Exercise 1

TMA4300 Computer Intensive Statistical Models

Mads Adrian Simonsen, William Scott Grundeland Olsen

08 februar, 2021

Problem A: Stochastic simulation by the probability integral transform and bivariate techniques

Subproblem 1.

Let $X \sim \text{Exponential}(\lambda)$, with the cumulative density function

$$F_X(x) = 1 - e^{-\lambda x}, \quad x \geq 0.$$

Then the random variable $Y := F_X(X)$ has a $\text{Uniform}(0,1)$ distribution. The probability integral transform becomes

$$Y = 1 - e^{-\lambda X} \quad \Leftrightarrow \quad X = -\frac{1}{\lambda}\ln(1-Y).$$

It is clear that if $U \sim \text{Uniform}(0,1)$, then $1-U \sim \text{Uniform}(0,1)$, and therefore we may as well say that

$$X = -\frac{1}{\lambda}\ln(Y). \tag{1}$$

Thus, we sample Y from `runif()` and transform it using Equation (1), to sample from the exponential distribution. Figure 1 shows one million samples drawn from the `generate_from_exp()` function defined in the code chunk below. It also shows the theoretical PDF of the exponential distribution with rate parameter $\lambda = 2$.

```
set.seed(69)

generate_from_exp <- function(n, rate) {
  Y <- runif(n)   # Generate n Uniform(0, 1) variables
  X <- -(1 / rate) * log(Y)   # Transformation
  return(X)
}

# sample
n <- 1000000  # One million samples
lambda <- 2
exp_samp <- generate_from_exp(n, lambda)

# plot
ggplot() +
  geom_histogram(
```

```r
    data = as.data.frame(exp_samp),
    mapping = aes(x = exp_samp, y = ..density..),
    binwidth = 0.05,
    boundary = 0
  ) +
  stat_function(
    fun = dexp,
    args = list(rate = lambda),
    aes(col = "Theoretical density")
  ) +
  ylim(0, lambda) +
  xlim(0, 2) +
  ggtitle("Simulating from an exponential distribution") +
  xlab("x") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

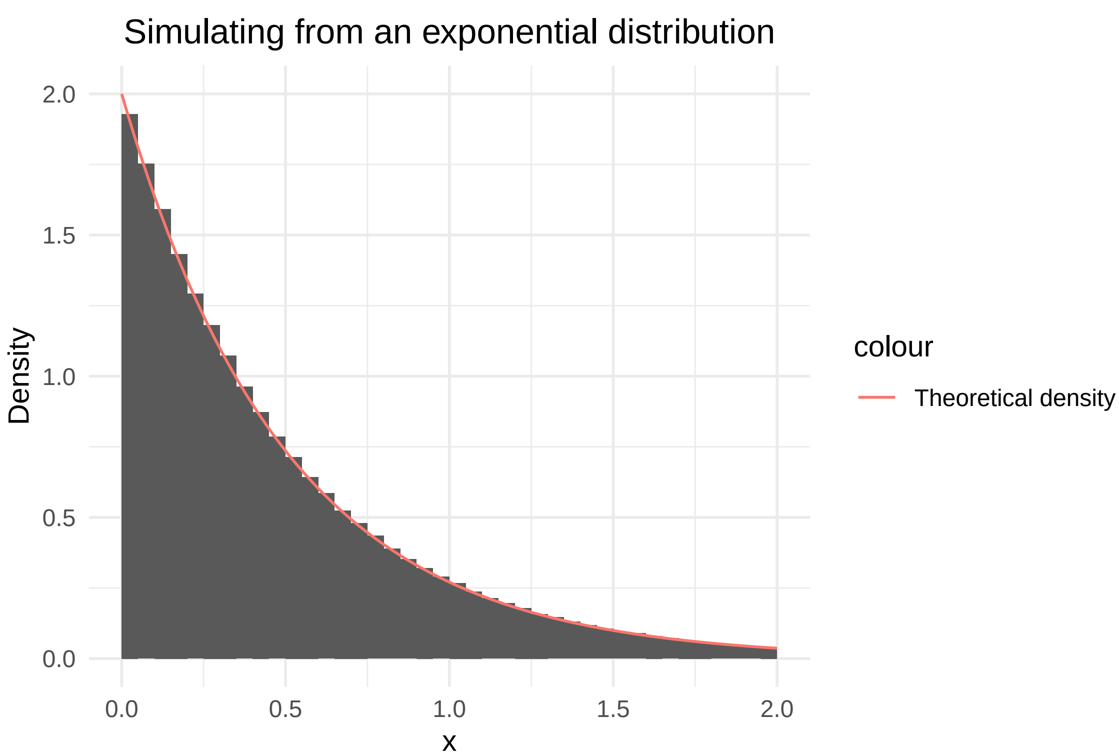

Figure 1: Normalized histogram of one million samples drawn from the exponential distribution, together with the theoretical PDF, with $\lambda = 2$.

Theoretically, the mean and variance of $X \sim \text{Exponential}(\lambda)$ is $\text{E}(X) = \lambda^{-1}$ and $\text{Var}(X) = \lambda^{-2}$. So for $\lambda = 2$ we would expect $\text{E}(X) = 1/2$ and $\text{Var}(X) = 1/4$. For the simulation we get the mean and variance as calculated in the code block below, showing what we would expect.

```r
mean(exp_samp)
```

```
## [1] 0.499487
```

```
var(exp_samp)
```

```
## [1] 0.24952
```

Subproblem 2.

Subsubproblem (a)

We are considering the probability density function

$$g(x) = \begin{cases} cx^{\alpha-1} & \text{if } 0 < x < 1, \\ c\mathrm{e}^{-x} & \text{if } x \geq 1, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where c is a normalizing constant and $\alpha \in (0,1)$. If $x \leq 0$ the cumulative distribution function is zero. In the interval $0 < x < 1$ it becomes

$$G(x) = \int_{-\infty}^{x} g(\xi)\,\mathrm{d}\xi = \int_{0}^{x} c\xi^{\alpha-1}\,\mathrm{d}\xi = \frac{c}{\alpha}[\xi^{\alpha}]_0^x = \frac{c}{\alpha}x^{\alpha},$$

and finally for $x \geq 1$ we have

$$G(x) = \int_{-\infty}^{x} g(\xi)\,\mathrm{d}\xi = \int_{0}^{1} c\xi^{\alpha-1}\,\mathrm{d}\xi + \int_{1}^{x} c\mathrm{e}^{-\xi}\,\mathrm{d}\xi = \left[\frac{c}{\alpha}\xi^{\alpha}\right]_0^1 - \left[c\mathrm{e}^{-\xi}\right]_1^x = c\left(\frac{1}{\alpha} - \mathrm{e}^{-x} + \frac{1}{\mathrm{e}}\right),$$

for $\alpha \in (0,1)$. That is, the cumulative density function is

$$G(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ \frac{c}{\alpha}x^{\alpha} & \text{if } 0 < x < 1, \\ c\left(\frac{1}{\alpha} - \mathrm{e}^{-x} + \frac{1}{\mathrm{e}}\right) & \text{if } x \geq 1. \end{cases}$$

In this case it is trivial to find c. We solve

$$1 = \int_{\mathbb{R}} g(x)\,\mathrm{d}x = \int_0^1 cx^{\alpha-1}\,\mathrm{d}x + \int_1^{\infty} c\mathrm{e}^{-x}\,\mathrm{d}x = \frac{c}{\alpha} + \frac{c}{\mathrm{e}},$$

which gives that

$$c = \frac{\alpha\mathrm{e}}{\alpha + \mathrm{e}}.$$

Writing the cumulative density function using this as c we obtain

$$G(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ \frac{\mathrm{e}}{\alpha+\mathrm{e}}x^{\alpha} & \text{if } 0 < x < 1, \\ 1 - \frac{\alpha}{\alpha+\mathrm{e}}\mathrm{e}^{1-x} & \text{if } x \geq 1, \end{cases}$$

for $\alpha \in (0,1)$.

We may then find the inverse cumulative function. For $x \leq 0$ this is just zero, and for $0 < x < 1$, that is $0 < G(x) < \frac{\mathrm{e}}{\alpha+\mathrm{e}}$, we solve $x = \frac{\mathrm{e}}{\alpha+\mathrm{e}}y^{\alpha}$ for y giving $G^{-1}(x) = \left(\frac{\alpha+\mathrm{e}}{\mathrm{e}}x\right)^{1/\alpha}$. Similarly for $x \geq 1$, that is $G(x) \geq 1 - \frac{\alpha}{\alpha+\mathrm{e}} = \frac{\mathrm{e}}{\alpha+\mathrm{e}}$, we solve $x = 1 - \frac{\alpha}{\alpha+\mathrm{e}}\mathrm{e}^{1-y}$ for y, such that

$$G^{-1}(x) = \begin{cases} \left(\frac{\alpha+\mathrm{e}}{\mathrm{e}}x\right)^{1/\alpha} & \text{if } 0 \leq x < \frac{\mathrm{e}}{\alpha+\mathrm{e}}, \\ \ln\left[\frac{\alpha\mathrm{e}}{(1-x)(\alpha+\mathrm{e})}\right] & \text{if } \frac{\mathrm{e}}{\alpha+\mathrm{e}} \leq x \leq 1, \end{cases}$$

for $\alpha \in (0,1)$.

Subsubproblem (b)

Using what we found in (a) we may use the inversion method to sample from $g(x)$ given in Equation (2), as shown in the code block beneath. Figure 2 shows one million samples drawn from `generate_from_gx()` and also the theoretical PDF.

```r
set.seed(69)

generate_from_gx <- function(n, alpha) {
  U <- runif(n)   # Generate n Uniform(0, 1) variables
  bound <- exp(1) / (alpha + exp(1))   # Boundary where G^(-1) changes
  left <- U < bound   # The left of the boundary
  U[left] <- (U[left] / bound)^(1 / alpha)   # Left CDF
  U[!left] <- 1 + log(alpha) - log(1 - U[!left]) - log(alpha + exp(1))   # Right CDF
  return(U)
}

# Sample
n <- 1000000  # One million samples
alpha <- 0.75
gx_samp <- generate_from_gx(n, alpha)   # Generating n samples from g(x)

# The theoretically correct PDF
theo_gx <- function(x, alpha) {
  const <- alpha * exp(1) / (alpha + exp(1))    # Normalizing constant
  func <- rep(0, length(x))   # Vector of zeros of same length as x
  left <- x > 0 & x < 1   # The PDF has one value for 0 < x < 1
  right <- x >= 1         # ... and one value for x >= 1
  func[left] <- const * x[left]^(alpha - 1)   # The value to the left
  func[right] <- const * exp(-x[right])       # The value to the right
  return(func)
}

# Plot
ggplot() +
  geom_histogram(
    data = as.data.frame(gx_samp),
    mapping = aes(x = gx_samp, y = ..density..),
    binwidth = 0.05,
    boundary = 0
  ) +
  stat_function(
    fun = theo_gx,
    args = list(alpha = alpha),
    aes(col = "Theoretical density")
  ) +
  xlim(0, 5) +
  ggtitle("Simulating from g(x) given in Equation (2)") +
  xlab("x") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

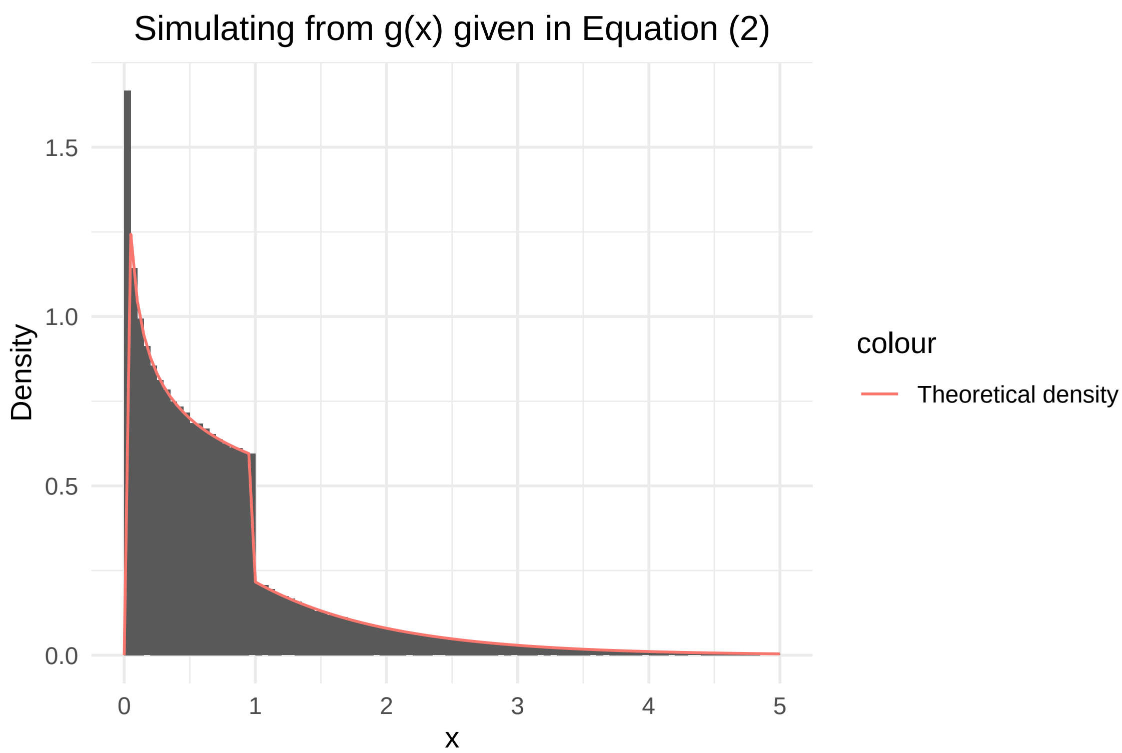

Figure 2: Normalized histogram of one million samples drawn from $g(x)$ given in Equation (2), together with the theoretical PDF, with $\alpha = 0.75$.

Assuming $X \sim g(x)$ we may find the expectation to be

$$\mathrm{E}(X) = \int_{\mathbb{R}} x g(x)\,\mathrm{d}x = \int_0^1 c x^{\alpha}\,\mathrm{d}x + \int_1^{\infty} c x \mathrm{e}^{-x}\,\mathrm{d}x = \frac{\alpha \mathrm{e}}{(\alpha+1)(\alpha+\mathrm{e})} + \frac{2\alpha}{\alpha+\mathrm{e}} \approx 0.768,$$

when $\alpha = 0.75$. This corresponds approximately to the sample mean shown in the following code block.

```
mean(gx_samp)
```

```
## [1] 0.76749
```

Similarly we may find the theoretical variance to be

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 \approx 0.705,$$

also for $\alpha = 0.75$, corresponding approximately to the sample variance given in the code block below.

```
var(gx_samp)
```

```
## [1] 0.703663
```

Subproblem 3.

Subsubproblem (a)

We consider the probability density function

$$f(x) = \frac{c\mathrm{e}^{\alpha x}}{(1+\mathrm{e}^{\alpha x})^2},$$

for $-\infty < x < \infty$ and $\alpha > 0$. To find the normalizing constant c we make sure that the integral over $\mathbb{R}$ of $f(x)$ is one. That is

$$1 = \int_{\mathbb{R}} f(x)\,\mathrm{d}x = c\int_{\mathbb{R}} \frac{\mathrm{e}^{\alpha x}}{(1+\mathrm{e}^{\alpha x})^2}\,\mathrm{d}x,$$

and letting $u = 1 + \mathrm{e}^{\alpha x}$, it follows that

$$1 = \frac{c}{\alpha}\int_1^{\infty} \frac{\mathrm{d}u}{u^2} = \frac{c}{\alpha}\left[-\frac{1}{u}\right]_1^{\infty} = \frac{c}{\alpha}.$$

That is, the normalizing constant is $c = \alpha$, for $\alpha > 0$. We may then write the probability density function as

$$f(x) = \frac{\alpha \mathrm{e}^{\alpha x}}{(1+\mathrm{e}^{\alpha x})^2}, \tag{3}$$

for $-\infty < x < \infty$ and $\alpha > 0$.

Subsubproblem (b)

The cumulative distribution function is given as

$$F(x) = \int_{-\infty}^{x} f(\xi)\,\mathrm{d}\xi = \int_{-\infty}^{x} \frac{\alpha \mathrm{e}^{\alpha\xi}}{(1+\mathrm{e}^{\alpha\xi})^2}\,\mathrm{d}\xi.$$

Again letting $u = 1 + \mathrm{e}^{\alpha\xi}$ it follows that

$$F(x) = \int_1^{1+\mathrm{e}^{\alpha x}} \frac{\alpha \mathrm{e}^{\alpha\xi}}{u^2}\frac{\mathrm{d}u}{\alpha \mathrm{e}^{\alpha\xi}} = \int_1^{1+\mathrm{e}^{\alpha x}} \frac{\mathrm{d}u}{u^2} = \left[\frac{1}{u}\right]_{1+\mathrm{e}^{\alpha x}}^{1} = 1 - \frac{1}{1+\mathrm{e}^{\alpha x}} = \frac{\mathrm{e}^{\alpha x}}{1+\mathrm{e}^{\alpha x}},$$

which holds for $-\infty < x < \infty$ and $\alpha > 0$.

Solving $x = \mathrm{e}^{\alpha y}/(1+\mathrm{e}^{\alpha y})$ for y then gives us the inverse cumulative distribution function. Some algebra then gives that

$$F^{-1}(x) = \frac{1}{\alpha}\ln\left(\frac{x}{1-x}\right) = \frac{1}{\alpha}\left[\ln(x) - \ln(1-x)\right],$$

for $0 < x < 1$ and $\alpha > 0$.

Subsubproblem (c)

Simulating from $f(x)$ given in Equation (3) is shown in the code block below, and the result is shown in Figure 3. The sampling is done by letting $U \sim \text{Uniform}(0,1)$ and using inversion sampling.

```
generate_from_fx <- function(n, alpha) {
  U <- runif(n)   # Generate n Uniform(0, 1) variables
  X <- 1 / alpha * (log(U) - log(1 - U))    # Using the inverse CDF
  return(X)
}

# Sample
n <- 1000000  # One million samples
alpha <- 100  # Letting alpha be 100
fx_samp <- generate_from_fx(n, alpha)   # Generating n samples from f(x)

# The theoretically correct PDF
theo_fx <- function(x, alpha) {
  return(alpha * exp(alpha * x) / (1 + exp(alpha * x))^2)
}
```

```r
# Plot
ggplot() +
  geom_histogram(
    data = as.data.frame(fx_samp),
    mapping = aes(x = fx_samp, y = ..density..),
    binwidth = 0.001,
    boundary = 0
  ) +
  stat_function(
    fun = theo_fx,
    args = list(alpha = alpha),
    aes(col = "Theoretical density")
  ) +
  ggtitle("Simulating from f(x) given in Equation (3)") +
  xlab("x") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

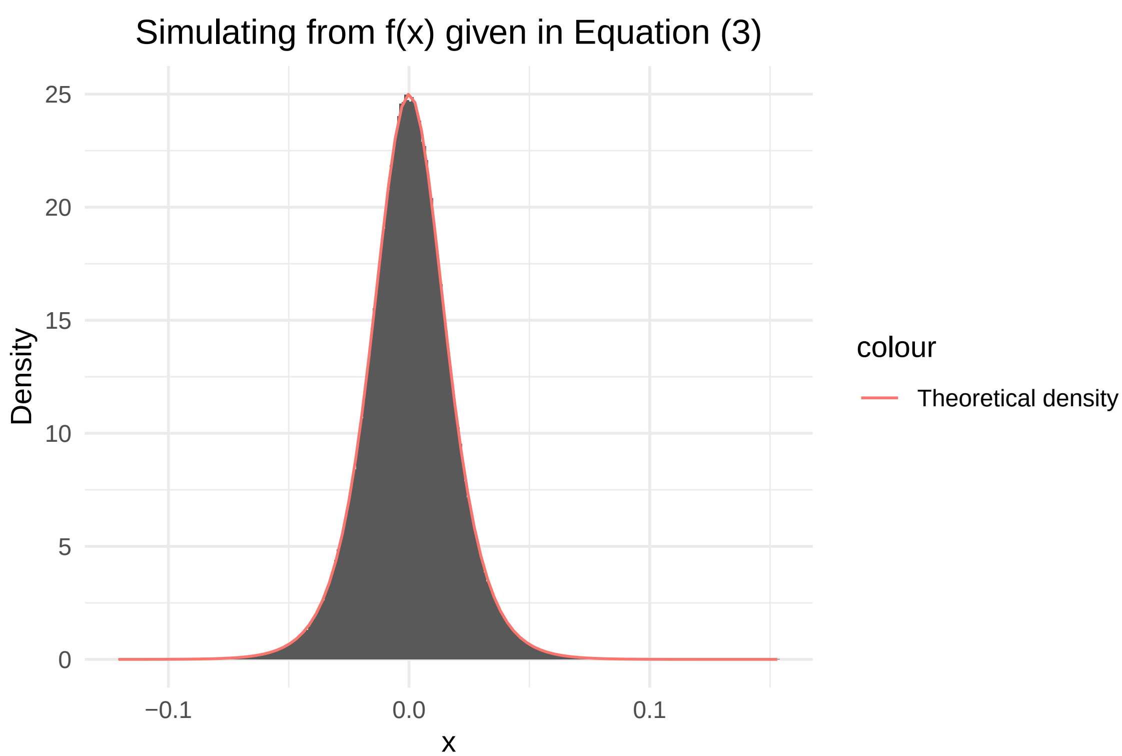

Figure 3: Normalized histogram of one million samples drawn from $f(x)$ given in Equation (3), together with the theoretical PDF, with $\alpha = 100$.

Letting $X \sim f(x)$ we may find the expected value to be

$$\mathrm{E}(X) = \int_{\mathbb{R}} x f(x) \, \mathrm{d}x = \int_{\mathbb{R}} \frac{\alpha x \mathrm{e}^{\alpha x}}{(1 + \mathrm{e}^{\alpha x})^2} \, \mathrm{d}x = 0,$$

because of symmetry. This is confirmed in the following code block where we see that the sample mean is approximately zero, and can also be seen from the figure. This holds for all $\alpha > 0$.

```
mean(fx_samp)
```

```
## [1] 2.957973e-05
```

We may also find the variance of X to be

$$\mathrm{Var}(X) = \mathrm{E}(X^2) = \int_{\mathbb{R}} \frac{\alpha x^2 \mathrm{e}^{\alpha x}}{(1+\mathrm{e}^{\alpha x})^2}\,\mathrm{d}x \approx 0.000329,$$

for $\alpha = 100$. This also corresponds to the sample variance as shown in the code block below.

```
var(fx_samp)
```

```
## [1] 0.0003288562
```

Subproblem 4.

We wish to simulate from a Normal$(0,1)$ distribution using the Box-Muller algoritm. If $X_1 \sim \mathrm{Uniform}(0,\pi)$ and $X_2 \sim \mathrm{Exponential}(1/2)$, then $Y_1 = \sqrt{X_2}\cos(X_1)$ and $Y_2 = \sqrt{X_2}\sin(X_1)$ are standard normal distributed. We use $Z = \sqrt{X_2}\cos(X_1)$ in the following code block. The result of the simulation can be seen in Figure 4, and it also shows the theoretical probability density function.

```
std_normal <- function(n) {
  X1 <- pi * runif(n)   # n samples from Uniform(0, pi)
  X2 <- generate_from_exp(n, 1/2)   # n samples from Exponential(1/2)
  Z <- X2^(1/2) * cos(X1)   # Z ~ Normal(0, 1)
  return(Z)
}

# Sample
n <- 1000000  # One million samples
Box_Muller <- std_normal(n)   # Generating n samples from Normal(0, 1)

# Plot
ggplot() +
  geom_histogram(
    data = as.data.frame(Box_Muller),
    mapping = aes(x = Box_Muller, y = ..density..),
    binwidth = 0.05,
    boundary = 0
  ) +
  stat_function(
    fun = dnorm,
    aes(col = "Theoretical density")
  ) +
  ggtitle("Simulating from standard normal distribution") +
  xlab("z") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

We know that if $Z \sim \mathrm{Normal}(0,1)$, then $\mathrm{E}(Z) = 0$ and $\mathrm{Var}(Z) = 1$, and this corresponds to the approximate sample mean and variance shown in the code block below.

```
mean(Box_Muller)
```

```
## [1] -0.0006589413
```

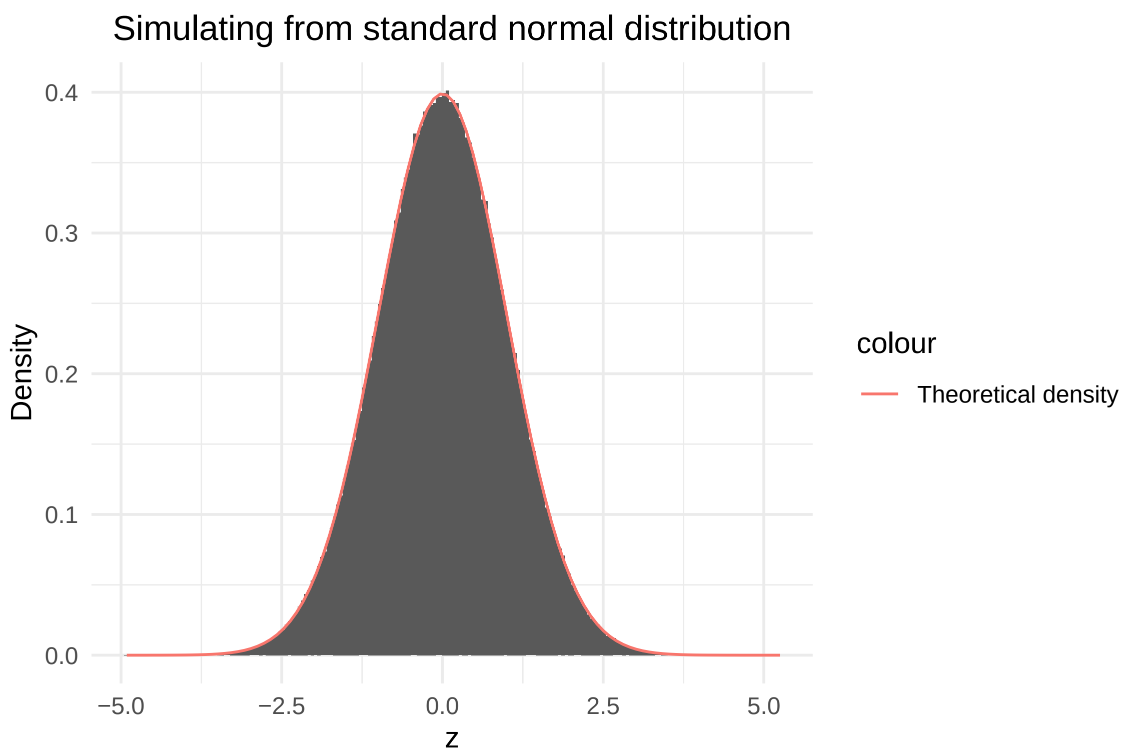

Figure 4: One million samples drawn from a standard normal distribution using the Box-Muller algorithm, together with the theoretical PDF.

```
var(Box_Muller)
```

```
## [1] 1.002119
```

Subproblem 5.

We wish to simulate from a d-variate normal distribution with mean vector μ and covariance matrix Σ. Let $Z \sim \text{Normal}_d(0, I_d)$, where I_d is the identity matrix in $\mathbb{R}^{d \times d}$ and 0 is the zero-vector in $\mathbb{R}^d$. Then

$$X = \mu + AZ \sim \text{Normal}_d(\mu, AA^\top),$$

such that we need to find A such that $\Sigma = AA^\top$, this is done using `chol()` in `R`, and we construct the function in the following code block.

```
d_variate_normal <- function(n, mu, Sigma) {
  d <- length(mu)    # The dimension d is the dimension of mu
  A <- t(chol(Sigma)) # Cholesky decomposition of Sigma. Transpose to get lower triangular
  z <- std_normal(d * n)     # Create vector of d*n independent Normal(0, 1)
  Z <- matrix(z, nrow = d, ncol = n)     # Make z into (d X n) matrix Z
  X <- mu + A %*% Z    # X ~ Normal_d(mu, Sigma)
  return(X)
}
```

We now test the implementation using one million samples in $\mathbb{R}^3$, with

$$\mu = \begin{bmatrix} 1 \\ 7 \\ 2 \end{bmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

We then expect the sample mean and sample covariance matrix to be approximately equal to these. In the following code block we see that this is indeed the case.

```r
# Sample
n <- 1000000    # One million samples
mu <- c(1, 7, 2)   # Create mu
Sigma <- matrix(c(2, -1, 0, -1, 2, -1, 0, -1, 2), nrow = 3)    # Create Sigma
normal <- d_variate_normal(n, mu, Sigma)    # Sample from d-variate Normal(mu, Sigma)

# Test
rowMeans(normal)  # Finding the mean of the rows of normal, sample mean
```

```
## [1] 1.000201 6.999770 1.997882
```

```r
cov(t(normal))    # Transpose because we want with respect to the rows
```

```
##              [,1]       [,2]         [,3]
## [1,]  1.999879520 -0.9984456  0.003097756
## [2,] -0.998445598  1.9978025 -0.998476727
## [3,]  0.003097756 -0.9984767  1.999870756
```

Problem B: The gamma distribution

1.

(a)

(b)

2.

(a)

(b)

3.

(a)

(b)

4.

5.

(a)

(b)

Problem C: Monte Carlo integration and variance reduction

1.

2.

3.

(a)

(b)

Problem D: Rejection sampling and importance sampling

1.

2.

3.

4.