homework 8

G201958013 신지은

2021.12.02

8.1

$X \sim \exp(v)$일 때 $\hat{v}_t$는 다음의 과정에 의해 구해진다.

$$\begin{aligned}\hat{v}_t &= \text{argmax}_{v\in V}\mathbb{E}_u[I\{X \geq \gamma_t\}\ln f(X;v)] \\ &= \text{argmax}_{v\in V}\mathbb{E}_u\left[-I\{X \geq \gamma_t\}\ln v - \frac{I\{X \geq \gamma_t\}X}{v}\right] \\ &= \frac{\mathbb{E}_u[I\{X \geq \gamma_t\}X]}{\mathbb{E}_u[I\{X \geq \gamma_t\}]} \\ &= r_t + u\end{aligned}$$

따라서 $r_t = 32, u = 1$이므로 $\hat{v}_t = 33$ 이다.

8.2

```
library(dplyr)
```

```
##
## 다음의 패키지를 부착합니다: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
set.seed(111)
n = 1000
rho = 0.05  # rarity parameter
N = 10^6
gamma = 32
u = 1

#define the functions
W = function(x, u, v) (v/u) * exp(-x * (v - u) / (u * v))

# initialize
v = 1
r = NULL

for(i in 1:3){
```

```
  x = rexp(n, 1/v)
  r[i+1] = -v[i] * log(rho)
  xx = x[x > r[i+1]]
  v[i+1] = sum(W(xx, u, v[i]) * xx) / sum(W(xx, u, v[i]))
}

rbind(r, v)
```

```
##   [,1]     [,2]     [,3]     [,4]
## r   NA 2.995732 11.82233 38.58331
## v    1 3.946391 12.87943 40.04081
```

```
# importance sampling
x = rexp(N, 1 / v[4])
I_hat = mean( W(x, 1, v[4]) * (x > gamma) )
I_hat
```

```
## [1] 1.262586e-14
```

```
# calculate the relative error
## true value
I = integrate(dexp, gamma, Inf)$value
I
```

```
## [1] 1.266402e-14
```

```
## relative error
ggf = function(x) {
  u = 1; v = 40
  return( (v / u^2) * exp( x * (- 2 / u + 1 / v) ) )
  }

Var = ( integrate(ggf, gamma, Inf)$value - I^2 ) / N

sqrt(Var) / I_hat
```

```
## [1] 0.006658985
```

```
# confidence interval
alpha = 0.05
CI = I_hat + c(1, -1) * qnorm(alpha / 2) * sqrt(Var)
CI
```

```
## [1] 1.246108e-14 1.279064e-14
```

```
# check if the true value of I is contained in 95% CI
between(I, CI[1], CI[2])
```

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

8.3

```
# initialize
v_opt = 33
r = NULL

# importance sampling
x = rexp(N, 1 / v_opt)
```

```r
I_hat = mean( W(x, 1, v_opt) * (x > gamma) )
I_hat
```

```
## [1] 1.271136e-14
```

```r
# calculate the relative error
## true value
I = integrate(dexp, gamma, Inf)$value
I
```

```
## [1] 1.266402e-14
```

```r
## relative error
ggf = function(x) {
  u = 1; v = v_opt
  return( (v / u^2) * exp( x * (- 2 / u + 1 / v) ) )
  }

Var = ( integrate(ggf, gamma, Inf)$value - I^2 )
RE = sqrt(Var) / I_hat / sqrt(N)
RE
```

```
## [1] 0.006546926
```

```r
# confidence interval
alpha = 0.05
CI = I_hat + c(1, -1) * qnorm(alpha / 2) * sqrt(Var)
CI
```

```
## [1] -1.503975e-13  1.758203e-13
```

```r
# check if the true value of I is contained in 95% CI
between(I, CI[1], CI[2])
```

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

```r
# CMC estimator
CMC = exp(-32)
CMC
```

```
## [1] 1.266417e-14
```

```r
CMCVar = CMC * (1 - CMC)
CMCVar
```

```
## [1] 1.266417e-14
```

```r
# sample size
CMCVar/ ( RE * CMC )^2
```

```
## [1] 1.84225e+18
```

8.4

```r
set.seed(12345)

# define the function
W = function(x, u, v) exp( -sum(x * (v - u) /(u * v) )) * prod(v / u)

# initialize
```

```r
u = c(1, 1, 0.3, 0.2, 0.1)
N = 10^3
Gam = 6
rho = 0.1
v = u

# run the algorithm
for(t in 1:5){
  if(t == 1) cat("t = 0", ", gamma = _", ", v =", v, '\n')

  x = replicate(N, rexp(5, 1/v))

  S = sapply(1:N, function(j)
    min(x[1,j] + x[4,j], x[1,j] + x[3,j] + x[5,j], x[2,j] + x[3,j] + x[4,j], x[2,j] + x[5,j]))

  gamma = sort(S)[(1 - rho) * N]
  gamma = ifelse(gamma > Gam, Gam, gamma)

  id = which(S > gamma)

  xx = x[, id]    # elite samples
  N1 = length(id)
  w = sapply(1:N1, function(j) W(xx[,j], u, v))

  # update
  v = rowSums( sapply(1:N1, function(j) w[j] / sum(w) * xx[,j]) )
  cat("t =", t, ", gamma =", gamma, ", v =", v, '\n')
}
```

```
## t = 0 , gamma = _ , v = 1 1 0.3 0.2 0.1
## t = 1 , gamma = 1.299463 , v = 2.298672 2.08655 0.2984492 0.205937 0.1099808
## t = 2 , gamma = 2.71904 , v = 3.376338 3.563911 0.4314602 0.2242896 0.122251
## t = 3 , gamma = 4.03578 , v = 4.819086 5.28616 0.3040694 0.2960701 0.1349151
## t = 4 , gamma = 5.726108 , v = 6.319386 6.639746 0.4258949 0.20011 0.1237419
## t = 5 , gamma = 6 , v = 6.643022 6.982539 0.3032662 0.2622569 0.1099499
```

8.5

```r
set.seed(12345)

# define the function
W = function(x, u, v) exp( -sum(x * (v - u) /(u * v) )) * prod(v / u)

# initialize
u = c(1, 1, 0.3, 0.2, 0.1)
N = 10^3
alpha = 0.2
Gam = 10000
rho = 0.1
v = c(1, 1, 1, 1, 1)

# run the algorithm
for(t in 1:5){
```

```
  if(t == 1) cat("t = 0", ", gamma = _", ", v =", v, '\n')

  z = replicate( N, rexp(5, 1/v) )
  x = u * z^{1 / alpha}
  S = sapply(1:N, function(j)
    min(x[1,j] + x[4,j], x[1,j] + x[3,j] + x[5,j], x[2,j] + x[3,j] + x[4,j], x[2,j] + x[5,j]))

  gamma = sort(S)[(1 - rho) * N]
  gamma = ifelse(gamma > Gam, Gam, gamma)

  id = which(S > gamma)

  zz = z[, id]    # elite samples
  N1 = length(id)
  w = sapply(1:N1, function(j) W(zz[,j], u, v))

  # update
  v = rowSums( sapply(1:N1, function(j) w[j] / sum(w) * zz[,j]) )
  cat("t =", t, ", gamma =", gamma, ", v =", v, '\n')
}
```

```
## t = 0 , gamma = _ , v = 1 1 1 1 1
## t = 1 , gamma = 4.208592 , v = 2.1854 2.175826 0.2485883 0.210287 0.07278586
## t = 2 , gamma = 109.2883 , v = 3.410634 3.516961 0.4574106 0.1769459 0.09115396
## t = 3 , gamma = 894.094 , v = 4.848718 5.060608 0.2982611 0.3138284 0.08752262
## t = 4 , gamma = 4896.501 , v = 6.040548 6.214636 0.2959034 0.147542 0.1065969
## t = 5 , gamma = 10000 , v = 7.758173 7.595417 0.3046999 0.1577836 0.1038276
```

make new table

```
set.seed(12345)

# define the function
W = function(x, u, v) exp( -sum(x * (v - u) /(u * v) )) * prod(v / u)

# initialize
u = c(1, 1, 0.3, 0.2, 0.1)
N = 10^3
alpha = 5
Gam = 2
rho = 0.1
v = c(1, 1, 1, 1, 1)

# run the algorithm
for(t in 1:10){
  if(t == 1) cat("t = 0", ", gamma = _", ", v =", v, '\n')

  z = replicate( N, rexp(5, 1/v) )
  x = u * z^{1 / alpha}
  S = sapply(1:N, function(j)
    min(x[1,j] + x[4,j], x[1,j] + x[3,j] + x[5,j], x[2,j] + x[3,j] + x[4,j], x[2,j] + x[5,j]))

  gamma = sort(S)[(1 - rho) * N]
  gamma = ifelse(gamma > Gam, Gam, gamma)
```

```
  id = which(S > gamma)

  zz = z[, id]    # elite samples
  N1 = length(id)
  w = sapply(1:N1, function(j) W(zz[,j], u, v))

  # update
  v = rowSums( sapply(1:N1, function(j) w[j] / sum(w) * zz[,j]) )
  cat("t =", t, ", gamma =", gamma, ", v =", v, '\n')
}
```

```
## t = 0 , gamma = _ , v = 1 1 1 1 1
## t = 1 , gamma = 1.152576 , v = 1.762386 3.253925 0.2523653 0.292191 0.1247686
## t = 2 , gamma = 1.317967 , v = 3.276577 4.308504 0.3237641 0.2558097 0.1075148
## t = 3 , gamma = 1.430576 , v = 4.689255 6.370367 0.2999087 0.3266945 0.1395417
## t = 4 , gamma = 1.535705 , v = 5.749833 7.796578 0.418306 0.2961415 0.1401399
## t = 5 , gamma = 1.624056 , v = 7.936585 10.40142 0.3242105 0.3142981 0.1789565
## t = 6 , gamma = 1.703967 , v = 10.19481 13.1367 0.4404162 0.3242895 0.1514181
## t = 7 , gamma = 1.78076 , v = 13.03368 15.89305 0.3491608 0.4180354 0.1116494
## t = 8 , gamma = 1.866984 , v = 16.42539 19.4412 0.3911974 0.1909829 0.2162133
## t = 9 , gamma = 1.924702 , v = 20.94386 23.22687 0.1740067 0.1422523 0.3298813
## t = 10 , gamma = 2 , v = 20.30412 28.78071 0.1063461 0.8445343 0.1522298
```

8.6

```
set.seed(12345)

# define the function
W = function(x, u, v) exp( -sum(x * (v - u) /(u * v) )) * prod(v / u)

# initialize
u = c(1, 1, 0.3, 0.2, 0.1)
N = 10^3
alpha = 0.2
Gam = 10000
rho = 0.1
v = c(1, 1, 1, 1, 1)

# run the algorithm
for(t in 1:5){
  if(t == 1) cat("t = 0", ", gamma = _", ", v =", v, '\n')

  x = replicate( N, rweibull(5, alpha, 1/v) )
  S = sapply(1:N, function(j)
    min(x[1,j] + x[4,j], x[1,j] + x[3,j] + x[5,j], x[2,j] + x[3,j] + x[4,j], x[2,j] + x[5,j]))

  gamma = sort(S)[(1 - rho) * N]
  gamma = ifelse(gamma > Gam, Gam, gamma)

  id = which(S > gamma)

  xx = x[, id]    # elite samples
  N1 = length(id)
```

```
  # w = sapply(1:N1, function(j) W(xx[,j], u, v))j=1
  w = sapply(1:N1, function(j) prod(dweibull(xx[,j], alpha, 1/u) / dweibull(xx[,j], alpha, 1/v)))

  # update
  # zz = z[, id]    # elite samples
  v = rowSums( sapply(1:N1, function(j) w[j] / sum(w) * xx[,j]^{alpha}) )
  v = v^{1/alpha}
  cat("t =", t, ", gamma =", gamma, ", v =", v, '\n')
}
```

```
## t = 0 , gamma = _ , v = 1 1 1 1 1
## t = 1 , gamma = 7.871645 , v = 8.333738 9.308146 15.01881 86.00694 67.49439
## t = 2 , gamma = 0.431777 , v = 1.958129 15.62407 0.9640232 0.1398426 3.7701
## t = 3 , gamma = 6.300248 , v = 12.83321 2.330155 8.648915 43.51792 86.45658
## t = 4 , gamma = 0.6860065 , v = 68.38577 16.65706 19.53609 288.3174 132.5222
## t = 5 , gamma = 0.1155124 , v = 0.3318628 0.3905883 3.333455 0.7043272 7.769172
```