Exercise 9

107.330 - Statistical Simulation and Computerintensive Methods, WS24

11912007 - Yahya Jabary

10.10.2024

```
set.seed(11912007)
```

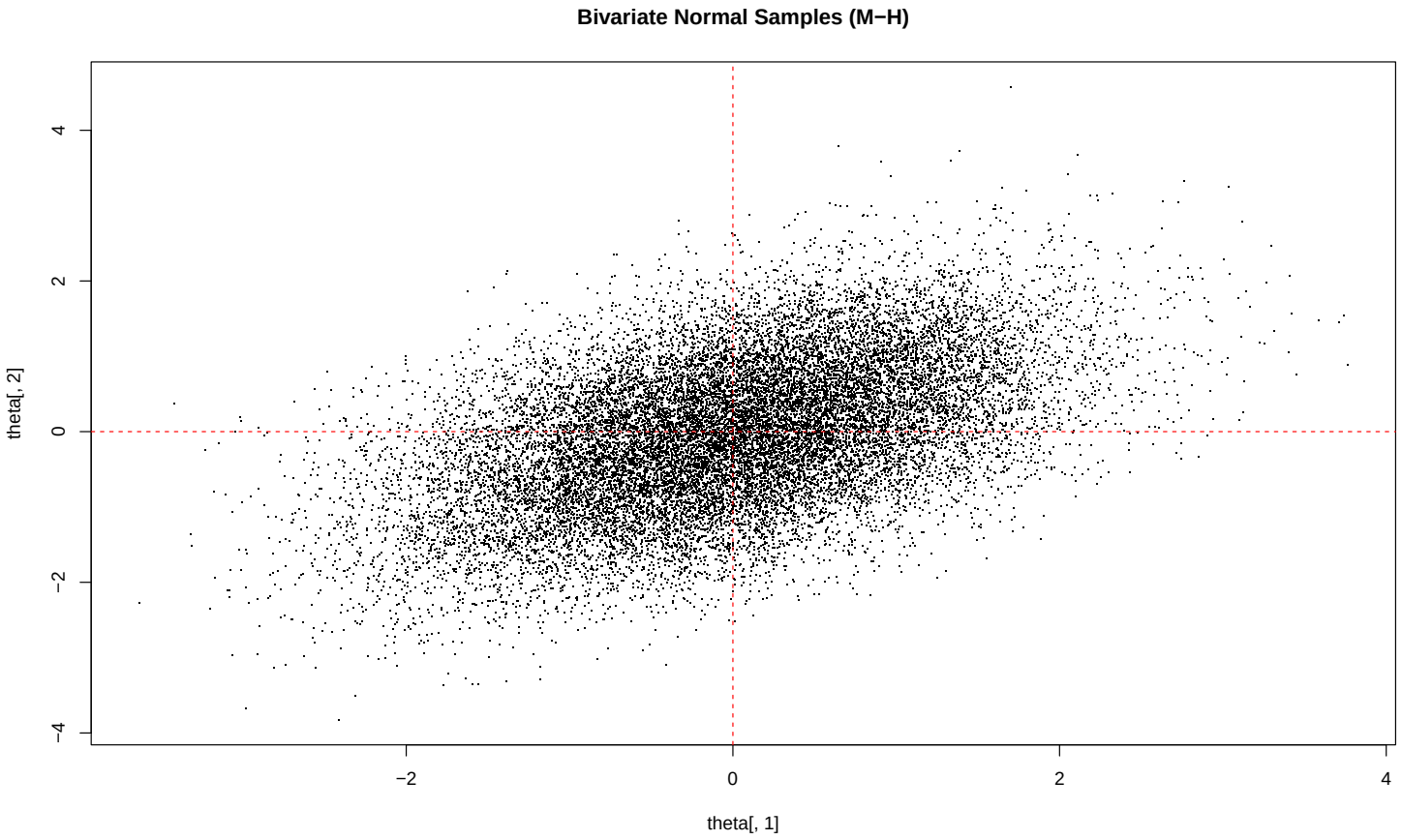
Consider the standard normal bivariate distribution with parameters $\mu = [0, 0]$ and $\Sigma = [1, \rho, \rho, 1]$ with p.d.f. given by $p(\theta_1, \theta_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}(\theta_1^2 - 2\rho\theta_1\theta_2 + \theta_2^2)}$ with $\theta_1, \theta_2 \in \mathbb{R}, \rho \in [0, 1]$ where we also know that the marginal distributions are given by $p(\theta_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\theta_i^2}$ for $i = 1, 2$.

In what follows, consider $\rho = 0.5$, chain size $M = 30000$. Choose a non-informative prior-setting to initialise the algorithm.

- Implement a Gibbs sampler to sample from this distribution.
- Use the Metropolis-Hastings algorithm with block-wise update to simulate from this distribution.
- Perform chain diagnostics on the resulting chain.

```
# parameters
rho <- 0.5
M <- 30000
burn_in <- 1000

# target distribution
target_pdf <- function(theta1, theta2) {
  return(1 / (2 * pi * sqrt(1 - rho^2)) *
         exp(-1 / (2 * (1 - rho^2)) * (theta1^2 - 2 * rho * theta1 * theta2 + theta2^2)))
}

# prooposal distribution (bivariate normal with independent components)
proposal_sd <- 0.5  # standard deviation for the proposal distribution

# initialize
theta <- matrix(0, nrow = M, ncol = 2)
theta[1,] <- rnorm(2) # initial values from standard normal

# metropolis-hastings algorithm with block-wise update
for (i in 2:M) {
  # propose new values
  theta_proposed <- rnorm(2, mean = theta[i-1,], sd = proposal_sd)

  # calculate acceptance ratio
  numerator <- target_pdf(theta_proposed[1], theta_proposed[2])
  denominator <- target_pdf(theta[i-1, 1], theta[i-1, 2])
  acceptance_ratio <- numerator / denominator

  # accept or reject
  if (runif(1) < acceptance_ratio) {
    theta[i,] <- theta_proposed
  } else {
    theta[i,] <- theta[i-1,]
  }
}

# Remove burn-in period
theta <- theta[(burn_in+1):M,]

#
```

```r
# plot
#

plot(theta[,1], theta[,2], pch = ".", main = "Bivariate Normal Samples (M-H)")
abline(h = 0, v = 0, col = "red", lty = 2)
```




```r
mean_theta1 <- mean(theta[,1])
mean_theta2 <- mean(theta[,2])
var_theta1 <- var(theta[,1])
var_theta2 <- var(theta[,2])
cov_theta1_theta2 <- cov(theta[,1], theta[,2])

cat("Sample mean of theta1:", mean_theta1, "\n")
```

```
## Sample mean of theta1: -0.01903233
```

```r
cat("Sample mean of theta2:", mean_theta2, "\n")
```

```
## Sample mean of theta2: -0.03151632
```

```r
cat("Sample variance of theta1:", var_theta1, "\n")
```

```
## Sample variance of theta1: 0.9980839
```

```r
cat("Sample variance of theta2:", var_theta2, "\n")
```

```
## Sample variance of theta2: 0.9717185
```

```r
cat("Sample covariance of theta1 and theta2:", cov_theta1_theta2, "\n")
```

```
## Sample covariance of theta1 and theta2: 0.4980701
```

```r
cat("Theoretical mean: 0\n")
```

```
## Theoretical mean: 0
```

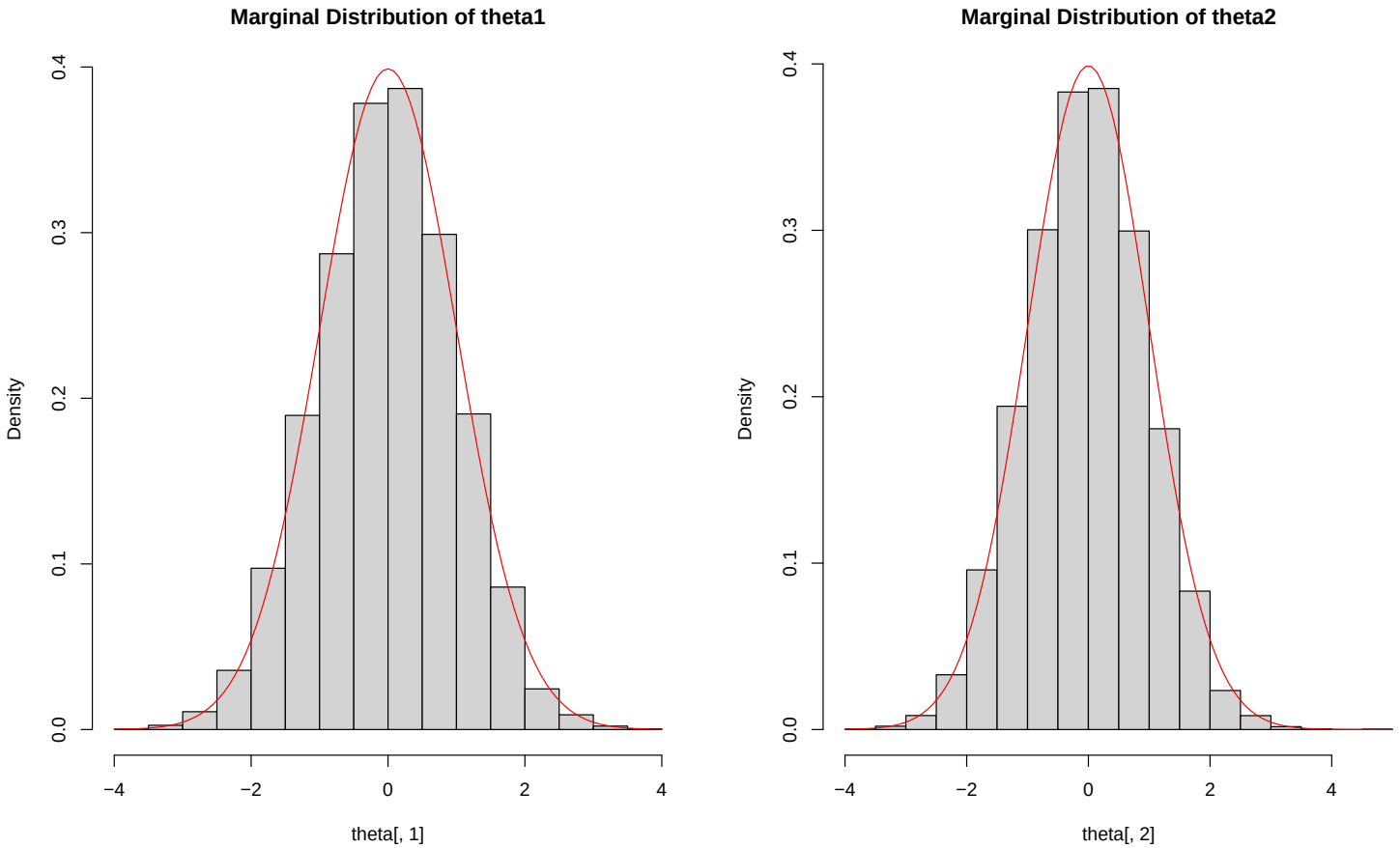
```r
cat("Theoretical variance: 1\n")
```

```
## Theoretical variance: 1
```

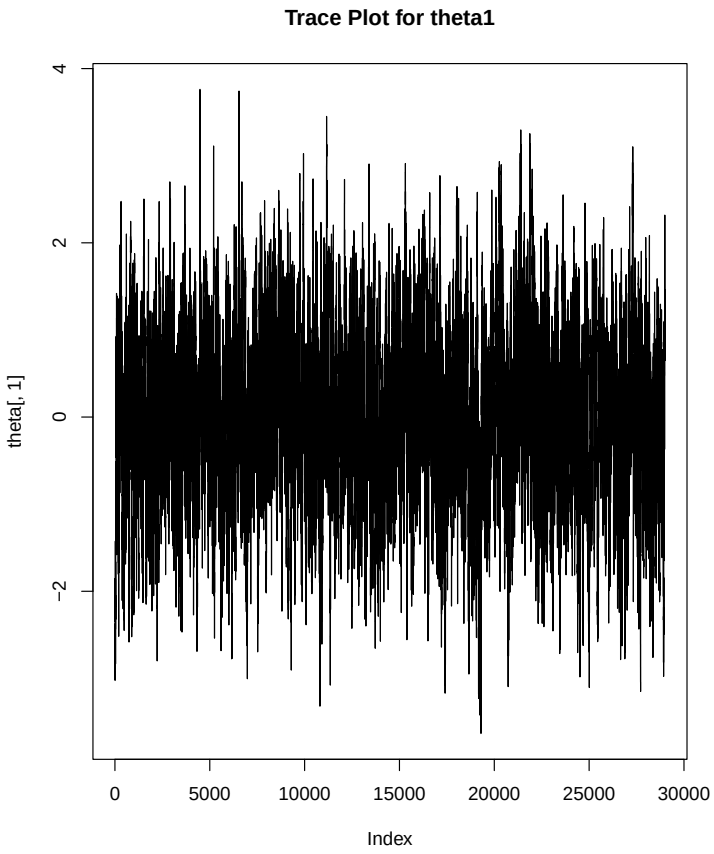
```r
cat("Theoretical covariance:", rho, "\n")
```

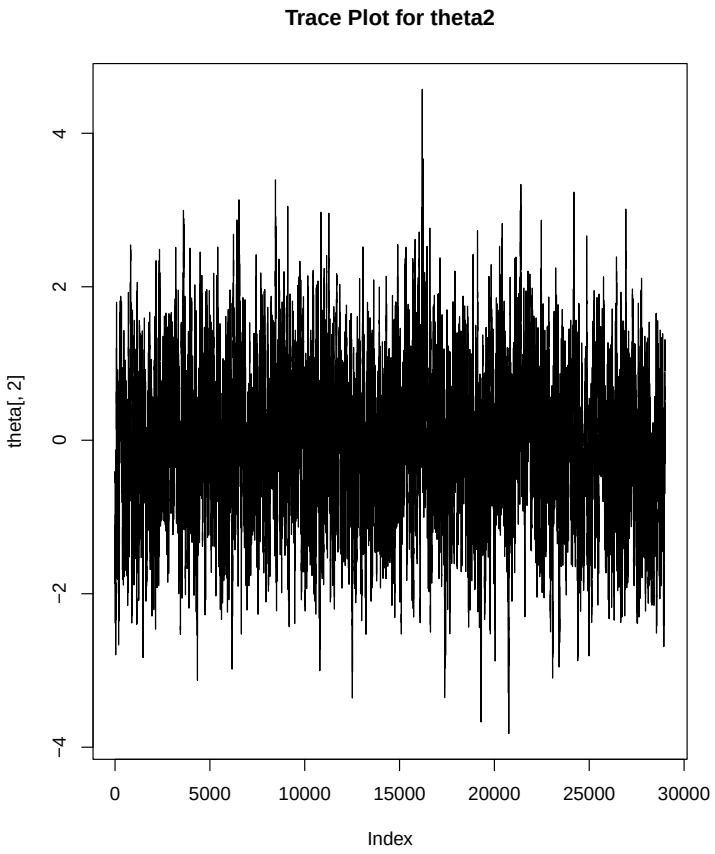
```
## Theoretical covariance: 0.5
```

```
par(mfrow = c(1, 2))
hist(theta[,1], prob = TRUE, main = "Marginal Distribution of theta1")
curve(dnorm(x), add = TRUE, col = "red")
hist(theta[,2], prob = TRUE, main = "Marginal Distribution of theta2")
curve(dnorm(x), add = TRUE, col = "red")
```

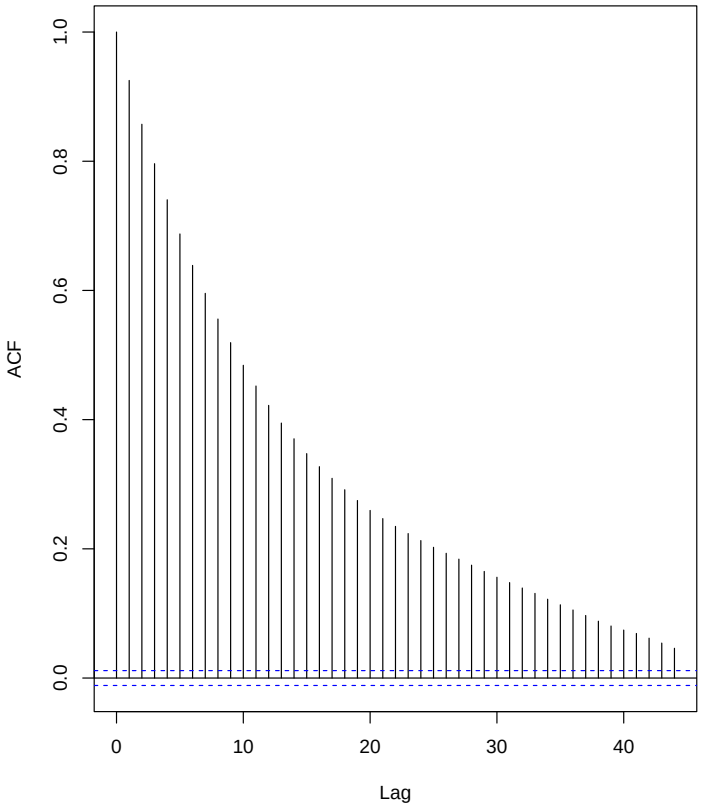


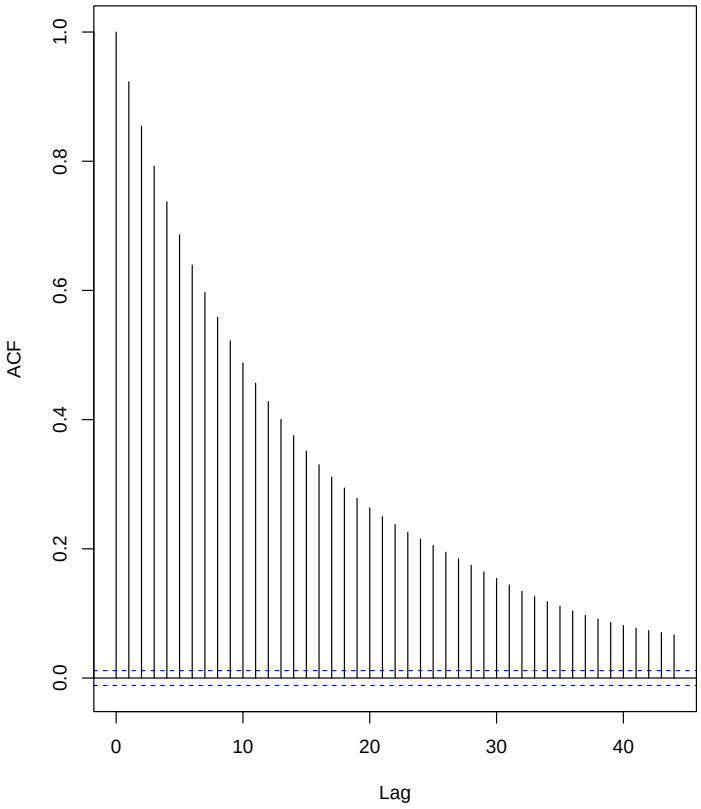

```
#
# additional diagnostics
#

# trace plots
plot(theta[,1], type = "l", main = "Trace Plot for theta1")
plot(theta[,2], type = "l", main = "Trace Plot for theta2")
```



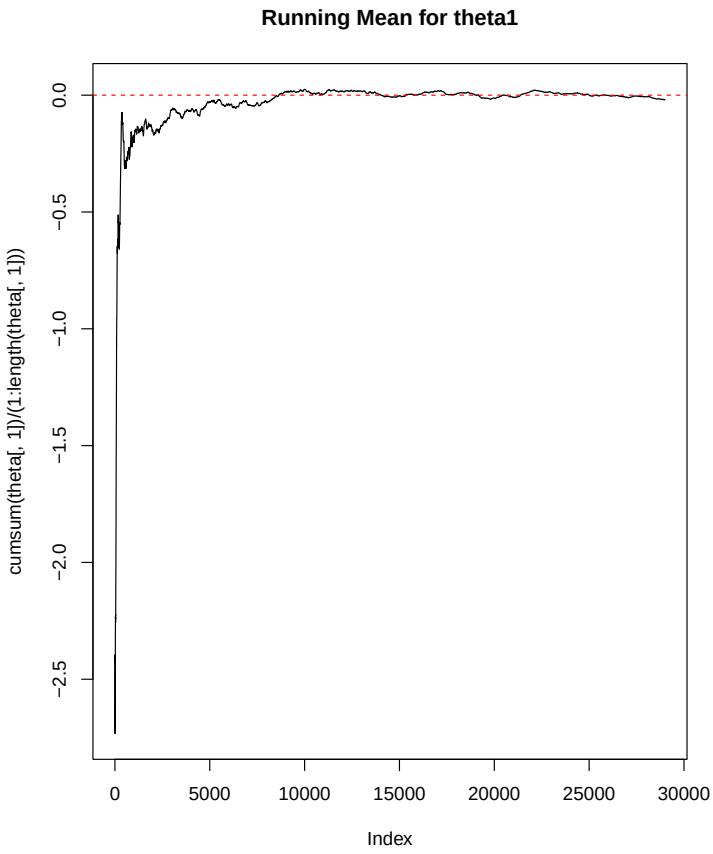


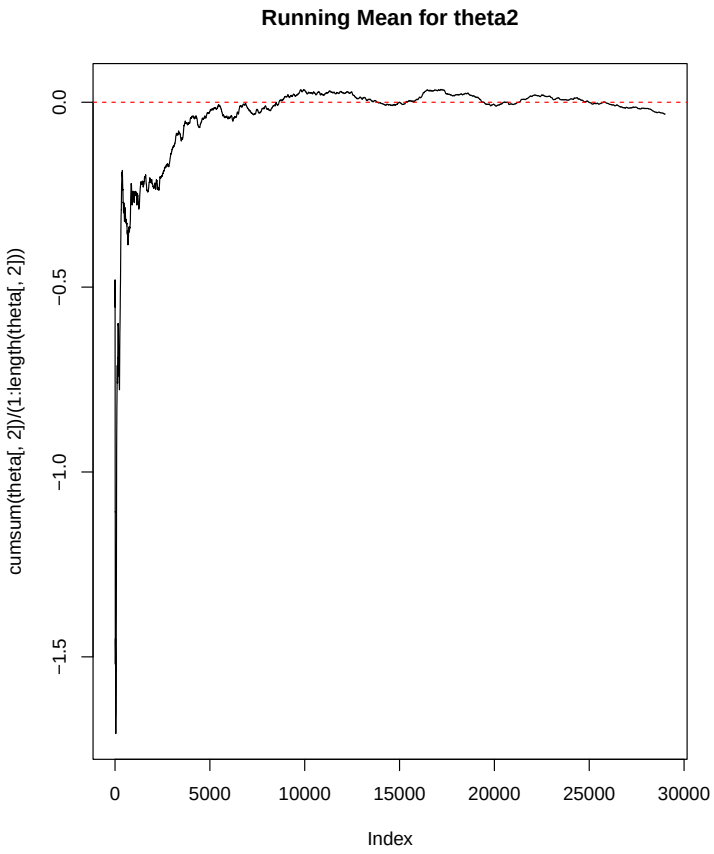

```r
# autocorrelation plots
acf(theta[,1], main = "ACF for theta1")
acf(theta[,2], main = "ACF for theta2")
```






```r
# running mean plots
plot(cumsum(theta[,1]) / (1:length(theta[,1])), type = "l", main = "Running Mean for theta1")
abline(h = 0, col = "red", lty = 2)
plot(cumsum(theta[,2]) / (1:length(theta[,2])), type = "l", main = "Running Mean for theta2")
abline(h = 0, col = "red", lty = 2)
```






```
# effective Sample Size
effectiveSize(as.mcmc(theta))
```

```
##     var1     var2
## 1077.896 1065.864
```

```
# geweke diagnostics
geweke.diag(as.mcmc(theta))
```

```
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
##    var1    var2
## -0.5296 -0.9684
```

Interpreting these additional diagnostics:

- Trace Plots: Look for good mixing (rapid up-and-down movement) and stationarity (no clear trends or patterns).
- Autocorrelation Plots: Check for quick decay in autocorrelation. High autocorrelation at large lags indicates poor mixing.
- Running Mean Plots: These should converge to the true mean (0 in this case). Look for stability in the latter part of the chain.
- Effective Sample Size: This estimates how many independent samples the chain is equivalent to. A low effective sample size relative to the actual chain length indicates high autocorrelation.
- Geweke Diagnostic: This test compares the mean of the first 10% of the chain to the last 50%. A z-score with absolute value > 2 suggests lack of convergence.