MATH5835M Practical Solutions

Jochen Voss

2023-12-12

These are example solutions for the statistical computing practical, together with a marking scheme.

## Task 1

If is the wind speed (in miles per hour), then the Monte Carlo Estimate for the probability is

where the are i.i.d. copies of . We can use the following R code to compute this estimate. This uses the function wind\_speed() given on the practical task sheet.

wind\_speed <- function(n, w, mu, sigma) {  
 k <- length(w)  
 i <- sample(k, n, replace = TRUE, prob = w)  
 rlnorm(n, mu[i], sigma[i])  
}  
w <- c(0.1, 0.4, 0.5)  
mu <- c(2.1, 1.6, -0.5)  
sigma <- c(0.55, 0.7, 0.6)  
  
N <- 1e6  
X <- wind\_speed(N, w, mu, sigma)  
p.MC <- mean(X >= 40)  
p.MC

## [1] 0.000756

Here we used, somewhat arbitrarily, . We will revisit this choice of  below. The R output shows that the computed estimate is .

The Root Mean Squared Error (RMSE) for this estimator is

where the variance can be approximated using the sample variance of the samples already used in the Monte Carlo Estimate:

RMSE.MC <- sqrt(var(X >= 40) / N)  
RMSE.MC

## [1] 2.748507e-05

Thus, the RMSE is approximately . We have , *i.e.* the RMSE is nearly of the estimated value. One could try to reduce the RMSE by increasing , but since we will try an improved method to estimate in the next task, we here accept this estimate as being `good enough’.

**Marking criteria:**

* The estimate is computed correctly.
* The RMSE is computed correctly.
* There is some (possibly very short) discussion about how was chosen.

## Task 2

The importance sampling estimate for is given by

where is the density of and the are i.i.d. samples with density . The distribution of the samples can be chosen arbitrarily, as long as for all , but the method is most efficient if is approximately proportional to .

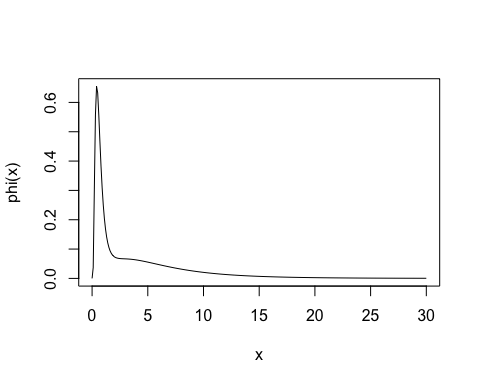
Here, the distribution of is a mixture of log-normal distribution and thus the density is

where is the density of the log-normal distribution (given on the practical task sheet or available as dlnorm() in R). We can compute in R using the following function:

phi <- function(x) {  
 phi1 <- dlnorm(x, mu[1], sigma[1])  
 phi2 <- dlnorm(x, mu[2], sigma[2])  
 phi3 <- dlnorm(x, mu[3], sigma[3])  
 w[1]\*phi1 + w[2]\*phi2 + w[3]\*phi3  
}

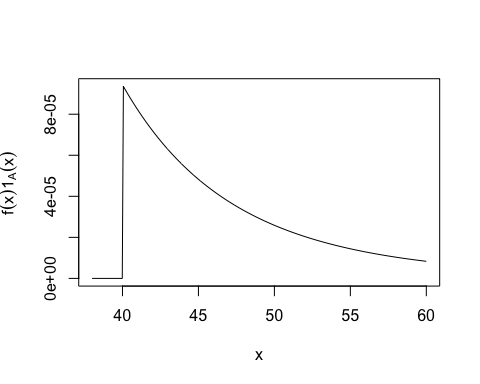
Here is a plot of :

x <- seq(0, 30, length.out = 300)  
plot(x, phi(x), type = "l")



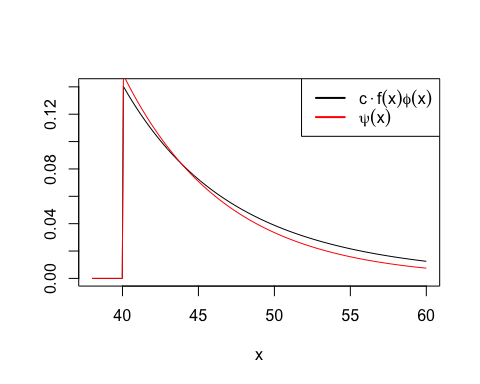
And a plot of :

x <- seq(38, 60, length.out = 300)  
plot(x, phi(x)\*(x >= 40), type = "l",  
 ylab = expression(f(x)\*1[A](x)))



The second plot suggests that a good distribution for might be a shifted exponential distribution: . Experiments, like in the following plot, show that might be a good choice of rate for the exponential distribution.

x <- seq(38, 60, length.out = 300)  
plot(x, 1500\*phi(x)\*(x >= 40), type = "l", ylab = "")  
lines(x, dexp(x-40, rate=0.15), col = "red")  
legend("topright",  
 legend = c(expression(c %.% f(x)\*phi(x)), expression(psi(x))),  
 col = c("black", "red"), lwd = 2)



In the plot the function is scaled by the (experimentally determined) factor to make it visible on the same scale as the exponential density.

The importance sampling estimate can be computed in R using code like the one shown below. Since is automatically true here, it is not required to include the indicator function in the R code.

psi <- function(x) dexp(x - 40, rate = 0.15)  
  
N <- 1e6  
Y <- rexp(N, rate = 0.15) + 40  
p.IS <- mean(phi(Y) / psi(Y))  
p.IS

## [1] 0.0007621195

The result is very similar to the result from the basic Monte Carlo method, so hopefully our implementation is correct.

The root mean squared error for this estimate is

Again, the variance in this formula can be approximated using the sample variance of the samples used in the Importance Sampling Estimate:

RMSE.IS <- sqrt(var(phi(Y) / psi(Y)) / N)  
RMSE.IS

## [1] 5.250842e-07

We note that the error obtained here is very small compared to the estimate , so the chosen was large enough.

Since we used the same for both estimates, the RMSE values are directly comparable: For the basic Monte Carlo method has RMSE and the Importance Sampling method has RMSE . Thus, for fixed sample size, Importance Sampling has considerably smaller error than basic Monte Carlo or, conversely, the Importance Sampling needs much fewer samples to achieve the same level of error than Monte Carlo.

**Marking Criteria:**

* The density of is determined correctly (this was not done in lectures, so should get some credit)
* The estimate and RMSE are computed correctly.
* The distribution of is chosen well and the choice is explained well.
* There is some (possibly very short) discussion about how was chosen.
* There is some meaningful comparison of the RMSE for the two estimates.

## Task 3

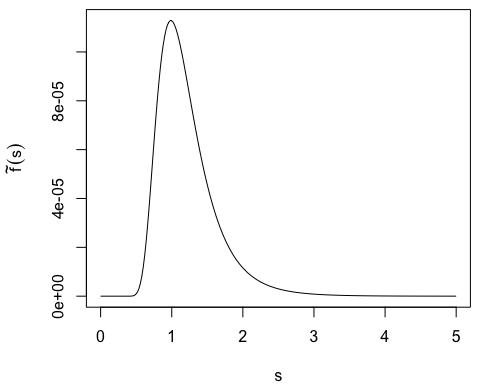
(Note that the roles of and in the practical are swapped, compared to the notation used in the book.)

We can compute the non-normalised target density in R as follows.

x <- c(0.50, 1.67, 2.22, 0.22, 4.36)  
pi <- function(s) dexp(s)  
tilde.f <- function(s) {  
 phi1 <- dlnorm(x[1], 0, s)  
 phi2 <- dlnorm(x[2], 0, s)  
 phi3 <- dlnorm(x[3], 0, s)  
 phi4 <- dlnorm(x[4], 0, s)  
 phi5 <- dlnorm(x[5], 0, s)  
 pi(s) \* phi1 \* phi2 \* phi3 \* phi4 \* phi5  
}

A plot of is shown below. The plot shows that is unimodal, with a maximum at .

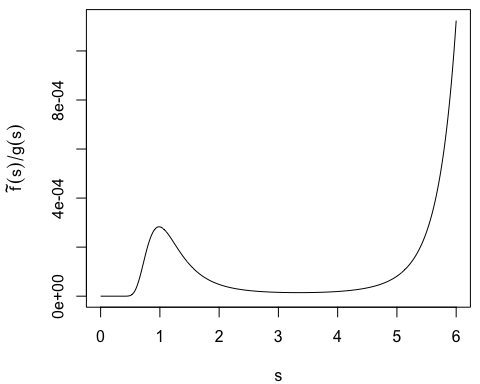
s <- seq(0, 5, by=0.02)  
par(mai = c(0.8, 0.9, 0.1, 0.1))  
plot(s, tilde.f(s), type="l", ylab = expression(tilde(f)(s)))



To implement envelope rejection sampling, we need to choose a proposal density  
 which is easy to sample from, and such that we can find a constant with for all . The method is most efficient if is large. To achieve this, we should choose to be approximately proportional to , and in particular should be large for . Also, must have heavy enough tails to cover the tails of .

**Attempt 1:** Use -distributed proposals.

s <- seq(0, 6, by=0.02)  
par(mai = c(0.8, 0.9, 0.1, 0.1))  
plot(s, tilde.f(s) / dnorm(s, mean = 1), type = "l",  
 ylab = expression(tilde(f)(s)/g(s)))



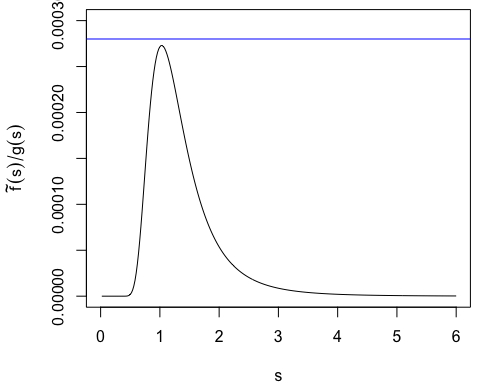
This does not work: the ratio starts increasing as and clearly is not bounded: the tails of are `heavier’ than these of a normal distribution. (Closer inspection of the formula for reveals that as .)

Possible choices of the proposal distribution include:

* Gamma distribution: To make sure that the gamma distribution has heavy enough tails, we need .
* Exponential distribution: the rate must be .

**Attempt 2:** Use a Gamma distribution. After a bit of experimenting we choose and , to get a reasonably small value of . The following plot shows that seems to be bounded by . (To allow for numerical error, we leave a small gap between the bound and the numerical ratio.)

s <- seq(0, 6, by=0.02)  
par(mai = c(0.8, 0.9, 0.1, 0.1))  
plot(s, tilde.f(s) / dgamma(s, 1.5, 1), type = "l", ylim = c(0, 0.0003),  
 ylab = expression(tilde(f)(s)/g(s)))  
  
c <- 0.00028  
abline(h = c, col = "blue")

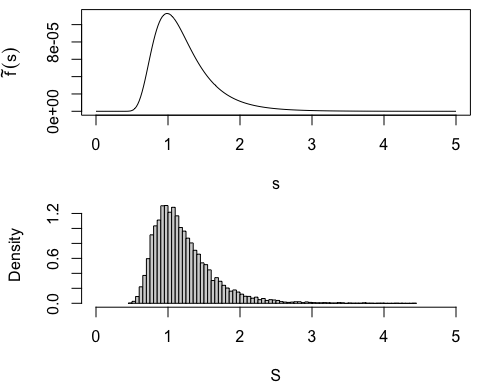


Now we can implement envelope rejection sampling:

N <- 10000  
S <- numeric(0)  
while (length(S) < N) {  
 prop <- rgamma(1, 1.5, 1)  
 U <- runif(1)  
 if (c \* dgamma(prop, 1.5, 1) \* U <= tilde.f(prop)) {  
 S <- c(S, prop)  
 }  
}

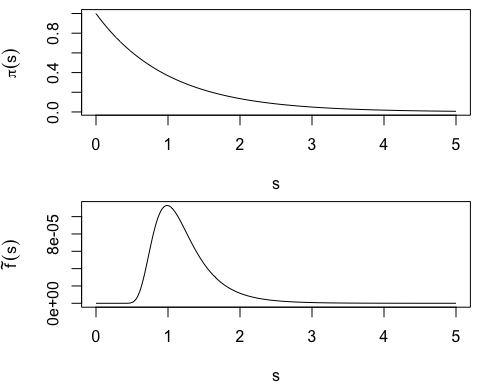
We plot a histogram of the samples together with the non-normalised density . Since the histogram has the same shape as the graph of , hopefully our implementation is correct.

s <- seq(0, 5, by=0.02)  
par(mfrow = c(2, 1), mai = c(0.8, 0.85, 0.1, 0.1))  
plot(s, tilde.f(s), type = "l", ylab = expression(tilde(f)(s)))  
hist(S, prob = TRUE, main = NULL, breaks=100, xlim=c(0, 5))

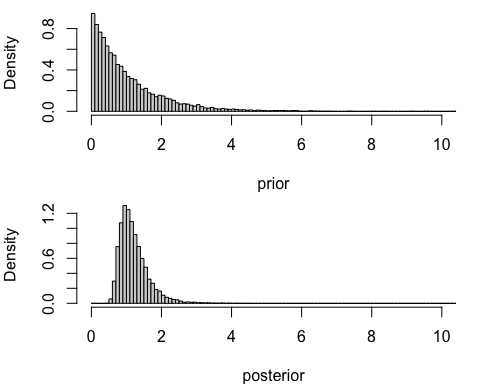


To compare the posterior to the prior we can either compare the densities, or the histograms.

s <- seq(0, 5, by=0.02)  
par(mfrow = c(2, 1), mai = c(0.8, 0.85, 0.1, 0.1))  
plot(s, pi(s), type = "l", ylab = expression(pi(s)))  
plot(s, tilde.f(s), type = "l", ylab = expression(tilde(f)(s)))



par(mfrow = c(2, 1), mai = c(0.8, 0.8, 0.1, 0.1))  
S0 <- rexp(N)  
breaks <- seq(0, max(S0)+0.1, by=0.1)  
hist(S0, breaks=breaks, prob=TRUE, main = NULL, xlab="prior")  
hist(S, breaks=breaks, prob=TRUE, main = NULL, xlab="posterior")



Both sets of pictures show that the posterior is much more concentrated than the prior. This is, because the posterior includes additional information about from the observations.

Alternatively, one could also compare the distributions using statistical measures like the mean and the variance.

**Marking Criteria:**

* The plot of is correct and has reasonable labels.
* and are chosen appropriately.
* In case problems with the tails have been overlooked, any resulting discrepancies have been noticed.
* Envelope Rejection Sampling has been implemented correctly.
* The histogram of the generated samples is correct and has reasonable labels.
* There is some meaningful comparison between the prior and posterior.