2022년 2학기

Computational Statistics

HW#6

222STG10

김희숙

Problem

1. **Riemann, trapezoidal, simpson 3가지 적분 방법을 R 함수로 구현하라.**

   Code appendix에 첨부하였습니다.

2. **교재 5.3, 5.4 문제를 풀이하라.**

5.3. Suppose the data (x1 , . . . , x7) = (6.52, 8.32, 0.31, 2.82, 9.96, 0.14, 9.64) are observed. Consider Bayesian estimation of μ based on a N (μ, 3^2 /7) likelihood for the minimally sufficient x¯| μ, and a Cauchy(5,2) prior.

a)

used	result	subint	time	niter
integrate	0.1274447		0.001647949 secs	
myIntegration - Riemann	0.1274447	134217728	6.776171 secs	18
myIntegration - Trapezoidal	0.1274447	134217728	5.468461 secs	18
myIntegration - Simpson	0.1274447	268435456	17.81934 secs	19

μ 를 표본평균으로 추정 후 이를 dnorm()의 mean값으로 이용한다. Result값은 0.1274447이며 이는 1/k를 의미한다. 따라서 k = 7.84654 이다.

b)

used	result	subint	time	niter
integrate	0.9960547		0.001242161 secs	
myIntegration - Riemann	0.9960314	2048	0.0002820492 secs	2
myIntegration - Trapezoidal	0.9960546	2048	0.0002448559 secs	2
myIntegration - Simpson	0.9960547	2048	0.0006558895 secs	2

Result 값을 보면 0.99605과 유사한 값을 도출하는 것을 확인 할 수 있다.

c)

used	result	subint	time	niter
integrate	0.9908595		0.02953291 secs	
myIntegration - Riemann	0.9908603	32768	0.00473094 secs	6
myIntegration - Trapezoidal	0.9908595	4096	0.0007679462 secs	3
myIntegration - Simpson	0.9908595	4096	0.001387119 secs	3

문제에서 주어진 transformation을 하면 범위가 1/(1+exp(-3))에서 1까지로 바뀌게 된다. 이때 함수 분모에 (1-u)가 존재하게 되며, 1에서의 값을 구할 수 없어진다. 첫번째 방법은 1을 무시하여, 1/(1+exp(-3))에서 0.999999까지 적분을 하며, 그 결과는 위 표와 같다. 0.99086과 유사한 값을 출력한다.

used	result	subint	time	niter
integrate	0.9908595		0.001649141 secs	
myIntegration - Riemann	0.9908603	32768	0.03341103 secs	6
myIntegration - Trapezoidal	0.9908595	4096	0.0009338856 secs	3
myIntegration - Simpson	0.9908595	4096	0.00107789 secs	3

두번째 방법은 1에서의 값을 0.99999999999으로 고정하여 적분을 진행하도록 하였다. 그 결과 첫번째 결과와 result, subint, niter 값은 동일한 것을 확인 할 수 있었다. 하지만 적분 함수를 정의 하는 과정에서 for문을 통해 값을 확인하며 1일 때 0.99999999999을 통한 계산을 하는 코드가 추가되어 time이 더 오래 걸리는 것을 볼 수 있다.

d)

used	result	subint	time	niter
integrate	0.9908595		0.00852704 secs	
myIntegration - Riemann	0.990859	131072	0.01119494 secs	8
myIntegration - Trapezoidal	0.9908595	2048	0.0003600121 secs	2
myIntegration - Simpson	0.9908595	2048	0.0003068447 secs	2

문제에서 주어진 transformation을 하면 범위가 (c)에서와 마찬가지로 분모에 u가 있어 0에서의 대체값이 필요해진다. 우선 이 부분을 무시하고 적분을 0.0001 에서 1/3 까지 진행한 결과는 위 표와 같다. 0.99086과 유사한 값을 보이는 것을 확인 할 수 있다.

used	result	subint	time	niter
integrate	0.9908595		0.02139306 secs	
myIntegration - Riemann	0.990859	131072	0.02803612 secs	8
myIntegration - Trapezoidal	0.9908595	2048	0.000535965 secs	2
myIntegration - Simpson	0.9908595	2048	0.0008840561 secs	2

두번째 방법은 0에서의 값을 0.00000001으로 고정하여 적분을 진행하도록 하였다. 그 결과 (c)와 같이, 첫번째 결과와 result, subint, niter 값은 동일하나 for문을 통해 값을 확인하여 0일 때 0.00000001을 통한 계산을 하는 코드가 추가되어 time이 더 오래 걸리는 것을 볼 수 있다.

5.4. Let X~Unif[1,a] and Y=(a−1)/X, for a>1. Compute E{Y}=loga using Romberg's algorithm for m = 6. Table the resulting triangular array. Comment on your results.

$X \sim Unif[1,a] \rightarrow f_X(x) = \frac{1}{b-a} = \frac{1}{a-1}$ $\quad$ $1 \quad a \quad \frac{1}{a-1}$

$Y = (a-1)/X$ $\qquad f_Y(y) = f(x) \cdot \left|\frac{dx}{dy}\right|$

$X = (a-1)/Y$ $\qquad = \frac{1}{a-1} \cdot (a-1)\frac{1}{y^2} = \frac{1}{y^2}$

$E\{Y\} = \log a$

$E\{Y\} = E((a-1)/X) = (a-1)E\left(\frac{1}{X}\right)$

$E(Y) = \int y \cdot f(y)\,dy = \int y \cdot \frac{1}{y^2} \cdot dy = [\ln y]_a^a$

a = 5, log(a) = log 5 = 1.609438

used	result	subint	time	niter
integrate	1.609438		0.001215219 secs	
myIntegration - Riemann	1.609536	32768	0.004497051 secs	6
myIntegration - Trapezoidal	1.609439	2048	0.0001060963 secs	2
myIntegration - Simpson	1.609438	2048	9.584427e-05 secs	2

a = 10, log(a) = log 10 = 2.302585

used	result	subint	time	niter
integrate	02.302585		0.001567841 secs	
myIntegration - Riemann	2.302647	131072	0.002078056 secs	8
myIntegration - Trapezoidal	2.302591	2048	0.0001049042 secs	2
myIntegration - Simpson	2.302585	2048	0.0001249313 secs	2

E(Y)에 대하여 적분 함수를 이용하여 a=5, a=10일 때를 각각 3가지 적분법을 통해 계산한 결과는 위 표와 같다. E(Y)에 대한 함수는 첨부한 그림과 같이 계산하여 정의하였다. 그 결과 a=5, a=10인 경우에 대해 E(Y) = log a 인 결과를 확인 할 수 있었다.

Code appendix

```
myIntegration = function(f, a, b, epsilon=10^(-6), option){
  n = 2^9 # sub interval 개수

  error = 1
  result = 0
  maxiter = 1000

  start.time = Sys.time()
  niter = 0
  if (option=='r'){ #riemann
    while (error>=epsilon && niter<=maxiter){
      result_1 = result
      i = seq(1, n-1)
      h = (b-a)/n

      result = h*(f(a)+sum(f(a+i*h)))

      error = abs(result_1-result)
      n = n*2
      niter = niter+1
    }
  }

  else if (option=='t'){ #trapezoidal
    while (error>=epsilon && niter<=maxiter){
      result_1 = result
      i = seq(1, n-1)
      h = (b-a)/n

      result = h*f(a)/2 + h*sum(f(a+i*h)) + h*f(b)/2

      error = abs(result_1-result)
      n = n*2
      niter = niter+1
    }
  }

  else if (option=='s'){ # simpson
```

```
    while (error>=epsilon && niter<=maxiter){
      result_1 = result
      i = seq(1, n/2)
      h = (b-a)/n

      result = h/3*sum(f(a+(2*i-2)*h) + 4*f(a+(2*i-1)*h) + f(a+2*i*h)) # xj = a+j*h

      error = abs(result_1-result)
      n = n*2
      niter = niter+1
    }
  }
  end.time = Sys.time()
  lst = list(result=result,
             subint=n,
             time=end.time-start.time,
             niter=niter)
  return(lst)
}

integrate(dnorm, -1.645, 1.645)

myIntegration(dnorm, -1.645, 1.645, option='r')
myIntegration(dnorm, -1.645, 1.645, option='t')
myIntegration(dnorm, -1.645, 1.645, option='s')


integrate(dnorm, -Inf, Inf)
myIntegration(dnorm,  -10^5, 10^5, option='s')




# 5.3
# a
x <- c(6.52, 8.32, 0.31, 2.82, 9.96, 0.14, 9.64)
mu.hat = mean(x) #x.bar

dcauchy(x, location=5, scale=2)
```

```r
prior_lik <- function(x){
  dnorm(x,mean=mu.hat,sd=sqrt(3^2/7))*dcauchy(x, location=5, scale=2)
}

a = Sys.time()
integrate(prior_lik, -Inf, Inf)
b = Sys.time()
b-a

myIntegration(prior_lik, -10^7, 10^7, option='r')
myIntegration(prior_lik, -10^7, 10^7, option='t')
myIntegration(prior_lik, -10^7, 10^7, option='s')

# 0.1274447 = 1/k
k = 1/0.1274447
k # 7.84654

# b
k_prior_lik<-function(x){
  k*dnorm(x,mean=mu.hat,sd=sqrt(3^2/7))*dcauchy(x, location=5, scale=2)
}



a = Sys.time()
integrate(k_prior_lik, 2, 8)
b = Sys.time()
b-a

myIntegration(k_prior_lik, 2, 8, epsilon = 10^(-4), option='r')
myIntegration(k_prior_lik, 2, 8, epsilon = 10^(-4), option='t')
myIntegration(k_prior_lik, 2, 8, epsilon = 10^(-4), option='s')

## another way
rie = myIntegration(prior_lik, 2, 8, epsilon = 10^(-4), option='r')
tra = myIntegration(prior_lik, 2, 8, epsilon = 10^(-4), option='t')
sim = myIntegration(prior_lik, 2, 8, epsilon = 10^(-4), option='s')
c(rie$result*k, rie$time) # 느린 방법
c(tra$result*k, tra$time)
c(sim$result*k, sim$time)
```

```
# c
myu <- function(u){
  k*dnorm(-log((1-u)/u), mean=mu.hat, sd=sqrt(3^2/7))*dcauchy(-log((1-u)/u), location=5, scale=2)/(u*(1-u))
}

# first way
# transformation -> 범위가 1/(1+exp(-3))에서 1까지로 바뀜
# 위 함수에서 분모에 (1-u)이 존재 -> 1에서의 값을 구할 수 없음
# 첫 번째 방법은 1을 무시, 그러므로 1/(1+exp(-3))에서 0.999999까지 적분.

a = Sys.time()
integrate(myu, 1/(1+exp(-3)), 0.999999)
b = Sys.time()
b-a
myIntegration(myu, 1/(1+exp(-3)), 0.999999, option='r')
myIntegration(myu, 1/(1+exp(-3)), 0.999999, option='t')
myIntegration(myu, 1/(1+exp(-3)), 0.999999, option='s')
# 0.99086과 가까운 값을 출력


# second way
# 1에서의 값을 고정 -> integration 진행
# 1에서의 값: 0.99999999999으로 fix
myu2 = function(u){
  result = k*dnorm(-log((1-u)/u), mean=mu.hat, sd=sqrt(3^2/7))*dcauchy(-log((1-u)/u), location=5, scale=2)/(u*(1-u))
  for (i in seq(1:length(u))){
    if (is.na(result[i])){
      a = 0.99999999999
      result[i] = k*dnorm(-log((1-a)/a), mean=mu.hat, sd=sqrt(3^2/7))*dcauchy(-log((1-a)/a), location=5, scale=2)/(a*(1-a))
    }
  }
  return(result)
}


a = Sys.time()
```

```
integrate(myu, 1/(1+exp(-3)), 1)

b = Sys.time()

b-a

myIntegration(myu2, 1/(1+exp(-3)), 1, option='r')

myIntegration(myu2, 1/(1+exp(-3)), 1, option='t')

myIntegration(myu2, 1/(1+exp(-3)), 1, option='s')

# 첫번째와 결과 같음. 단 시간은 더 오래걸림




# d

myu3 <- function(u){

  k*dnorm(1/u, mean=mu.hat, sd=sqrt(3^2/7))*dcauchy(1/u, location=5, scale=2)*(1/u^2)

}


# c번과 마찬가지로 분포에 u가 있음

# 0에서 적절한 대체값이 필요.


# first way

a = Sys.time()

integrate(myu3, 0.0001, 1/3)

b = Sys.time()

b-a

myIntegration(myu3, 0.0001, 1/3, option='r')

myIntegration(myu3, 0.0001, 1/3, option='t')

myIntegration(myu3, 0.0001, 1/3, option='s')



# second way

myu4 = function(u){

  result = k*dnorm(1/u, mean=mu.hat, sd=sqrt(3^2/7))*dcauchy(1/u, location=5, scale=2)*(1/u^2)

  for (i in seq(1:length(u))){

    if (is.na(result[i])){

      a = 0.00000001

      result[i]   =   k*dnorm(1/a,   mean=mu.hat,   sd=sqrt(3^2/7))*dcauchy(1/a,   location=5,
scale=2)*(1/a^2)

    }

  }

  return(result)

}
```

```r
a = Sys.time()
integrate(myu4, 0, 1/3)
b = Sys.time()
b-a
myIntegration(myu4, 0, 1/3, option='r')
myIntegration(myu4, 0, 1/3, option='t')
myIntegration(myu4, 0, 1/3, option='s')

# 유사한 결과



# 5.4
# a=5
a = 5
myy <- function(y){
  y*(1/y^2)
}

aa = Sys.time()
integrate(myy, (a-1)/a, a-1)
b = Sys.time()
b-aa

myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='r')
myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='t')
myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='s')
log(a)

# a=10
a = 10
aa = Sys.time()
integrate(myy, (a-1)/a, a-1)
b = Sys.time()
b-aa

myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='r')
myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='t')
myIntegration(myy, (a-1)/a, a-1, epsilon = 10^(-4), option='s')
```

```
log(a)
```