Module 7: Solutions for supplementary exercises

Exercise: Potential rejection sampling problems

First try to answer the following questions without using the computer – then reuse the code from the supplementary slides to check your answer:

- Suppose we could not easily determine M and hence had to make a conservative choice; say M = 100 or M = 500 in this context.
  1. Which effect will that have on the number of accepted samples? **The acceptance rate goes down.**

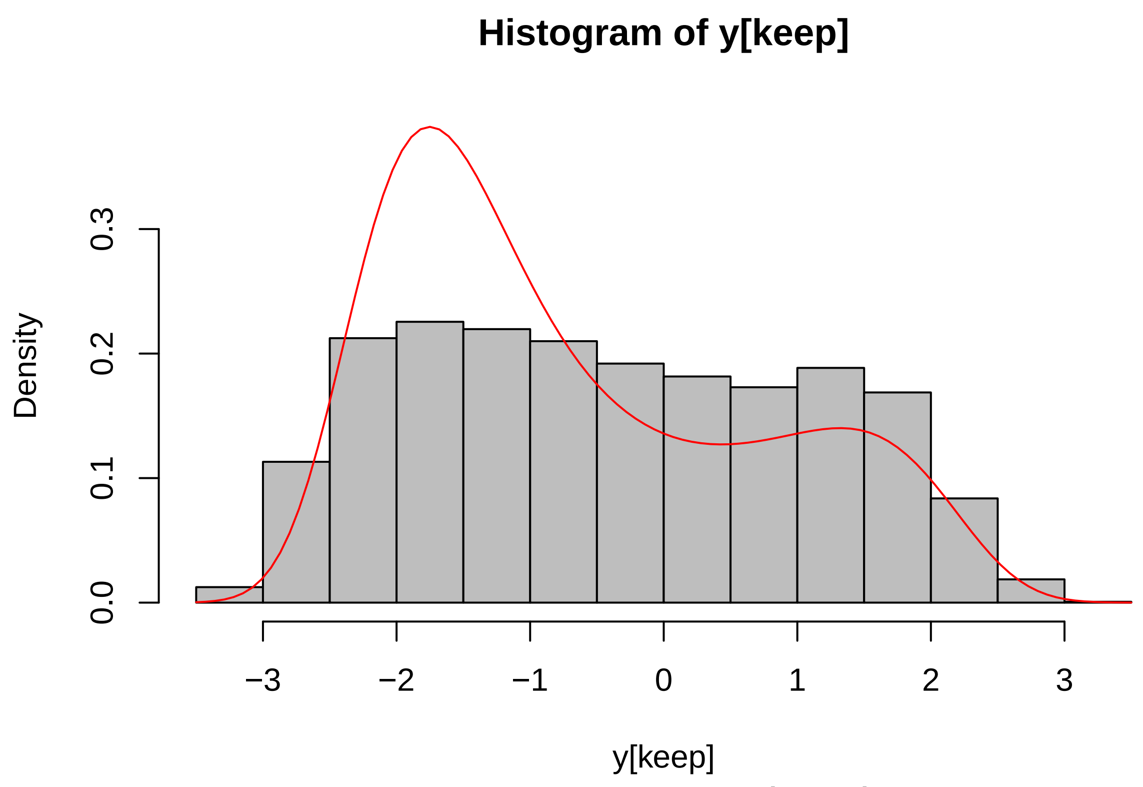
```
f0 <- function(x, a=.4, b=.08){exp(a * (x - a)^2 - b * x^4)}
N <- 10000
M <- 500
y <- runif(N, -4, 4)
p_accept <- f0(y)/(M*dunif(y, -4, 4))
u <- runif(N, 0, 1)
keep <- u<p_accept
mean(keep)
```

```
## [1] 0.0175
```

```
2. How would you have to compensate for a too large value of M if you want a
given number of samples from the target distribution?
**Increase the number of proposals, which will increase the overall computation time.**
```

- What happens if you do not choose M large enough (e.g. M = 10 in our example)? **Then you sample from the wrong distribution.**

```
M <- 10
y <- runif(N, -4, 4)
p_accept <- f0(y)/(M*dunif(y, -4, 4))
u <- runif(N, 0, 1)
keep <- u<p_accept
hist(y[keep], prob = TRUE, col = "gray", ylim = c(0, .38))
norm_const <- integrate(f0, -4, 4)$value
curve(f0(x)/norm_const, col = "red", add = TRUE)
```



- What would be the effect of using a uniform proposal distribution on $[-10, 10]$? **Acceptance rate goes down since proposals outside $[-4, 4]$ always are rejected.**

```
M <- 3.1/20
y <- runif(N, -20, 20)
p_accept <- f0(y)/(M*dunif(y, -20, 20))
u <- runif(N, 0, 1)
keep <- u<p_accept
mean(keep)
```

```
## [1] 0.1692
```

- What happens if the proposal distribution is an standard normal distribution (i.e. mean zero and standard deviation 1? Hints:
    1. You can use `dnorm()` for the normal density.
    2. You may have to create a sequence `x <- seq(-4, 4, by = 0.01)` to numerically evaluate the bound M relating `f0(x)` and `dnorm(x)`.

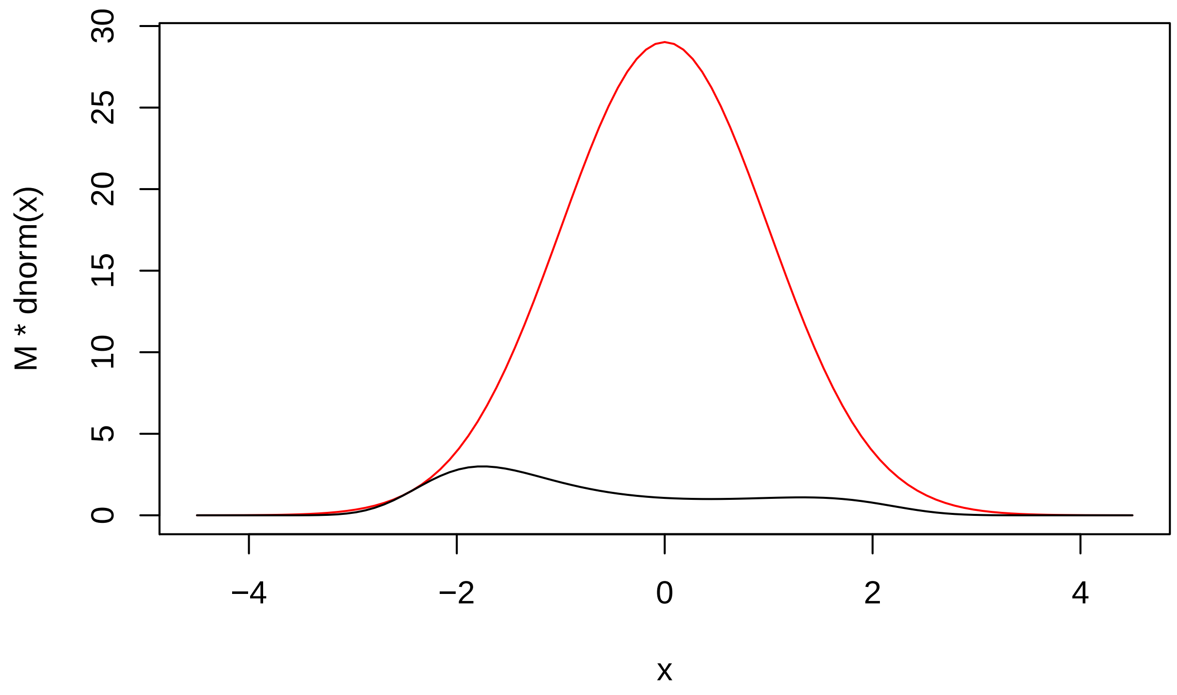
The upper bound is higher and we get lower acceptance rates. I.e. we have to sample for longer time to obtain the number of target samples we want.

```
x <- seq(-4, 4, by = 0.01)
M <- max(f0(x)/dnorm(x))
y <- rnorm(N)
p_accept <- f0(y)/(M*dnorm(y))
u <- runif(N, 0, 1)
keep <- u<p_accept
mean(keep)
```

```
## [1] 0.1047
```

The reason that the acceptance rate is so low is that the upper bound is very poor:

```r
curve(M*dnorm(x), col = "red", from = -4.5, to = 4.5)
curve(f0(x), add = TRUE)
```



Exercise: Improving the proposal distribution

If $f(x)$, $x \in [0,1]$ is a pdf on $[0,1]$ then for $a > 0$, $1/a \cdot f(x/a)$, $x \in [0,a]$ is a pdf on $[0,a]$. Furthermore, a pdf on $[b, a+b]$ can be obtained by simple translation.

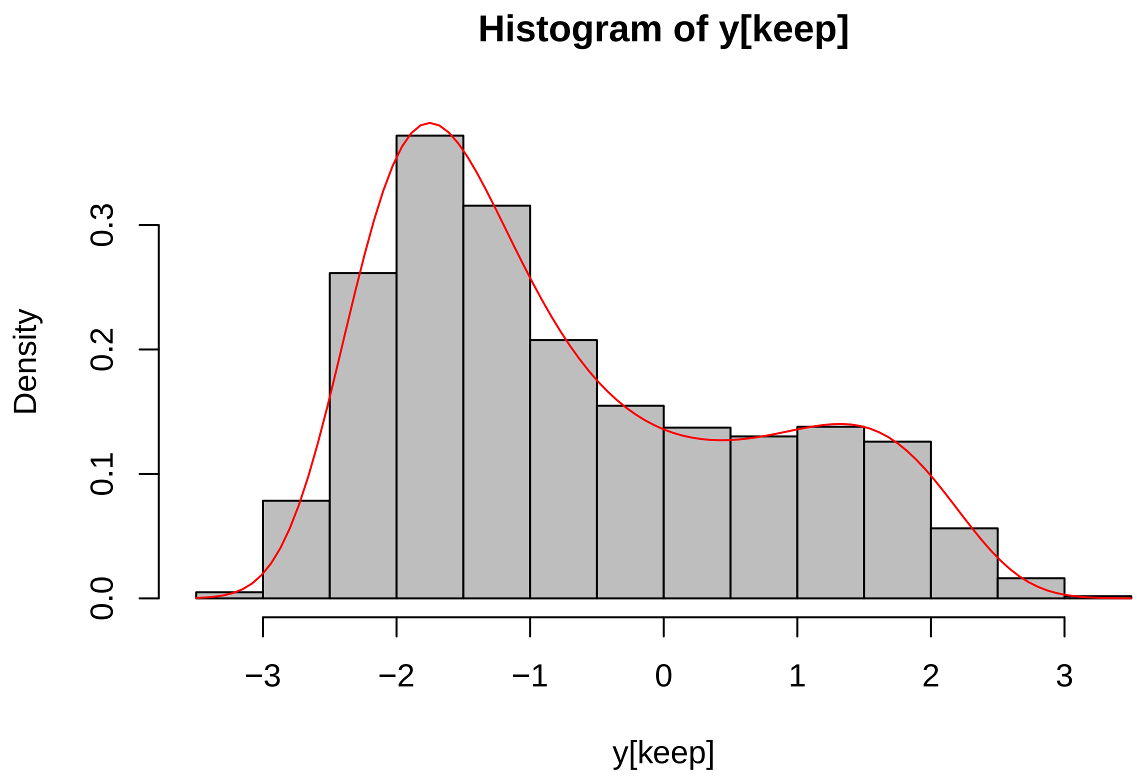
- Based on these facts how can a beta distribution $\mathrm{Be}(\alpha, \beta)$ indirectly be used as the proposal distribution for our example? -Implement the rejection sampling algorithm using Be(2.5,3.5) transformed to $[-4.1, 4.1]$ (but with M determined on $[-4,4]$).
- Check with a histogram that you are sampling the correct distribution.
- Find the acceptance rate.

For $X \sim \mathrm{Be}(2.5, 3.5)$ transform it to the interval $I = [-4.1, 4.1]$ by $Y = 8.2 \cdot X - 4.1$. Then the density for Y is $f((y+4.1)/8.2)/8.2$ for x in I and zero otherwise, where f is the original Beta density on $[0,1]$:

```r
x <- seq(-4, 4, by = 0.01)
g <- function(y, a = 2.5, b = 3.5){x <- (y+4.1)/8.2; dbeta(x, a, b)/8.2}
M <- max(f0(x)/g(x))
y <- 8.2*rbeta(N, 2.5, 3.5) - 4.1
p_accept <- f0(y)/(M*g(y, 2.5, 3.5))
u <- runif(N, 0, 1)
keep <- u<p_accept
mean(keep)
```

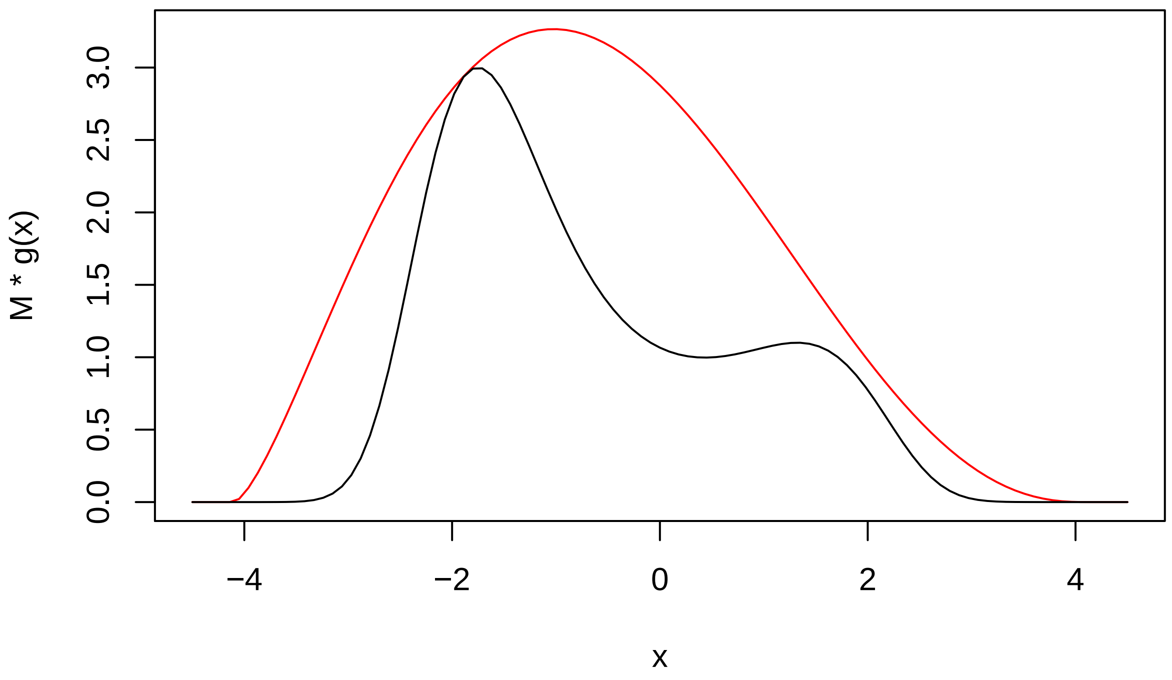
```
## [1] 0.5685
```

```
hist(y[keep], prob = TRUE, col = "gray", ylim = c(0, .38))
curve(f0(x)/norm_const, col = "red", add = TRUE)
```



Now the upper bound is much better:

```
curve(M*g(x), col = "red", from = -4.5, to = 4.5)
curve(f0(x), add = TRUE)
```



M * g(x)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
−4
−2
0
2
4
x