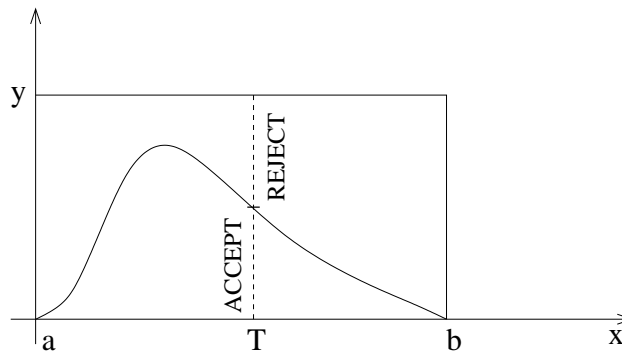


Rejection sampling algorithm.

- *Step 1:* Generate T with the density m , where $f(t) < l \times m(t) = M(t)$, $l = \text{const}$.
Sampling from $f(x)$ distribution is hard. Sampling from distribution $m(x)$ is easy.
- *Step 2:* Generate U , uniform on $[0, 1]$ and independent of T .

If $M(T) \times U \leq f(T) \rightarrow$ accept, then set $X = T$.

Otherwise \rightarrow reject, go back to step 1



Why does this work?

Let A be a subset of $[a : b]=\text{support}(f(\cdot))$, a and/or b may be infinite. To show that:

$$P(X \in A) = \int_A f(t) dt, \text{ we expand the left hand side}$$

$$P(X \in A) = P(T \in A \mid \text{Accept}) = \frac{P(T \in A \text{ and Accept})}{P(\text{Accept})}$$

Condition on $T = t$ ($I = \int_a^b M(t) dt$, since $m(t)$ is a density and $I \times m(t) = M(t)$)

$$\begin{aligned} P(T \in A \text{ and Accept}) &= \int_a^b P(T \in A \text{ and Accept} \mid T = t) m(t) dt \\ &= \int_a^b P(U \leq f(t)/M(t) \text{ and } t \in A) m(t) dt \\ &= \int_A \frac{f(t)}{M(t)} m(t) dt = \frac{1}{I} \int_A f(t) dt \end{aligned}$$

Similarly

$$P(\text{Accept}) = \int_a^b P(\text{Accept} \mid T = t) m(t) dt = \int_a^b \frac{f(t)}{M(t)} m(t) dt = \frac{1}{I}.$$

Remark : High efficiency if algorithm accepts with high probability, i.e. M close to f .

Example

Suppose we want to sample from a density whose graph is shown below.

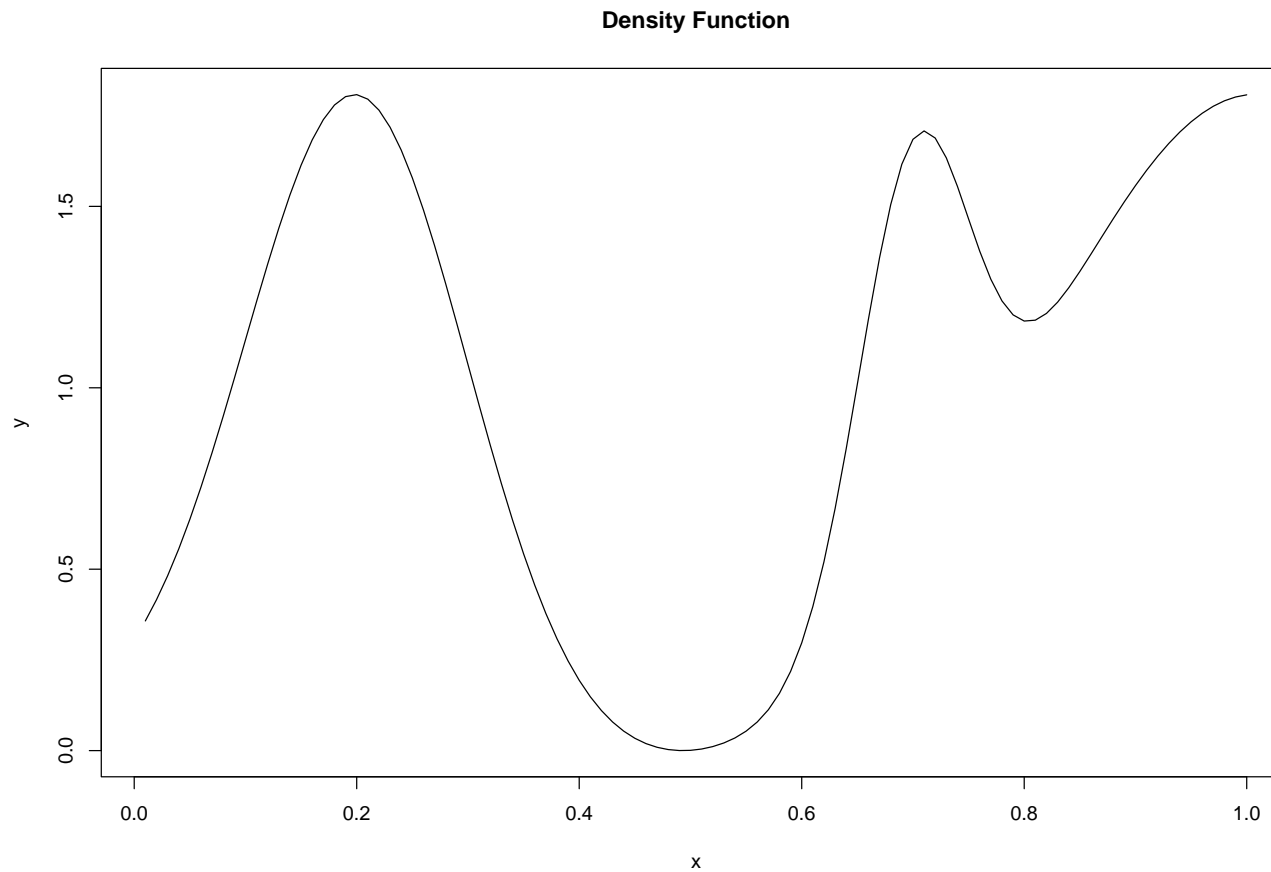


Figure 1: Density function

In this case we let $M(T)$ be the maximum of f over the interval $[0, 1]$, namely

$$M(x) = \max(f), \quad 0 \leq x \leq 1$$

so that m is the uniform density over the interval $[0, 1]$.

Implementation

R : Copyright 2000, The R Development Core Team
Version 1.0.1 (April 14, 2000)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type `"?license"` or `"?licence"` for distribution details.

R is a collaborative project with many contributors.
Type `"?contributors"` for a list.

Type `"demo()"` for some demos, `"help()"` for on-line help, or
`"help.start()"` for a HTML browser interface to help.
Type `"q()"` to quit R.

```
x <- 0:100
M <- max(knownDensity(x))
```

Routine for sampling once from the density f

```
OK <- 0
while(OK<1)
{
  # Generate T
  T <- runif(1, min = 0, max = 1)
  # Generate U
  U <- runif(1, min = 0, max = 1)
  if(M*U <= knownDensity(T))
  {
    OK <- 1
    RN <- T
  }
}
```

This routine will sample n iid samples from the density f

```
RejectionSampling <- function(n)
{
  RN <- NULL
  for(i in 1:n)
  {
    OK <- 0
    while(OK<1)
    {
      T <- runif(1,min = 0, max = 1)
      U <- runif(1,min = 0, max = 1)
      if(U <= knownDensity(T))
      {
        OK <- 1
        RN <- c(RN,T)
      }
    }
  }
  return(RN)
}
# Demo:: R-File: R_scriptHelp.txt
# C:\Documents and Settings\ivo\Desktop\Applications\R
```

Visualization of the results

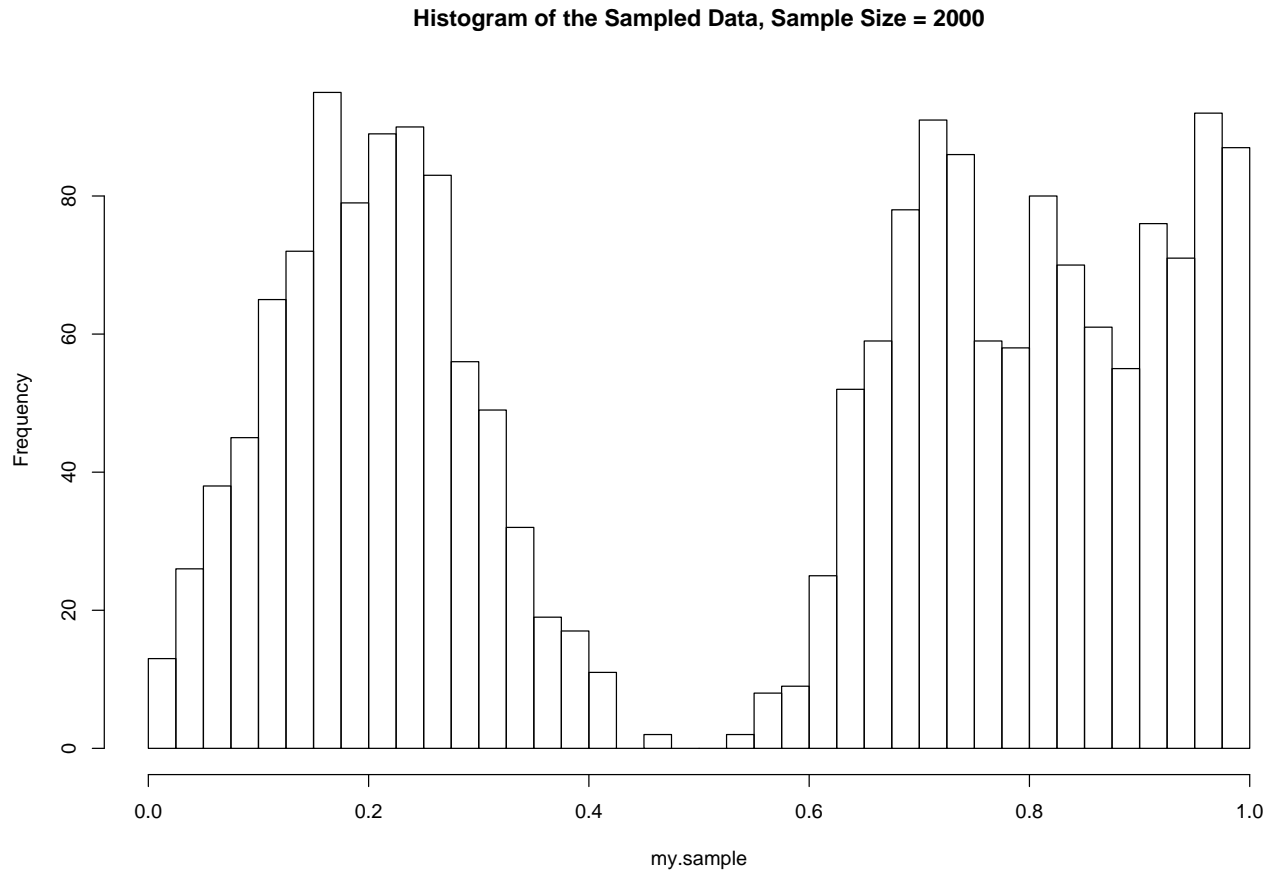


Figure 2: Histogram of the Sampled Data

```

#      1. Define a density of interest that will be approximated by "REJECTION
#      SAMPLING"
minRgDensity <- 0
maxRgDensity <- 10
maxDensityValue <- 1
sampleSize <- 3000

knownDensity <- function(x)
{
  minRgDensity <- 0
  maxRgDensity <- 20
  maxDensityValue <- 1
  return(dbeta(x, 3, 10))
}

rawDensity <- rbeta(sampleSize, 3, 10)

#      2. Rejection sampling method
RejectionSampling <- function(n)
{
  RN <- NULL
  for(i in 1:n)
  {
    OK <- 0
    while(OK<1)
    {
      T <- runif(1,min = minRgDensity, max = maxRgDensity )
      U <- runif(1,min = 0, max = 1)
      if(U*maxDensityValue <= knownDensity(T))
      {
        OK <- 1
        RN <- c(RN,T)
      }
    }
  }
  return(RN)
}

#      3. Generate n=sampleSize samples from the model-simulation density
#      (RejectionSampling)
simulatedDensity <- RejectionSampling(sampleSize)

#      4. Calculate the two histograms
histoRaw <- hist(rawDensity)
histoSimulated <- hist(simulatedDensity)

#      5. Q-Q plot raw vs simulated densities
plot( rawDensity )
plot( simulatedDensity, rawDensity )
qqplot(simulatedDensity, rawDensity )
qqline(simulatedDensity, col = 2)

# qqplot( rawDensity, simulatedDensity);
# abline(0,1)

#      6. for comparison Q-Q plot of simulated Beta is quite diff from N(0,1)

```

```
#qqplot(simulatedDensity, rnorm(1:sampleSize, 0, 1))
```