Mth422A RMDfile

Vijay Soren

2024-02-29

3

```
set.seed(1)
n = 15
m = 10
lambda.1 <- 2
lambda.2 <- 2.5
a <- 0.1
b <- 0.1
X <- rpois(n,lambda = lambda.1)
Y <- rpois(m, lambda = lambda.2)

lambda.1.samples <- rgamma(1e4, shape = sum(X) + a, rate = m+b)
lambda.2.samples <- rgamma(1e4, shape = sum(Y) + a, rate = n + b)
theta.samples <- lambda.1.samples/(lambda.1.samples + lambda.2.samples)
#  95% HPD CI based on Monte Carlo samples
opt.x.s <- optimize(function(x){
  l <- quantile(theta.samples, c(x,x + 0.95))
  return(l[2]-l[1])
}, lower = 0,upper = 0.05)$minimum


posCI95.HPD.s <- quantile(theta.samples, c(opt.x.s,opt.x.s + 0.95))

print("posCI95.HPD.s")
```

```
## [1] "posCI95.HPD.s"
```

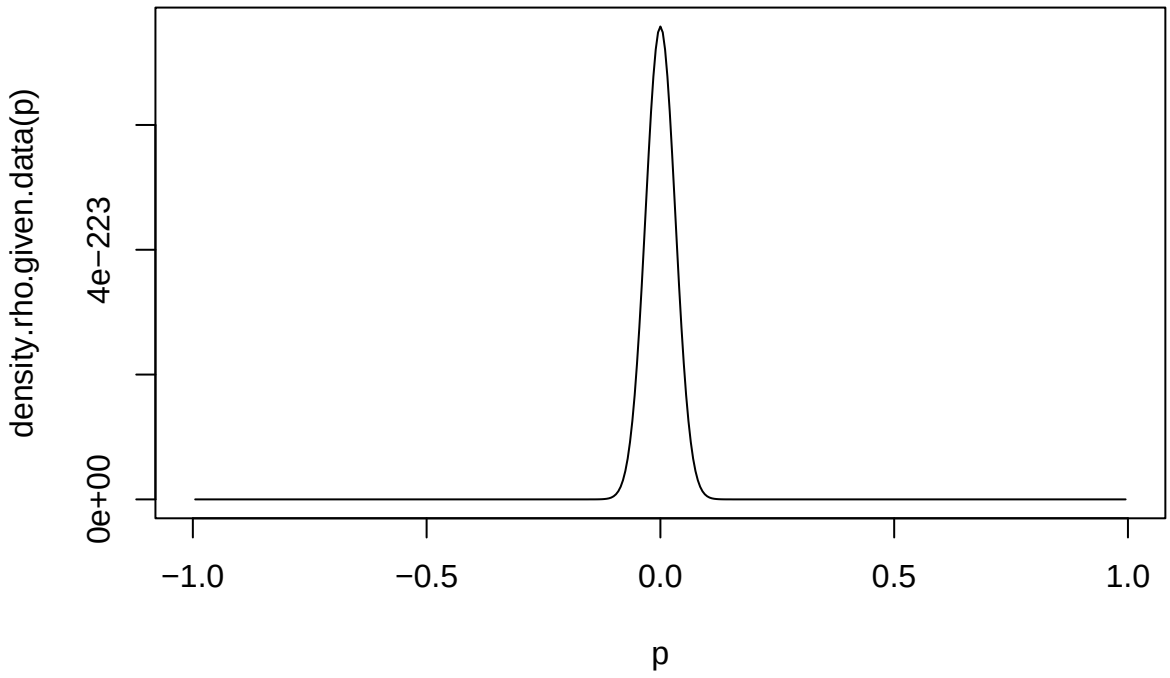
```
posCI95.HPD.s
```

```
##  3.009567% 98.00957%
## 0.4836529 0.7250935
```

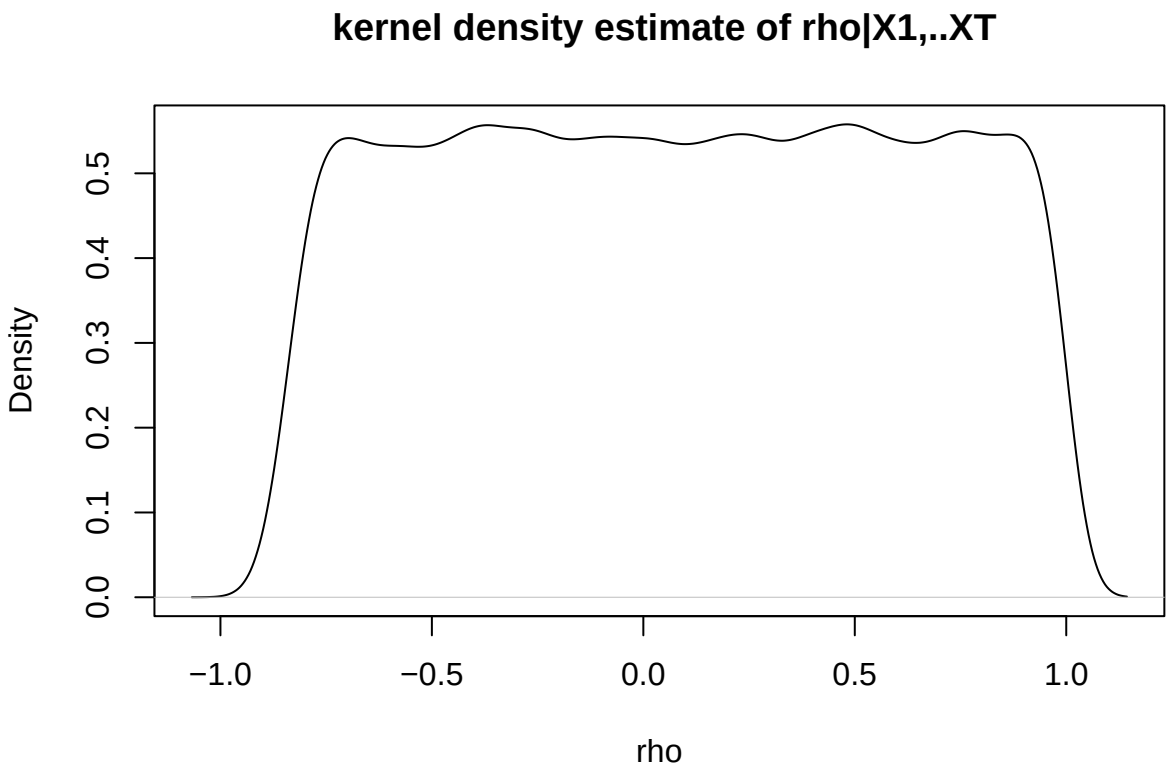
4

```
 ###########4
#Generate simulated data
```

```r
set.seed(123)
T <- 10
rho <- 0.5
X <- numeric(T)
X[1] <- rnorm(1, 0, 1)
for (t in 2:T) {
  X[t] <- rnorm(1, mean = rho * X[t-1], sd = sqrt(1 - rho^2))
}

density.rho.given.data <- function(rho)
{
  X.t <- 0
  for (t in 2:T)
  {
    X.t <- (X.t +  sum((X[t])- rho*(X[t-1])))
  }
  e.fun.val <- exp(-X.t/(2*(1-rho^2)))

  densty <- (2*pi)^(-T/2) *(1-rho^2)^( (1-T)/2) * exp( (-X[1]^2)/2 ) *e.fun.val
  return(densty)
}
p <- seq(-1,1, 0.005)
plot(p, density.rho.given.data(p), type = "l")
```

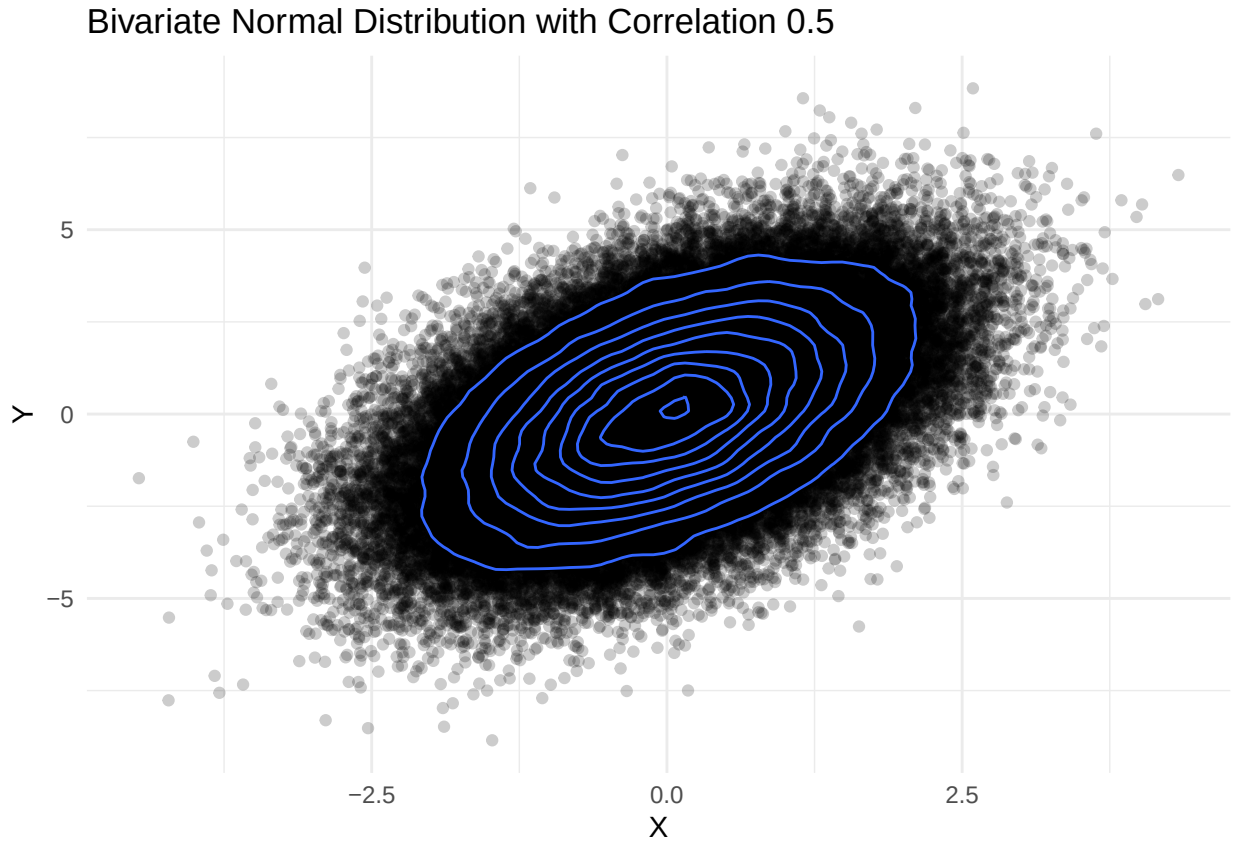


```r
#### max val occur at rho = 0 by viewing fig
c <- density.rho.given.data(0)

accept.reject <- function()
{
  accept <- 0
  while (accept == 0) {
    pros <- runif(1, min = -1, max = 1) ## proposal density U(-1,1)
    U <- runif(1)
    ratio <- density.rho.given.data(pros)/(c*dunif(pros, min = -1, max = 1))
    if(U <= ratio)
    {
      accept <- 1
      return(pros)
    }

  }
}


N <- 1e5
samp <- numeric(length = N)

counts <- numeric(length = N)
for(i in 1:N)
{
  samp[i] <- accept.reject()
}
plot(density(samp), main = " kernel density estimate of rho|X1,..XT", xlab = "rho")
```



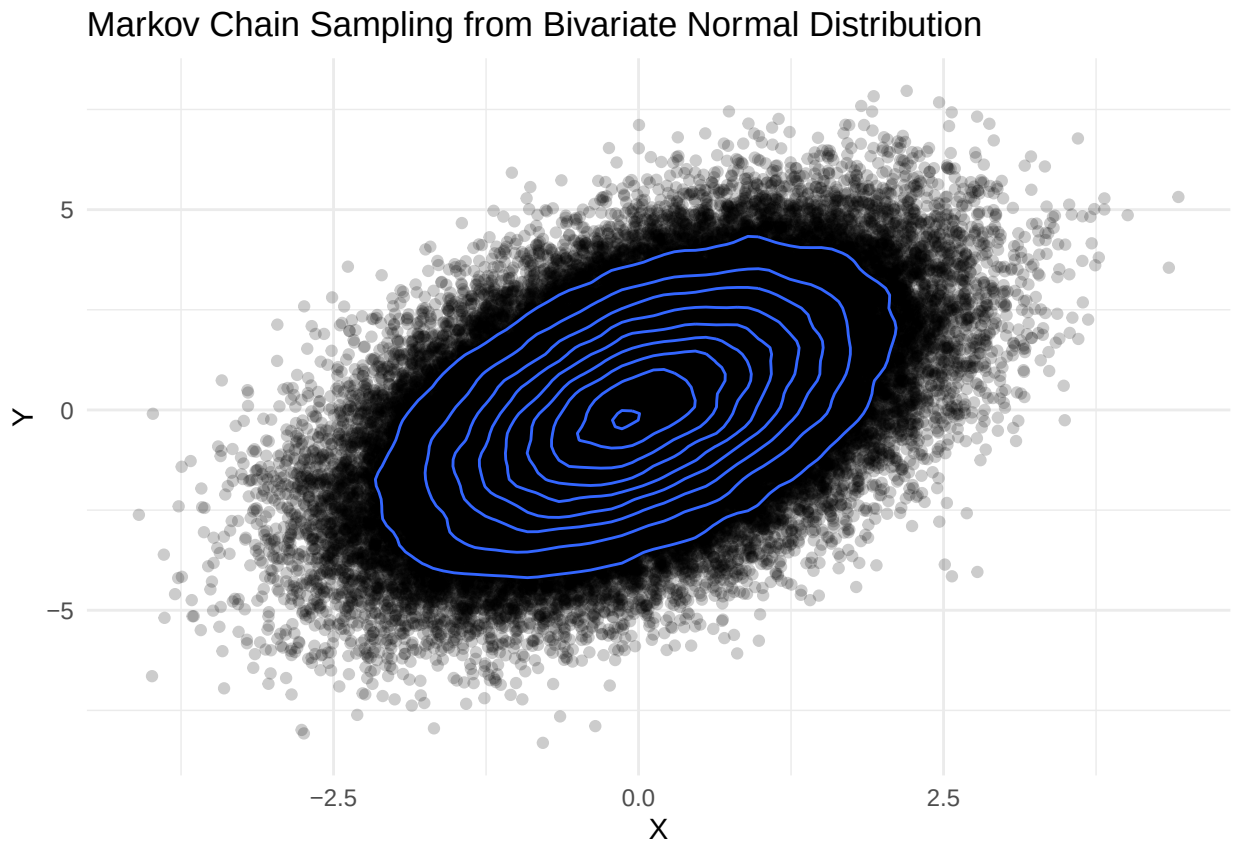
5

```r
library(MASS)
library(ggplot2)

# parameters
mu <- c(0, 0)           # means
sigma <- matrix(c(1, 1, 1, 4), nrow = 2)  # covariance matrix
B <- 10^5               # number of samples

# samples
samples <- mvrnorm(n = B, mu = mu, Sigma = sigma)

# Create data frame
df <- data.frame(X = samples[,1], Y = samples[,2])

#  scatter plot with 2D-kernel density heatmap
ggplot(df, aes(x = X, y = Y)) +
  geom_point(alpha = 0.2) +
  geom_density_2d() +
  labs(x = "X", y = "Y", title = "Bivariate Normal Distribution with Correlation 0.5") +
  theme_minimal()
```

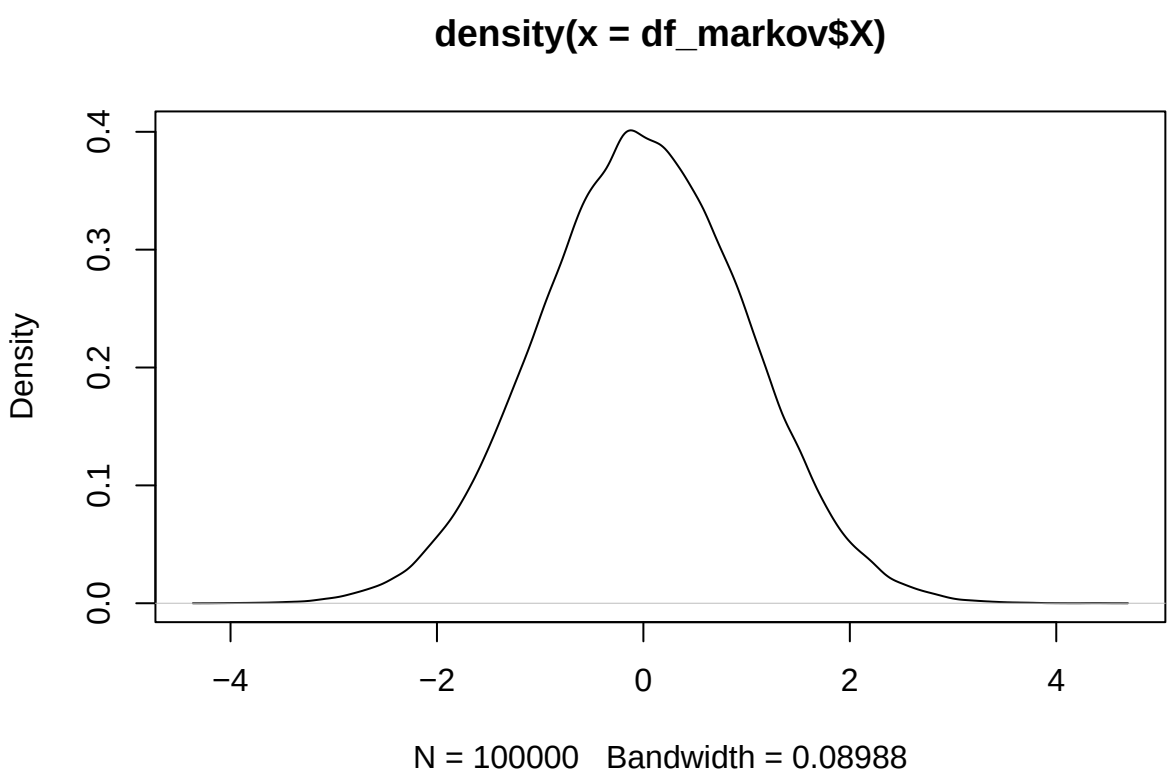



```r
# initial conditions and parameters
B0 <- 10^3
B <- 10^5
rho <- 0.5  # Correlation
X <- numeric(B0 + B)
Y <- numeric(B0 + B)
X[1] <- 0  # Initial value of X
Y[1] <- 0  # Initial value of Y


# functions for conditional distributions
conditional.X.given.Y <- function(Y, rho, mu1, mu2, sigma1, sigma2) {
  mu.x.given.y <- mu1 + rho * (sigma1/sigma2) * (Y - mu2)
  sd.x.given.y <- sqrt(sigma1^2 * (1 - rho^2))

   return(rnorm(1, mean = mu.x.given.y, sd = sd.x.given.y))
}

conditional.Y.given.X <- function(X, rho, mu1, mu2, sigma1, sigma2) {
  mu.y.given.x <- mu2 + rho * (sigma2/sigma1) * (X - mu1)
  sd.y.given.x <- sqrt(sigma2^2 * (1 - rho^2))

  return(rnorm(1, mean = mu.y.given.x, sd = sd.y.given.x))
}

#  sampling
```

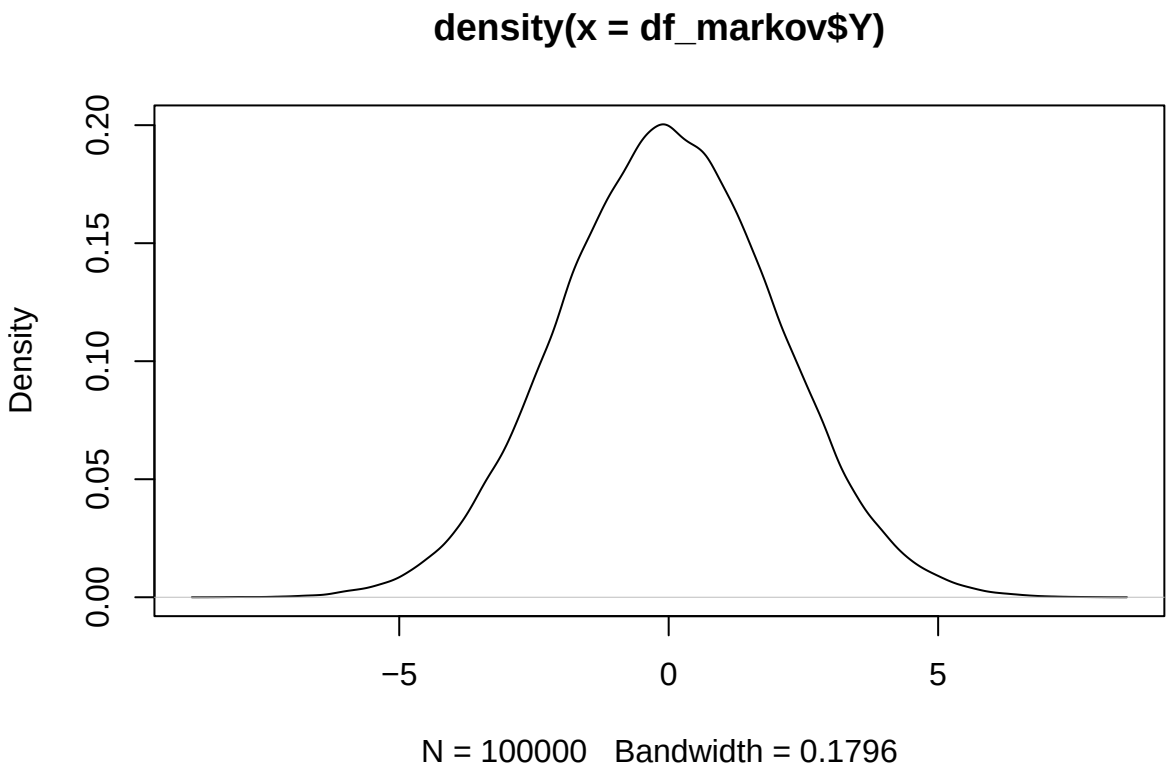
```r
for (b in 2:(B0 + B))
  {
 X[b] <- conditional.X.given.Y(Y[b-1], rho, 0, 0, 1, 2) #  mu1 = 0, mu2 = 0, sigma1 = 1, sigma2 = 2
 Y[b] <- conditional.Y.given.X(X[b], rho, 0, 0, 1, 2) #  mu1 = 0, mu2 = 0, sigma1 = 1, sigma2 = 2
}

# scatter plot with 2D-kernel density heatmap by removing B0 sample
df_markov <- data.frame(X = X[(B0 + 1):(B0 + B)], Y = Y[(B0 + 1):(B0 + B)])

ggplot(df_markov, aes(x = X, y = Y)) +
  geom_point(alpha = 0.2) +
  geom_density_2d() +
  labs(x = "X", y = "Y", title = "Markov Chain Sampling from Bivariate Normal Distribution") +
  theme_minimal()
```




```r
##1-D kernel estimate
## for X componet
kde <- density(df_markov$X)
plot(kde)
```




```r
# for Y component
kde.y <- density(df_markov$Y)
plot(kde.y)
```


density(x = df_markov$Y)
Density
0.00
0.05
0.10
0.15
0.20
−5
0
5
N = 100000 Bandwidth = 0.1796