SDS 5531 Homework 1

Gifty Osei

2024-09-11

Remark: If you would like to insert images for your handwritten part into this file, please refer to this article.

Problem 1. Box-Muller transformation

The Box-Muller transformation method simulates random numbers from $N(0,1)$ as follows.

- Step 1: Generate U_1 and U_2 i.i.d. from $U(0,1)$.
- Step 2: Let $X_1 = \sqrt{-2\log U_1} cos(2\pi U_2)$ and $X_2 = \sqrt{-2\log U_1} sin(2\pi U_2)$.

Establish the theoretical validity of the method by proving the following results.

1. (15 points) Use the change-of-variable formula to derive that X_1 and X_2 are two independent draws from $N(0,1)$.

```
knitr::include_graphics(c("D:/WashU/First Year/Sem1/SDS5071_AdvLinearModel/Homework/HW2/Q1.pdf",
    "D:/WashU/First Year/Sem1/SDS5071_AdvLinearModel/Homework/HW2/Q1b.pdf"))
```

1. To prove that X_1 and X_2 are independent draws from $N(0,1)$

We show that

$$f_{X_1,X_2}(x_1,x_2) = f_{X_1}(x_1) \cdot f_{X_2}(x_2)$$

Using transformation, we recall;

$$x_1 = \sqrt{-2\log U_1}\,\cos(2\pi U_2)$$
$$x_2 = \sqrt{-2\log U_1}\,\sin(2\pi U_2)$$

find the

$$(U_1, U_2) \longrightarrow (X_1, X_2)$$

using change of variables;

Let $\underset{\substack{\uparrow \\ \text{radius}}}{R} = \sqrt{-2\log U_1}$ $\qquad \underset{\substack{\uparrow \\ \text{angle}}}{\theta} = 2\pi U_2$

$f_{U_1,U_2}(u_1,u_2) = 1$, for $U_1 \in (0,1)$, $U_2 \in (0,1)$

by the inverse transformation;

$$U_1 = e^{-R^2/2} \qquad U = \frac{\theta}{2\pi}$$

by the bivariate transformation;

$$f_{R,\theta}(r,\theta) = f_{U_1,U_2}(u_1,u_2) \cdot |J|$$

$$|J| = \begin{vmatrix} \frac{\partial x_1}{\partial U_1} & \frac{\partial x_1}{\partial U_2} \\ \frac{\partial x_2}{\partial U_1} & \frac{\partial x_2}{\partial U_2} \end{vmatrix} = \frac{\partial x_1}{\partial U_1} \cdot \frac{\partial x_2}{\partial U_2} - \frac{\partial x_2}{\partial U_1} \cdot \frac{\partial x_1}{\partial U_2}$$

$$f_{R\theta}(r,\theta) = f_R(r) \cdot f_\theta(\theta)$$
$$= \frac{1}{2\pi} e^{-r^2/2}$$

then;

$$f_{X_1X_2}(x_1,x_2) = \frac{1}{2\pi} e^{-(x_1^2+x_2^2)/2}$$
$$= g(x_1) \cdot h(x_2)$$

Hence X_1 and X_2 are independent $\square$.

Figure 1: Part 1

2. (10 points) Show that the polar coordinates $r^2 = X_1^2 + X_2^2 \sim \chi_2^2$, hence $e^{-\frac{r^2}{2}} \sim U(0,1)$, and $\theta = \arctan \frac{X_1}{X_2} \sim U(0, 2\pi)$.

Solution:

```r
knitr::include_graphics(c("D:/WashU/First Year/Sem1/SDS5071_AdvLinearModel/Homework/HW2/Q1c.pdf",
    "D:/WashU/First Year/Sem1/SDS5071_AdvLinearModel/Homework/HW2/Q1d.pdf"))
```

Let $R=\sqrt{-2\log U_1}$ and $\theta = 2\pi U_2$

then; $X_1 = R\cos\theta$ and $X_2 = R\sin\theta$

$X_1, X_2 \sim N(0,1)$; we can show that

$X_1^2 + X_2^2 = r^2$ and

From 1, we show that $X_1 \ X_2 \sim N(0,1)$ then recall that the sum of 2 independent standard normal random variables follows a chi-squared with 2 degrees of freedom so;

$$r^2 = X_1^2 + X_2^2 \sim \chi^2_2$$

Now; $U_1 = e^{-r^2/2}$ then $r^2 = -2\log U_1$ and $U_1 \sim U(0,1)$ so;

$$\therefore P(U_1 \le u) = P[e^{-r^2/2} \le u]$$
$$= P[r^2 \le -2\log u]$$
$$= \int_{-2\log U_1}^{\infty} \frac{1}{2} e^{-x/2}\,dx \quad \text{b/c Pdf of a } \chi^2_2$$

$$P[U_1 \le u] = e^{\frac{-2\log U_1}{2}} = U \quad \square$$

$\therefore f_{U_1}(u) = u$ then it follows that $U_1 \in (0,1)$ hence $U_1 \sim$ Uniform $(0,1)$

Now given ratio $\frac{X_1}{X_2}$, X_1 & $X_2 \sim N(0,1)$

θ is the angle coordinate for an iid standard normal random variables so;

$$f_\theta(\theta) = \frac{1}{2\pi}, \quad \theta \in [0, 2\pi]$$

recall that
$$\frac{X_1}{X_2} = \frac{r\cos\theta}{r\sin\theta}$$
$$\frac{X_1}{X_2} = \tan\theta$$
$$\therefore \theta = \arctan\left(\frac{X_1}{X_2}\right)$$

Figure 2: Part 1

Problem 2. Generate Cauchy random numbers

The Cauchy distribution is Student's t-distribution with 1 degree of freedom.

1. (10 points) Derive an algorithm to simulate random numbers from the Cauchy distribution using the inverse cdf approach. (Hint: Show the Cauchy cdf is $F(x) = \tan^{-1}(x)/\pi$.)

Solution:

```r
knitr::include_graphics("D:/WashU/First Year/Sem1/SDS5071_AdvLinearModel/Homework/HW2/Q2a.pdf")
```

Problem 2.

1. $F_X(x) = \int_{-\infty}^{x} f(t)\,dt$, recall that $f(t) = \frac{1}{\pi(1+t^2)}$

$= \int_{-\infty}^{x} \frac{1}{\pi(1+t^2)}\,dt$

$\therefore F(x) = \frac{1}{\pi}\tan^{-1}(x) + \frac{1}{2}$ b/c $\int \frac{1}{1+t^2} = \tan^{-1}(t)$

Using the inverse cdf metho;

Let $U = F(x)$; $U = \frac{1}{2} + \frac{\tan^{-1}(x)}{\pi}$

$U - \frac{1}{2} = \frac{\tan^{-1}(x)}{\pi}$ making x the subject

$x = \tan\left[\pi\left(U - \frac{1}{2}\right)\right]$

$\therefore$ Algorithm ;

- Generate a random number U from uniform $(0,1)$
- Compute and Simulate the Cauchy random numbers from the inverse cdf as ;

$x = \tan\left[\pi\left(\underset{\uparrow\ U(0,1)}{U} - \frac{1}{2}\right)\right]$

This is a fast approach!.

Figure 3: Problem 2

2. (10 points) Alternatively, one can simulate from the Cauchy distribution by computing the ratio $\frac{X_1}{X_2}$, where X_1 and X_2 are two independent $N(0,1)$ random variables. Explain why this works.

Solution:

3. (20 points) Implement these two methods in R (or Python). Then compare their computing time and efficiency by simulating n Cauchy random numbers. (Choose n to be reasonably large to be able to

tell the time difference in running the two methods. The exact choice of n depends on your hardware and implementation.)

Solution:

```
# 1. Implement the two methods. (Make sure your implementation is efficient.
# For example, avoid loops if possible.)

# Part 1 Using the Inverse CDF to generate Cauchy random numbers
inv_cdf_cauc <- function(n) {
    # Generate n uniform random numbers between 0 and 1
    u <- runif(n)
    # Use the inverse CDF of the Cauchy distribution to simulate x
    x <- tan(pi * (u - 0.5))
    return(x)  # return x
}


# 2. Simulate n Cauchy random numbers and compare the execution time of the two
# methods. (You can find the execution time using the R function system.time().
# )

# Part 2

# Ratio of 2 standard Normals to generate Cauchy RV.
ratio_of_norms_cauc <- function(n) {
    # Generate n standard normal random numbers as x1
    X1 <- rnorm(n)

    # Generate n standard normal random numbers
    X2 <- rnorm(n)
    # Take the ratio of the two
    x <- X1/X2

    return(x)  # return x
}


# Part 3

## Comparing time to compute
library(microbenchmark)

# set a reasonable n
n <- 10000

# Compare the two methods using micro benchmark

benchmark_results <- microbenchmark(inverse_cdf = inv_cdf_cauc(n), ratio_of_stan_normals = ratio_of_norr
    times = 10  # Run each method 10 times
)
```

```r
## Summarize
kable(summary(benchmark_results), caption = "Time to Compute", latex_options = c("hold_position",
    "striped"))
```

Table 1: Time to Compute

expr	min	lq	mean	median	uq	max	neval
inverse_cdf	207.8	212.1	392.67	214.30	216.3	1967.4	10
ratio_of_stan_normals	661.8	664.0	879.98	676.15	728.8	2627.1	10

```r
# print(benchmark_results)

## We can see that inverse CDF is faster
```

Table 1: From the descriptive summary, we can see that inverse cdf is doing much better in time to compute and generate the cauchy random variables

```r
# 3. For both methods, draw the empirical cdf of your simulated numbers and see how close it is to the
## Plotting All 3 plots (Cauchy, Ratio, Inverse CDF)

## Empirical Values from inverse cdf and ratio

#inverse_cdf = inv_cdf_cauc(10000)

#ratio_of_stan_normals = ratio_of_norms_cauc(10000)

cauchy_samples <- list(
  inverse = inv_cdf_cauc(10000),
  ratio = ratio_of_norms_cauc(10000)
)


x_values <- seq(-5, 5, length.out = 10000)

# Calculate the true Cauchy CDF values
true_cdf <- pcauchy(x_values)


## Create Data

empirical_cdfs <- map(cauchy_samples, function(samples) {
  data.frame(
    x = sort(samples),
    cdf = seq(1/n, 1, length.out = n)
  )
})

data_true <- data.frame(x = x_values, cdf = true_cdf)

#data_inverse <- data.frame(x = empirical_cdfs$inverse, cdf = seq(1/n, 1, length.out = n))
```

```r
#data_ratio <- data.frame(x = empirical_cdfs$ratio, cdf = seq(1/n, 1, length.out = n))


ggplot() +
  # True Cauchy CDF
  geom_line(data = data_true, aes(x = x, y = cdf), color = "blue", size = 1, linetype = "solid", alpha =
  # CDF from inverse CDF method
  geom_line(data = empirical_cdfs$inverse , aes(x = x, y = cdf), color = "red", size = 1, linetype = "da
  # CDF from ratio of normals method
  geom_line(data =  empirical_cdfs$ratio, aes(x = x, y = cdf), color = "green", size = 1, linetype = "do
  xlim(-10, 10) +

    ggtitle("Comparison of Simulated Cauchy Distribution CDFs")+
    xlab("x")+
    ylab("CDF")+
    labs(caption = "Blue: True Cauchy CDF,
         Red: Inverse CDF Method, Green: Ratio of Normals Method"
  ) + theme_bw()
```

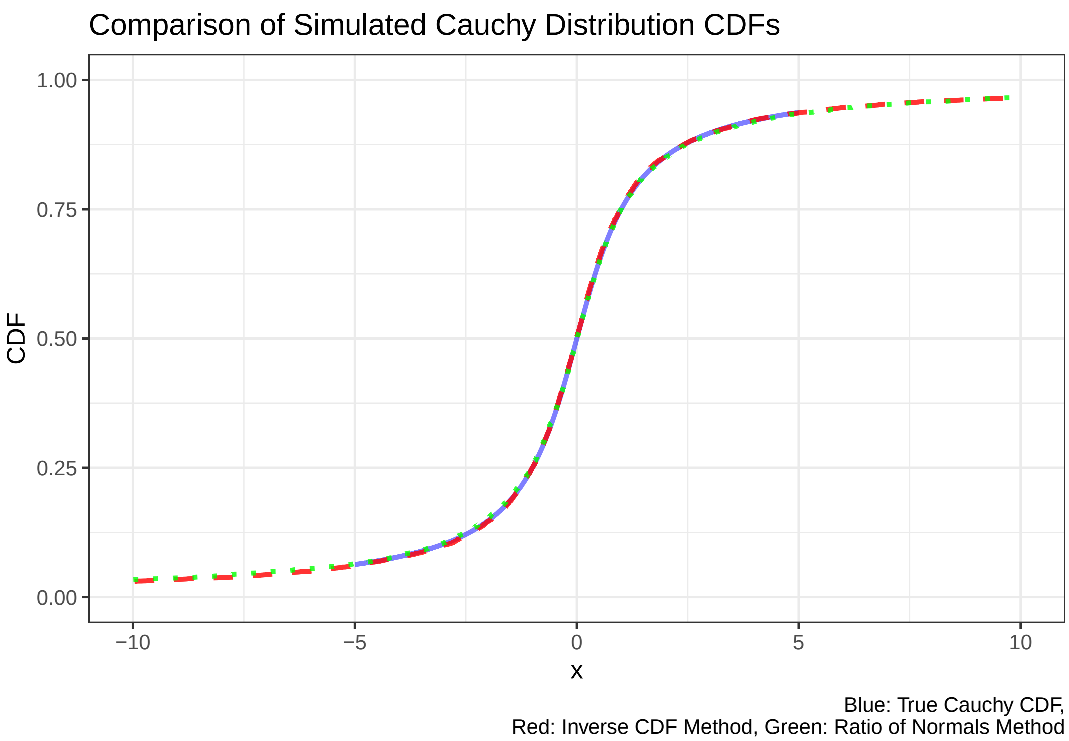

Figure 4: Plot showing the true, inverse cdf and ratio of normals approach of simulating cauchy random number

Problem 3. Accept-Reject sampling

Consider simulating from $N(0,1)$ using the accept-reject sampling. Pretend you do not know the normalizing constant of the pdf, so $f(x) = e^{-\frac{x^2}{2}}$. First, consider using the standard Cauchy distribution as an envelope distribution. Let $g(x) = \frac{1}{1+x^2}$. (Note we have dropped the normalizing constant in the Cauchy pdf.)

1. (10 points) Show that the ratio
$$\frac{f(x)}{g(x)} = (1+x^2)e^{-\frac{x^2}{2}} \leq \frac{2}{\sqrt{e}},$$
with the equality attained at $x = \pm 1$.

Solution:

2. (15 points) Show that the probability of acceptance is $\sqrt{\frac{e}{2\pi}} \approx 0.66$. Also run an empirical evaluation of the probability of acceptance.

Solution:

```r
# 1. implement your algorithm and record the number of acceptances or
# rejections 2. Simulate n=1000 random numbers, and find the empirical
# proportion of acceptances or rejections

set.seed(300)

# Number of simulations
n <- 10000

# Generate n samples from the standard Cauchy distribution using vectorized
# operations
x <- rcauchy(n)

# target (unnormalized standard normal)
f_x <- exp(-x^2/2)

# envelope density (unnormalized standard Cauchy)
g_x <- 1/(1 + x^2)

# ratio M
M <- 2/sqrt(exp(1))

# Acceptance probability
acceptance_prob <- f_x/g_x

# simulate uniform random numbers for comparison
u <- runif(n)

# Accept if u is less or equal to acceptance_prob / M

accepted <- u <= (acceptance_prob/M)

# empirical proportion of acceptances
```

```r
acceptance_rate <- mean(accepted)

# Print results cat('Empirical proportion of acceptances:', acceptance_rate,
# '\n') cat('Empirical proportion of rejections:', 1 - acceptance_rate, '\n')
```

The Empirical proportion of acceptance is 0.66

3. (10 points) Now, consider using a scaled Cauchy distribution as the envelope distribution, i.e. $g_\sigma(x) = \frac{1}{\pi\sigma(1+\frac{x^2}{\sigma^2})}$. Find the upper bound for $\frac{f(x)}{g_\sigma(x)}$ and the value of σ that minimizes this bound.

Solution:

Including Images

Including PDF