Lab Report: Lab4 - Computational Statistics

Dhanush Kumar Reddy Narayana Reddy (dhana004), Udaya Shanker Mohanan Nair (udamo524)

2025-02-18

Introduction

Implementation of 2 Assignment questions of Computational Statistics Lab 4 .

Contributions

Member: Udaya Shanker Mohanan Nair, Liu Id: udamo524, Contribution: Report writing and coding of question 1. Member: Dhanush Kumar Reddy Narayana Reddy, Liu Id: dhana004, Contribution: Report writing and coding of question 2.

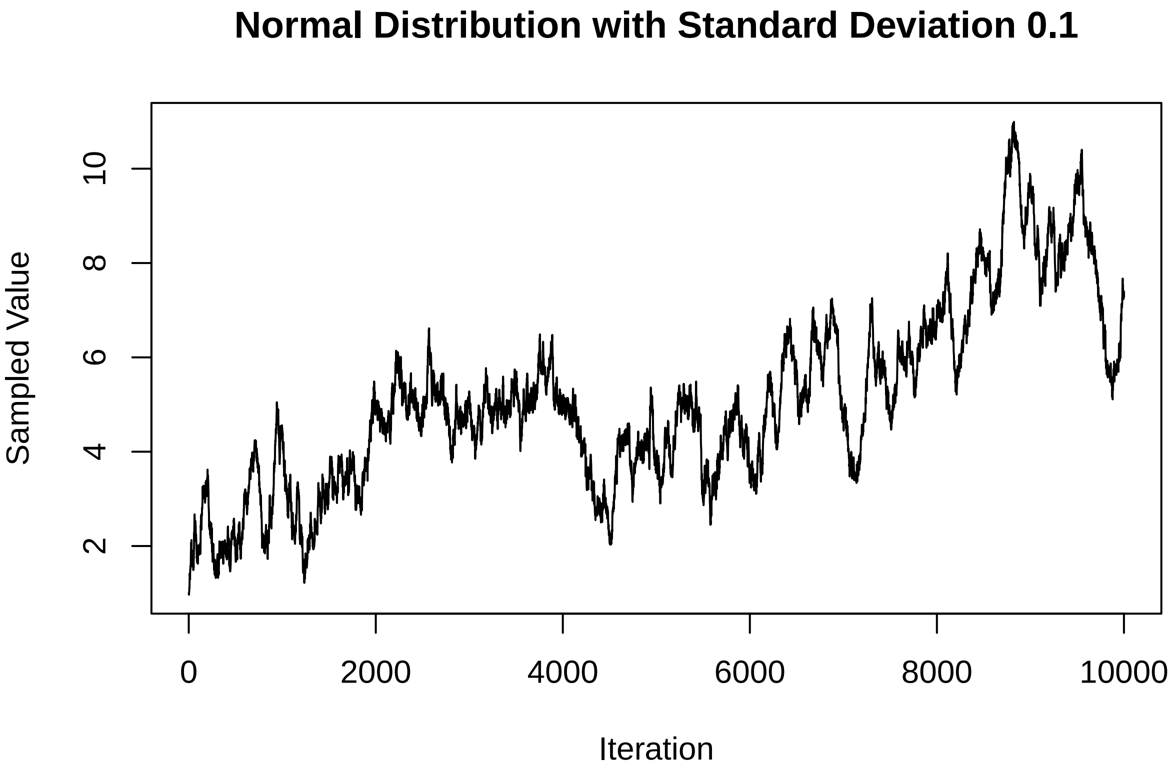
Question 1

Given below a target distribution,

$$f(x) = 120x^5e^{-x}, \quad x > 0.$$

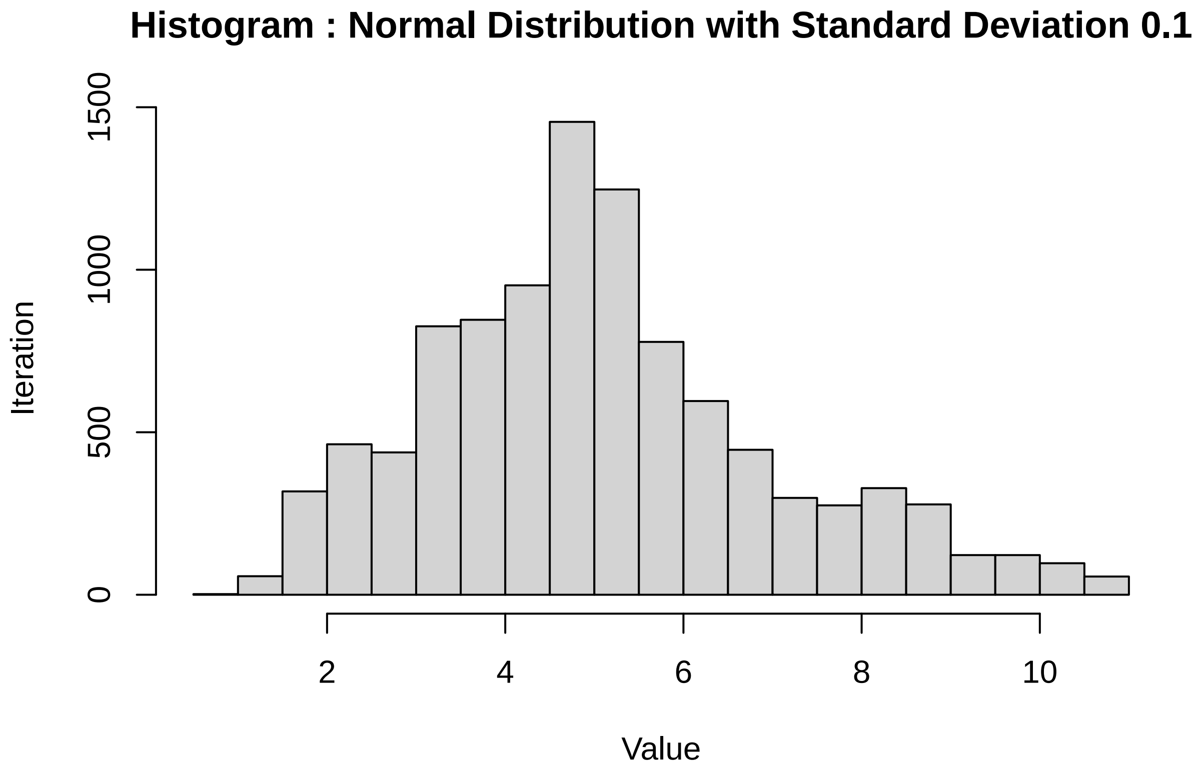
Part A

In this part we are asked to use Metropolis-Hastings Algorithm to generate 10000 samples using a normal distribution.



Convergence :Values appears to have a upward motion initially and then stabilizes after few iterations. After 3000-4000 iterations, the values oscillates around a stable range(4-10), which indicates that it is more likely converged to that taget distribution.

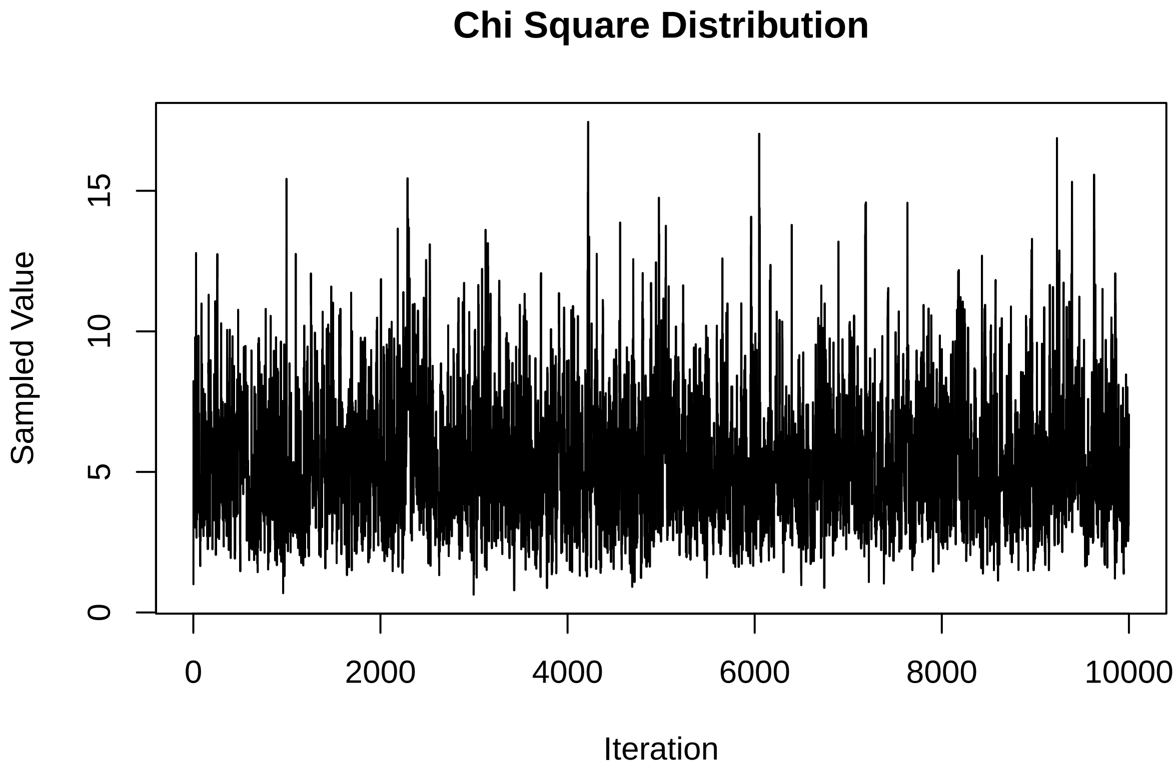
Burn-in Period : Initial few iterations up-to around 3000-4000 shows variations in values,which is considered to be burn-in period after which values reached a stable distribution.

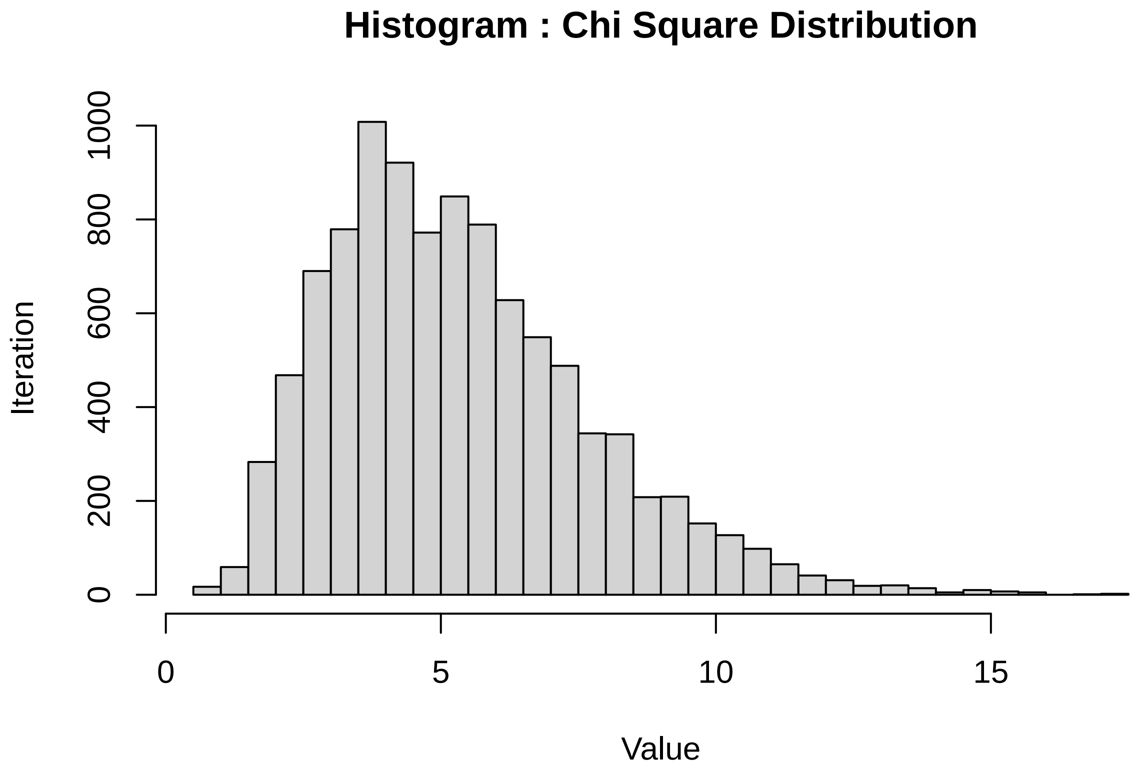


```
## Acceptance Rate of the First Distribution Sample : 9859
```

Part B

In this part we are asked to use Metropolis-Hastings Algorithm to generate 10000 samples using a chi square distribution.

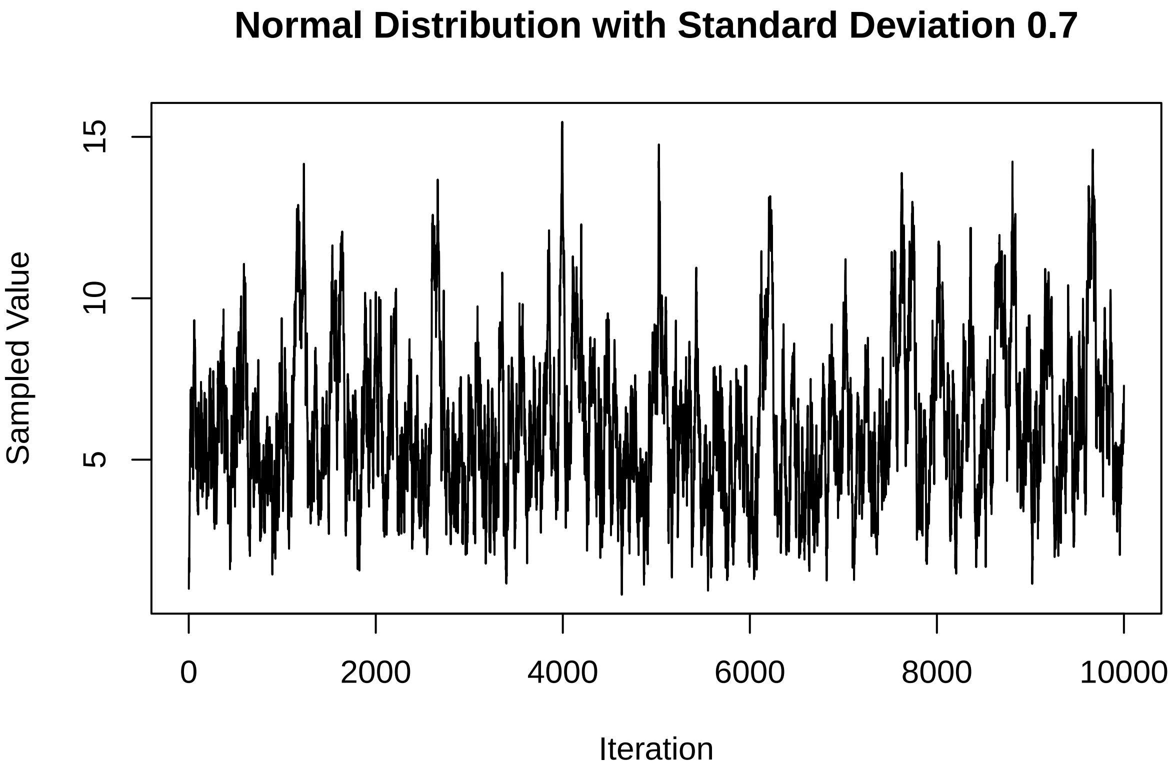

Chi Square Distribution
Sampled Value
0
5
10
15
0
2000
4000
6000
8000
10000
Iteration

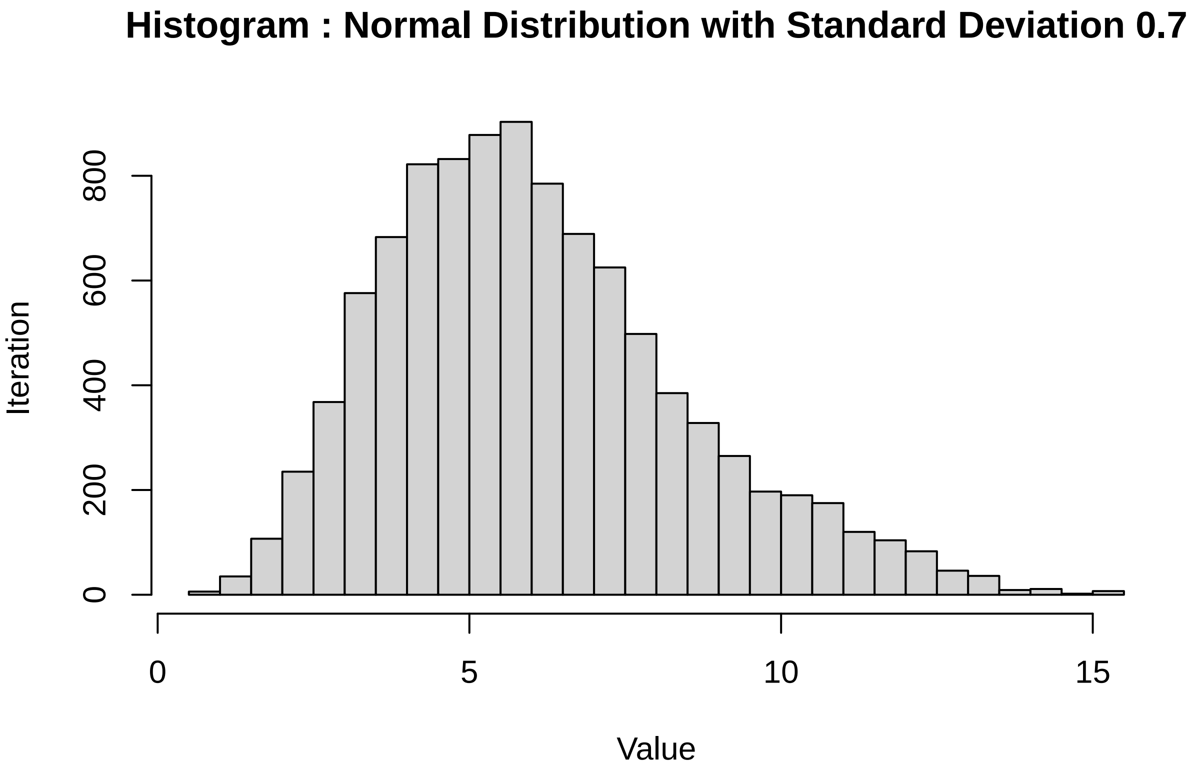


```
## Acceptance Rate: 6085
```

Part C

In this part we choose to use Metropolis-Hastings Algorithm to generate 10000 samples using a normal distribution with another Standard Deviation different from part A. Here in this case we opted 0.4 in for Standard Deviation.


Normal Distribution with Standard Deviation 0.7
Sampled Value
5
10
15
0
2000
4000
6000
8000
10000
Iteration



```
## Acceptance Rate: 9039
```

Part D

Report of the above three generated distributions

Distribution	Acceptance_Rate	E_X
Normal (SD = 0.1)	9859	5.078
Chi-Square (df = floor(X_t) + 1)	6085	5.345
Normal (SD = 0.7)	9039	6.072

Among all the above three distributions, the Normal Distribution with standard deviation 0.7 gives a better option as it is having almost equivalent acceptance rate in comparison with normal distribution(Standard Deviation 0.1) and greatest E(X) of all the three.

Part E

In this part we calculated the E(X) of all the three distributions,

```
## First Dirtibution- Normal Distribution with Standard Deviation 0.1 =  5.077624
```

```
## Second Dirtibution- Chi- Square Distribution =  5.345107
```

```
## First Dirtibution- Normal Distribution with Standard Deviation 0.7 =  6.072416
```

Part F

From this given PDF it is understood that it is a gamma distribution

$$f(x) = \frac{\lambda^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\lambda x}, \quad x \in (0, \infty)$$

From the above given standard format,

$$f(x) = 120x^5 e^{-x}, \quad x > 0.$$

is a gamma distribution with $\alpha = 6$ (shape) and $\lambda = 1$ (rate).

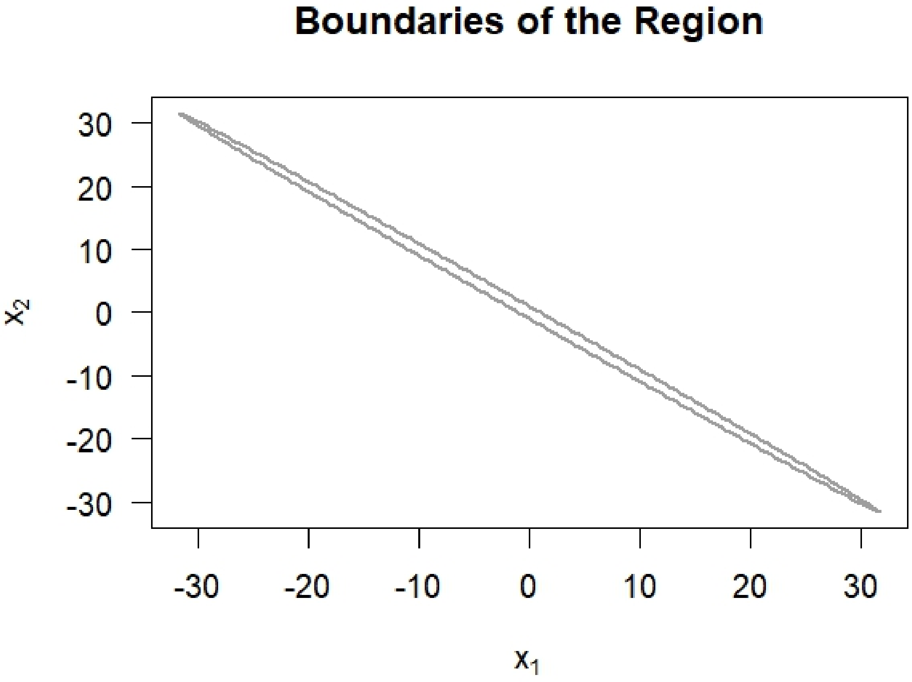
Mean of gamma distribution, $E(x) = \frac{\alpha}{\lambda} = \frac{6}{1} = 6$

Distribution	Acceptance_Rate	E_X	theoretical E(X)
Normal (SD = 0.1)	9859	5.078	6
Chi-Square (df = floor(X_t) + 1)	6085	5.345	6
Normal (SD = 0.7)	9039	6.072	6

Question 2

$$f(x_1, x_2) \propto 1\{x_1^2 + wx_1x_2 + x_2^2 < 1\}$$

Part A



Part B

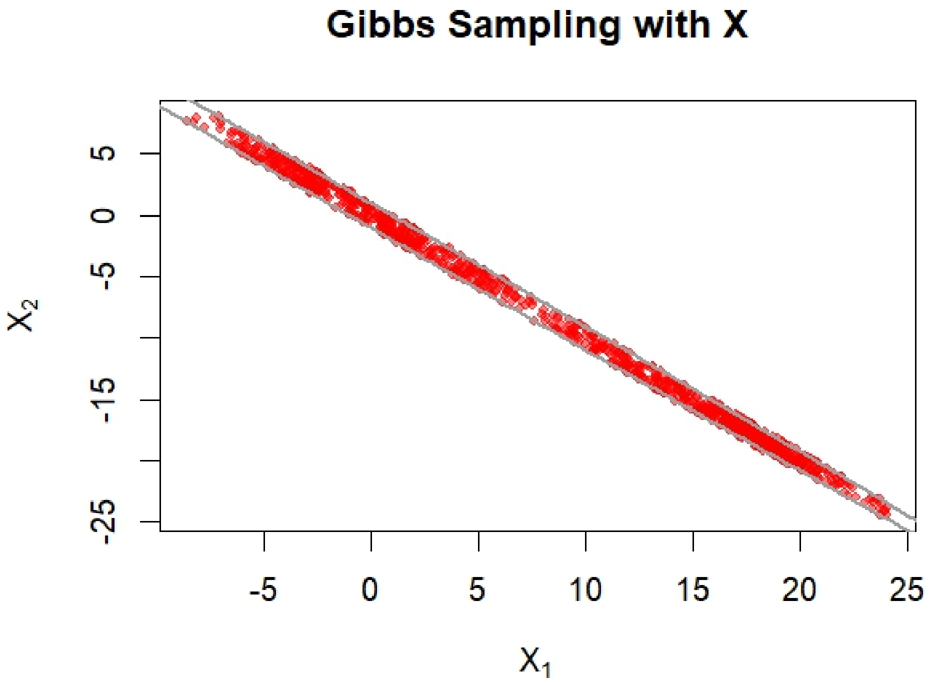
Conditional distribution of X1 given X2 = x2 The range of X1 is:

$$X_1^2 + wX_1x_2 + x_2^2 < 1$$

Conditional distribution of X1 given X2 = x1 The range of X2 is:

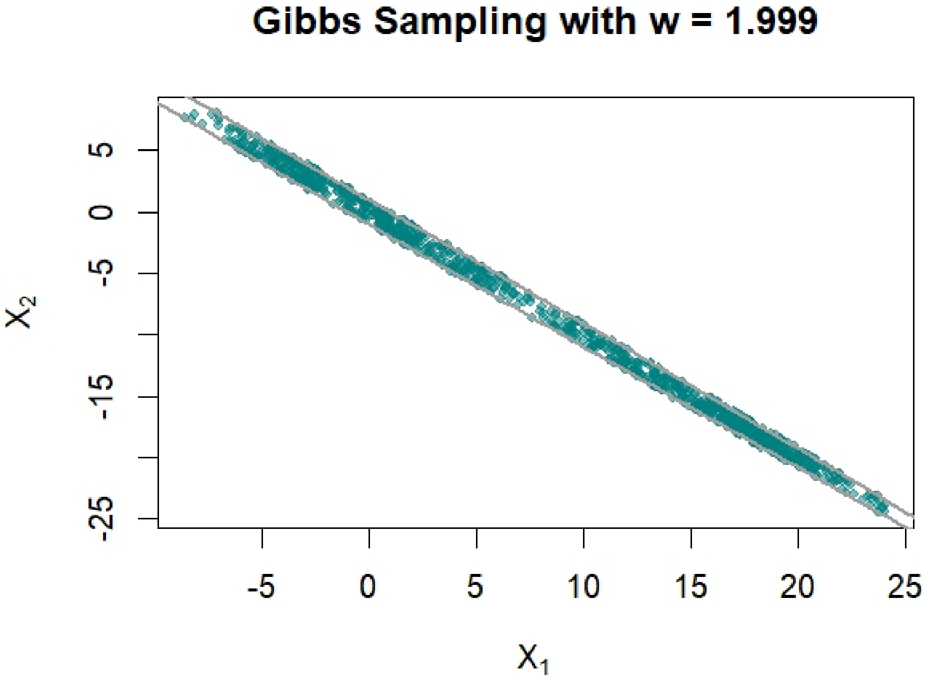
$$x_1^2 + wx_1X_2 + X_2^2 < 1$$

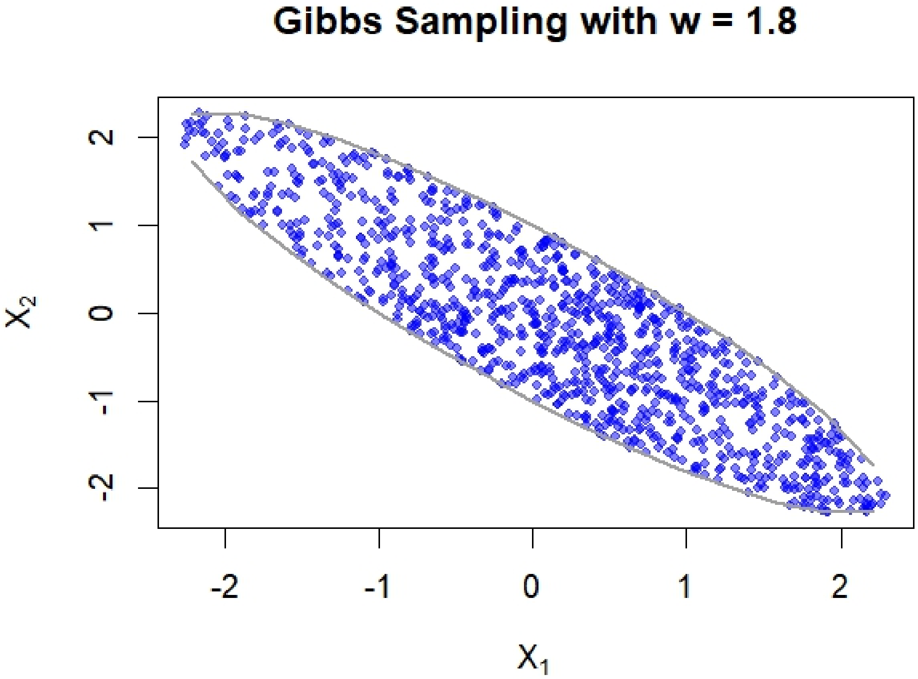
Part C



By running Gibbs sampling for n = 1000 random vectors, estimated P(X1 > 0) that is approximately 0.755. The true result for this probability, given the symmetry of the distribution, should be 0.5.

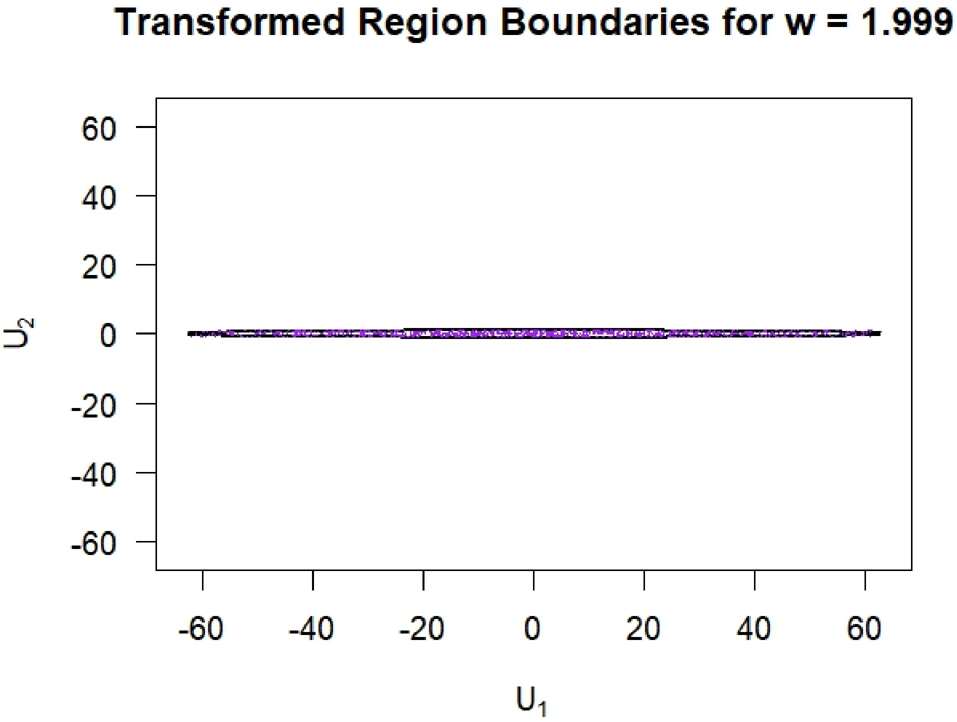
Part D





For w = 1.999, the elliptical region becomes highly elongated, resulting in a strong correlation between X1 and X2. As a consequence, the Gibbs sampler experiences slow mixing, as it takes longer to explore the entire region due to the narrow conditional distributions. Additionally, the sampler exhibits high autocorrelation, meaning successive samples are highly dependent, which reduces the effective sample size. In contrast, when w = 1.8, the region is less elongated, allowing the sampler to mix more efficiently and explore the distribution more effectively.

Part E



By transforming the variables and using Gibbs sampling for U = (U1,U2), estimated that P(X1 > 0) = P((U2 + U1)/2 > 0)is approximately 0.59.

Comparison:

(1) The two estimates, 0.755 and 0.491, are quite different. When w = 1.999,the elliptical region becomes highly elongated, resulting in a strong correlation between X1 and X2. This strong correlation leads to slow mixing in the Gibbs sampler, meaning the chain takes longer to explore the entire region. Consequently, the sampler may get “stuck” in certain areas, causing biased estimates. The high correlation between X1 and X2 results in the sampler underestimating or overestimating certain probabilities, such as P(X1>0).

(2) Transforming the variables to U = (U1,U2) =(X1 - X2 ,X1 + X2) changes the geometry of the region. The transformed region is less elongated, and the variables U1 and U2 are less correlated. This improved geometry enhances the mixing of the Gibbs sampler, allowing it to explore the region more efficiently. As a result, the estimate of P(X1>0) from the transformed Gibbs sampling is likely more accurate.

(3) The estimate from the transformed Gibbs sampling (0.491) is more reliable because the sampler mixes better and explores the region more thoroughly. In contrast, the estimate from the original Gibbs sampling (0.755) is likely biased due to the slow mixing caused by the high correlation between X1 and X2.

(4) Since the distribution of X is uniform over the elliptical region, the true value of P(X1>0) should be 0.5. This is because the region is symmetric about X1 = 0 when w = 1.999. The transformed Gibbs sampling estimate (0.491) is closer to the true value (0.5) compared to the original Gibbs sampling estimate (0.755).

Appendix

Question 1

```r
fx <- function(x){
  if(x>0){
    res <- 120 * x^5 * exp(-x)
  }
  else{
    res <- 0
  }
  return(res)
}
set.seed(101)
n <- 10000

first_distribution <- numeric(n)
#start value is set to 1
first_distribution[1] <- 1
first_accepted_rate <- 0
for (i in 2:n) {
  value <- rnorm(1, mean=first_distribution[i-1], sd=0.1)
  check_value <- fx(value)/fx(first_distribution[i-1])
  if (runif(1) < check_value) {
    first_distribution[i] <- value
    first_accepted_rate <- first_accepted_rate + 1
  } else {
    first_distribution[i] <- first_distribution[i-1]
  }
}
compare_df_a <- data.frame(
  Distribution = "Normal (SD = 0.1)",
  Acceptance_Rate = first_accepted_rate,
  E_X = mean(first_distribution)
)

plot(first_distribution, type='l', main='Normal Distribution with Standard Deviation 0.1',
     xlab='Iteration', ylab='Sampled Value')
hist(first_distribution, breaks=30, main='Histogram : Normal Distribution with Standard Deviation 0.1',
     xlab='Value', ylab='Iteration')


cat("Acceptance Rate of the First Distribution Sample :", first_accepted_rate)

second_distribution <- numeric(n)
second_distribution[1] <- 1
```

```r
second_acceptance_rate <- 0
for (i in 2:n) {
  value <- rchisq(1,df=floor(second_distribution[i-1])+1)
  check_value <- fx(value)/fx(second_distribution[i-1])
  if (runif(1) < check_value) {
    second_distribution[i] <- value
    second_acceptance_rate <- second_acceptance_rate + 1
  } else {
    second_distribution[i] <- second_distribution[i-1]
  }
}
compare_df_b <- data.frame(
  Distribution = "Chi-Square (df = floor(X_t) + 1)",
  Acceptance_Rate = second_acceptance_rate,
  E_X = mean(second_distribution)
)
plot(second_distribution, type='l', main='Chi Square Distribution',
     xlab='Iteration', ylab='Sampled Value')

hist(second_distribution, breaks=30, main='Histogram : Chi Square Distribution',
     xlab='Value', ylab='Iteration')


cat("Acceptance Rate:", second_acceptance_rate)

third_distribution <- numeric(n)
#start value is set to 1
third_distribution[1] <- 1
third_accepted_rate <- 0
for (i in 2:n) {
  value <- rnorm(1, mean=third_distribution[i-1], sd=0.7)
  check_value <- fx(value)/fx(third_distribution[i-1])
  if (runif(1) < check_value) {
    third_distribution[i] <- value
    third_accepted_rate <- third_accepted_rate + 1
  } else {
    third_distribution[i] <- third_distribution[i-1]
  }
}
compare_df_c <- data.frame(
  Distribution = "Normal (SD = 0.7)",
  Acceptance_Rate = third_accepted_rate,
  E_X = mean(third_distribution)
)

plot(third_distribution, type='l', main='Normal Distribution with Standard Deviation 0.7',
     xlab='Iteration', ylab='Sampled Value')

hist(third_distribution, breaks=30, main='Histogram : Normal Distribution with Standard Deviation 0.7',
     xlab='Value', ylab='Iteration')

cat("Acceptance Rate:", third_accepted_rate)
```

```
compare_df <- rbind(compare_df_a,compare_df_b,compare_df_c)
library(pander)
pander(compare_df)

cat("First Dirtibution- Normal Distribution with Standard Deviation 0.1 = ",mean(first_distribution))
cat("Second Dirtibution- Chi- Square Distribution = ",mean(second_distribution))
cat("First Dirtibution- Normal Distribution with Standard Deviation 0.7 = ",mean(third_distribution))

compare_df$`theoretical E(X)` <- 6

pander(compare_df)
```

Question 2

```
# Question 2: Gibbs Sampling
# Part A
w  <- 1.999

# a range of x1-values
xv <- seq(-1, 1, by=0.01) * 1/sqrt(1-w^2/4)

# plot
plot(xv, xv, type="n", xlab=expression(x[1]), ylab=expression(x[2]),
     las=1, main = "Boundaries of the Region")

# ellipse
lines(xv, -(w/2)*xv-sqrt(1-(1-w^2/4)*xv^2), lwd=2, col=8)
lines(xv, -(w/2)*xv+sqrt(1-(1-w^2/4)*xv^2), lwd=2, col=8)

# Part B
# Conditional distribution of X1 given X2 = x2
# The range of X1 is:
#
#   $$
#   X_1^2 + w X_1 x_2 + x_2^2 < 1\
# $$
#
#   Conditional distribution of X1 given X2 = x1
# The range of X2 is:
#
#   $$
#   x_1^2 + w x_1 X_2 + X_2^2 < 1\
# $$

# Part C
# Gibbs sampling function
gibbs_sampling <- function(n, w) {
  X1 <- 0
  X2 <- 0
  samples <- matrix(0, nrow = n, ncol = 2)
  for (i in 1:n) {
    X1_range <- sqrt(1 - X2^2 + (w^2 * X2^2) / 4)
    X1 <- runif(1, -X1_range - (w * X2 / 2), X1_range - (w * X2 / 2))
    X2_range <- sqrt(1 - X1^2 + (w^2 * X1^2) / 4)
    X2 <- runif(1, -X2_range - (w * X1 / 2), X2_range - (w * X1 / 2))
    samples[i, ] <- c(X1, X2)
  }
  return(samples)
}

set.seed(123)
n <- 1000
w <- 1.999
samples <- gibbs_sampling(n, w)

# Plot
plot(samples[, 1], samples[, 2], pch = 20, col = rgb(1, 0, 0, 0.5),
     xlab = expression(X[1]), ylab = expression(X[2]),
     main = "Gibbs Sampling with X")
lines(xv, -(w / 2) * xv - sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)
lines(xv, -(w / 2) * xv + sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)

# P(X1 > 0)
p_X1 <- mean(samples[, 1] > 0)
cat("Estimated P(X1 > 0) is:", p_X1, "\n")

# Part D
set.seed(123)
n <- 1000

# w = 1.999
w <- 1.999
samples <- gibbs_sampling(n, w)

# Plot
plot(samples[, 1], samples[, 2], pch = 20, col = rgb(0, 0.5, 0.5, 0.5),
     xlab = expression(X[1]), ylab = expression(X[2]),
     main = "Gibbs Sampling with w = 1.999")
lines(xv, -(w / 2) * xv - sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)
lines(xv, -(w / 2) * xv + sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)

# w = 1.8
w <- 1.8
samples_w <- gibbs_sampling(n, w)

# Plot
plot(samples_w[, 1], samples_w[, 2], pch = 20, col = rgb(0, 0, 1, 0.5),
     xlab = expression(X[1]), ylab = expression(X[2]),
     main = "Gibbs Sampling with w = 1.8")
lines(xv, -(w / 2) * xv - sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)
lines(xv, -(w / 2) * xv + sqrt(1 - (1 - w^2 / 4) * xv^2), lwd = 2, col = 8)

# For w = 1.999, the elliptical region becomes highly elongated, resulting in a strong correlation betw
```

```r
# Part E
w <- 1.999
n <- 1000

# a range of U
uv <- seq(-1, 1, by=0.01) * 2/sqrt(2-w)

# Plot
plot(uv, uv, type="n", xlab=expression(U[1]), ylab=expression(U[2]), las=1,
     main = "Transformed Region Boundaries for w = 1.999")
lines(uv, -sqrt((4-(2-w)*uv^2)/(2+w)), col="black", lwd=2)
lines(uv, sqrt((4-(2-w)*uv^2)/(2+w)), col="black", lwd=2)

# Gibbs sampling for U
u <- matrix(0, nrow = n, ncol = 2)
u[1, ] <- c(0, 0)
for (i in 2:n) {
  u_1 <- u[i, 1]
  lower_u_2 <- -sqrt((4-(2-w)*u_1^2)/(2+w))
  upper_u_2 <- sqrt((4-(2-w)*u_1^2)/(2+w))
  u[i, 2] <- runif(1, lower_u_2, upper_u_2)
  u_2 <- u[i-1, 2]
  lower_u_1 <- -sqrt((4-(2+w)*u_2^2)/(2-w))
  upper_u_1 <- sqrt((4-(2+w)*u_2^2)/(2-w))
  u[i, 1] <- runif(1, lower_u_1, upper_u_1)
}
# Plot for U
points(u[, 1], u[, 2], pch = ".", col = "purple")

# Estimate P(X1 > 0) = P((u_2 + u_1)/2 > 0)
p_X1_transformed <- mean(u[, 2] > -u[, 1])
cat("Estimated P(X1 > 0) = P((u_2 + u_1)/2 > 0) is:", p_X1_transformed, "\n")

# Comparison:
#
# The two estimates, 0.755 and 0.491, are quite different. When w = 1.999,the elliptical region becomes
# Transforming the variables to U = (u_1,u_2) =(X1 - X2 ,X1 + X2) changes the geometry of the region. T
# The estimate from the transformed Gibbs sampling (0.491) is more reliable because the sampler mixes b
# Since the distribution of X is uniform over the elliptical region, the true value of P(X1>0) should b
```