpson", "Simpson2")

print(result)
```

```
  pcauchy trapizoid rectangle   Simpson  Simpson2
0.9650915 0.9650909 0.9652624 0.9651493 0.9652053
```