Statistical Simulation and Computerintensive Methods

Exercise 9

Markus Kiesel | 1228952

19.01.2022

Contents

Noraml Bivariate Distribution

Consider the standard normal bivariate distribution with parameters $\mu = [0\ 0]$ and $\Sigma = [1\rho\rho1]$ with p.d.f. given by

$$p(\theta_1, \theta_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}(\theta_1^2 - 2\rho\theta_1\theta_2 + \theta_2^2)}, with \theta_1, \theta_2 \in \mathbb{R}, \rho \in [0, 1] \tag{1}$$

where we also know that the marginal distributions are given by

$$p(\theta_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\theta_i^2}, i = 1, 2 \tag{2}$$

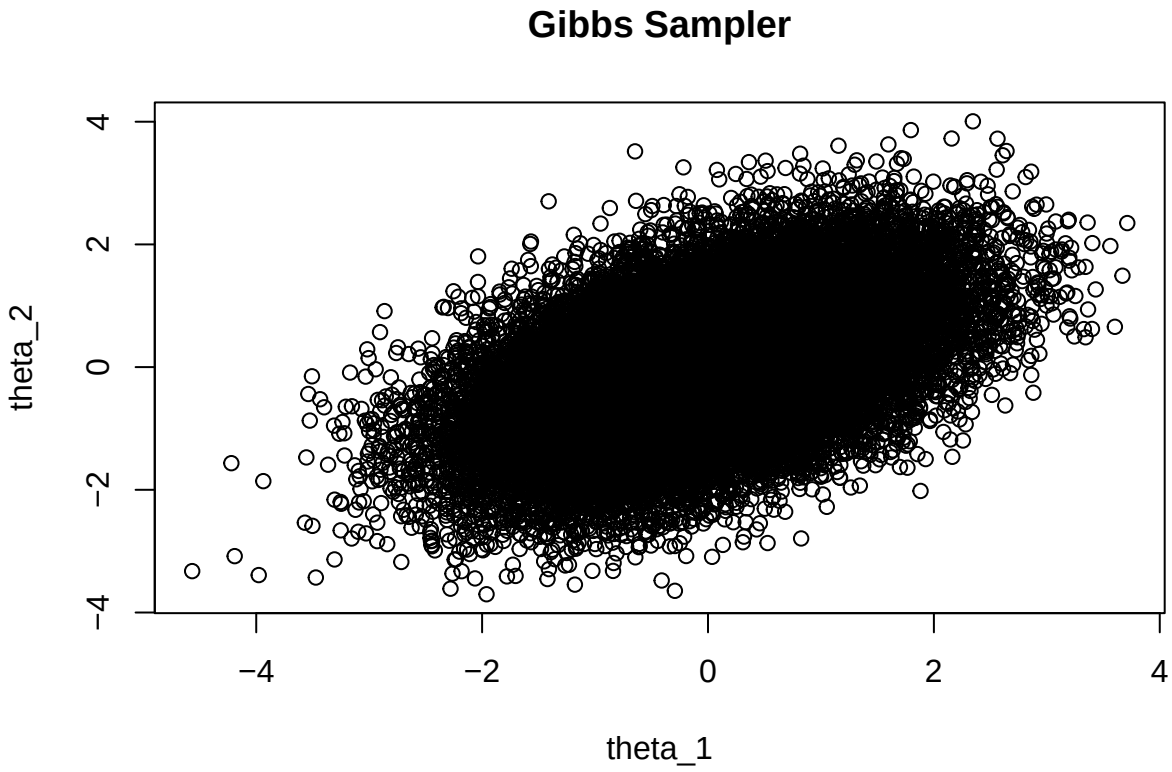
In what follows, consider $\rho = 0.5$, chain size M = 30000. Choose a non-informative prior-setting to initialize the algorithm.

```
# set Seed
MATRNR <- 1228952
set.seed(MATRNR)
```

Gibbs Sampler

Implement a Gibbs sampler to sample from this distribution.

```
# implement gibbs sampler with given defaults
gibbs_sampler <- function(rho = 0.5, M = 30000) {
    # choose non informative prior
    theta_1 <- rep(0, M)
    theta_2 <- rep(0, M)
    for (i in 2:M) {
        # compute theta_1 given theta_2
        theta_1[i] <- rnorm(1, mean = rho * theta_2[i - 1], sd = sqrt(1 - rho^2))
        # compute theta 2 given theta 1
        theta_2[i] <- rnorm(1, mean = rho * theta_1[i], sd = sqrt(1 - rho^2))
    }
    # return as matrix
    return(cbind(theta_1, theta_2))
}

# run gibbs sampler with given defaults
results_gs <- gibbs_sampler()
# plot output
plot(results_gs, main = "Gibbs Sampler")
```

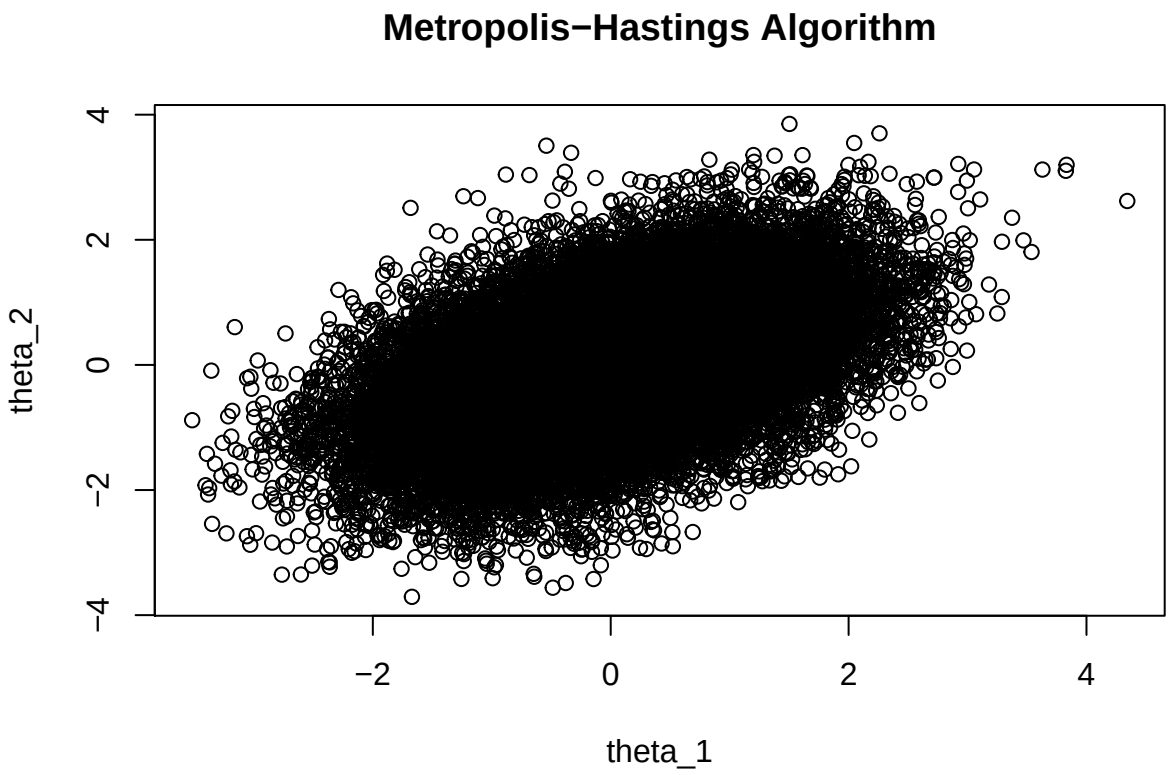


Metropolis-Hastings Algorithm

Use the Metropolis-Hastings algorithm with block-wise update to simulate from this distribution.

```
# implement Metropolis-Hastings algorithm
metropolis_hastings_algorithm <- function(rho = 0.5, M = 30000) {
    # set initial values
    theta_matrix <- matrix(0, M, 2)
    sigma <- matrix(c(1, rho, rho, 1), ncol = 2)
    # compute probability
    probability <- dmvnorm(theta_matrix[1, ], mean = c(0, 0), sigma = matrix(c(1,
        rho, rho, 1), ncol = 2))
    # loop until M values created
    i <- 1
    while (i <= M) {
        t_1 <- rnorm(1, ifelse(i > 1, theta_matrix[i - 1, 1], 0), 0.5)
        t_2 <- rnorm(1, ifelse(i > 1, theta_matrix[i - 1, 2], 0), 0.5)
        prob <- dmvnorm(c(t_1, t_2), sigma = sigma)
        alpha <- min(prob/probability, 1)
        # test if random value smaller alpha
        if (runif(1) <= alpha) {
            theta_matrix[i, ] <- c(t_1, t_2)
            probability = prob
            i <- i + 1
        }
    }
    colnames(theta_matrix) <- c("theta_1", "theta_2")
    return(theta_matrix)
}
```

```r
# run with default values
results_mh <- metropolis_hastings_algorithm()

plot(results_mh, main = "Metropolis-Hastings Algorithm")
```

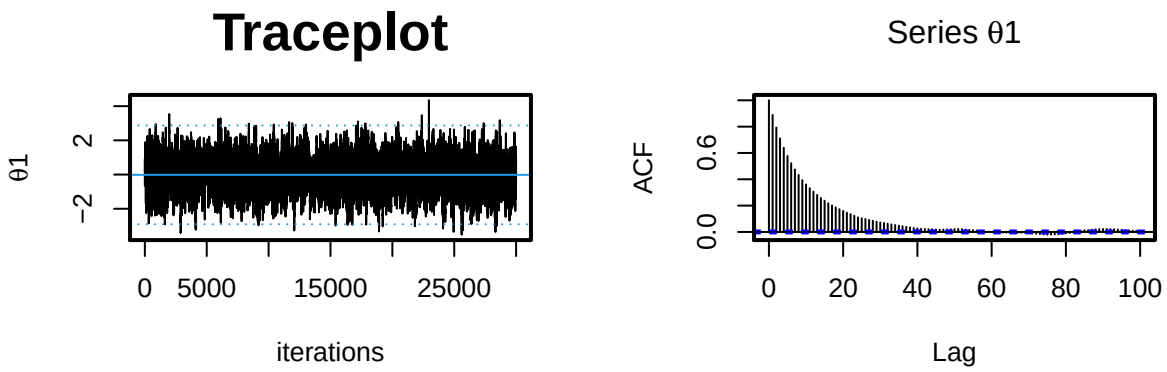


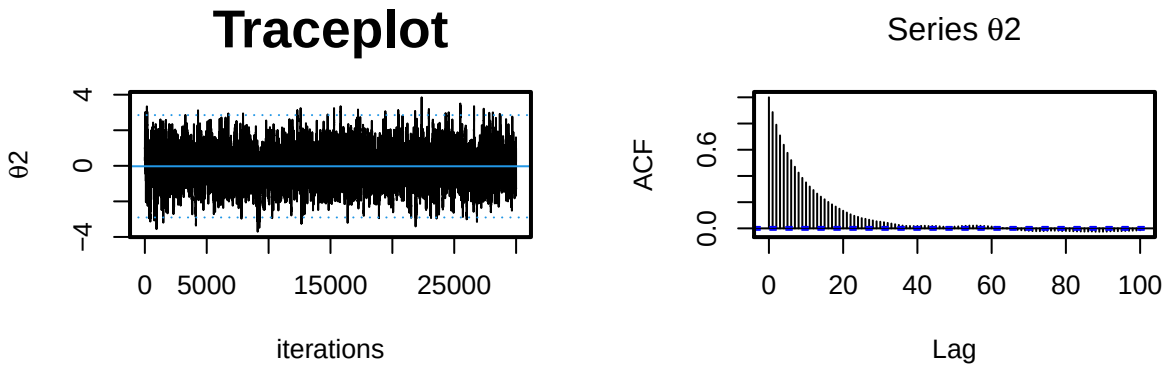
Chain Diagnostic

Perform chain diagnostics on the resulting chain.

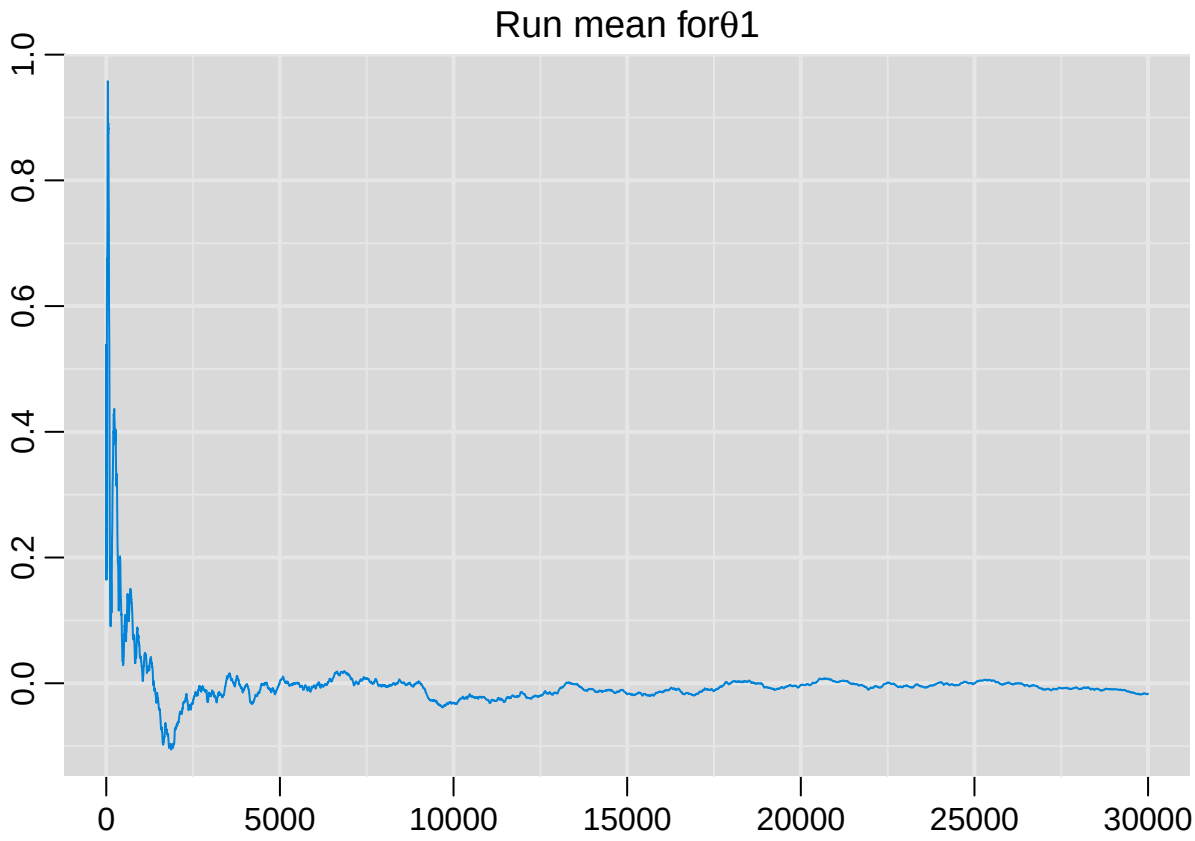
```r
# set values
M <- dim(results_mh)[1]
t1 <- results_mh[, 1]
t2 <- results_mh[, 2]
# trace plots
par(mfrow = c(2, 2), mar = c(5, 5, 4, 2))
plot(1:M, t1, type = "l", ylim = c(min(t1, mean(t1) - 3 * sd(t1)), max(t1, mean(t1) +
    3 * sd(t1))), xlab = "iterations", ylab = expression(paste(theta, "1")), cex.main = 2,
    main = "Traceplot")
abline(h = mean(t1), lty = 1, col = 4)
abline(h = mean(t1) + 3 * sd(t1), lty = 3, col = 4)
abline(h = mean(t1) - 3 * sd(t1), lty = 3, col = 4)
box(lwd = 2)

# ACF plots with run mean
acf(t1, lag.max = 100, main = expression(paste("Series ", theta, "1")), cex.main = 2)
box(lwd = 2)
plot(1:M, t2, type = "l", ylim = c(min(t2, mean(t2[-(1:200)]) - 3 * sd(t2[-(1:200)])),
    max(t2, mean(t2[-(1:200)]) + 3 * sd(t2[-(1:200)]))), xlab = "iterations", ylab = expression(paste(t
```

```
    "2")), cex.main = 2, main = "Traceplot")
abline(h = mean(t2[-(1:200)]), lty = 1, col = 4)
abline(h = mean(t2[-(1:200)]) + 3 * sd(t2[-(1:200)]), lty = 3, col = 4)
abline(h = mean(t2[-(1:200)]) - 3 * sd(t2[-(1:200)]), lty = 3, col = 4)
box(lwd = 2)
acf(t2, lag.max = 100, main = expression(paste("Series ", theta, "2")), cex.main = 2)
box(lwd = 2)
```

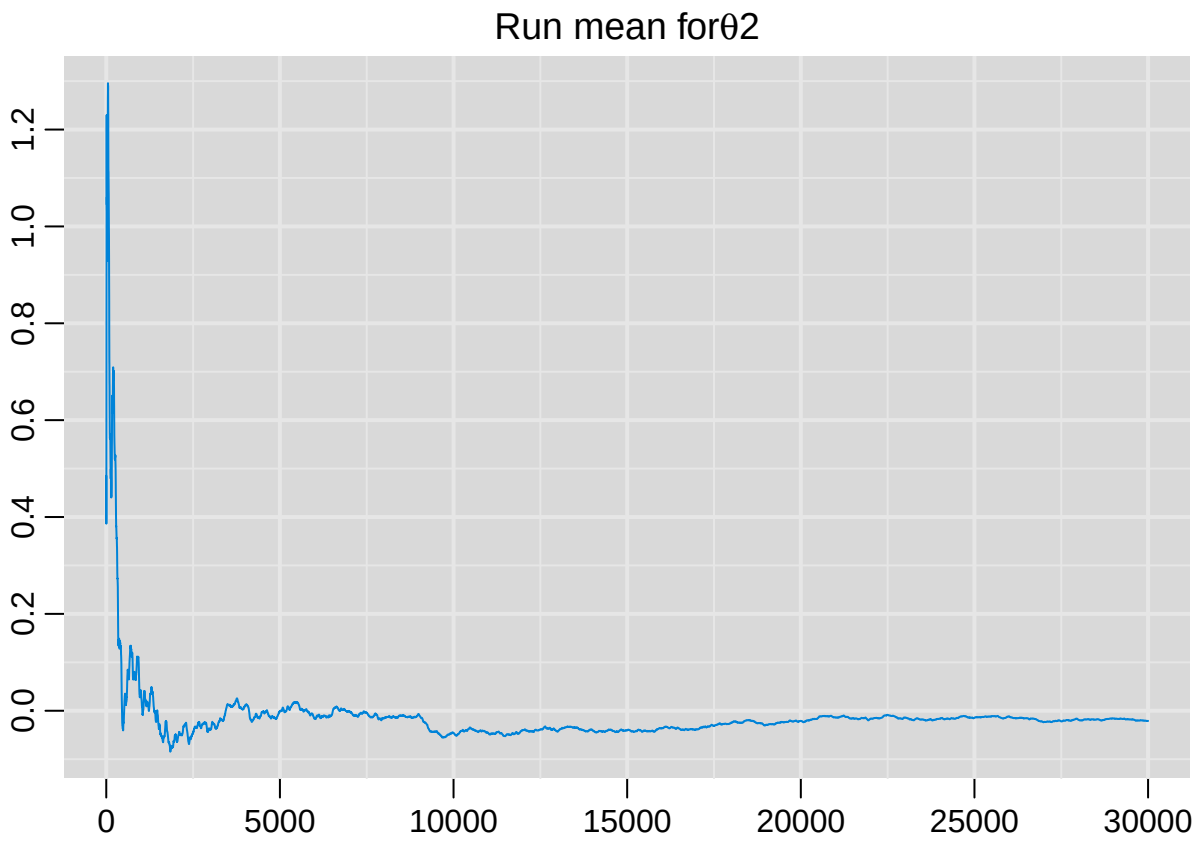




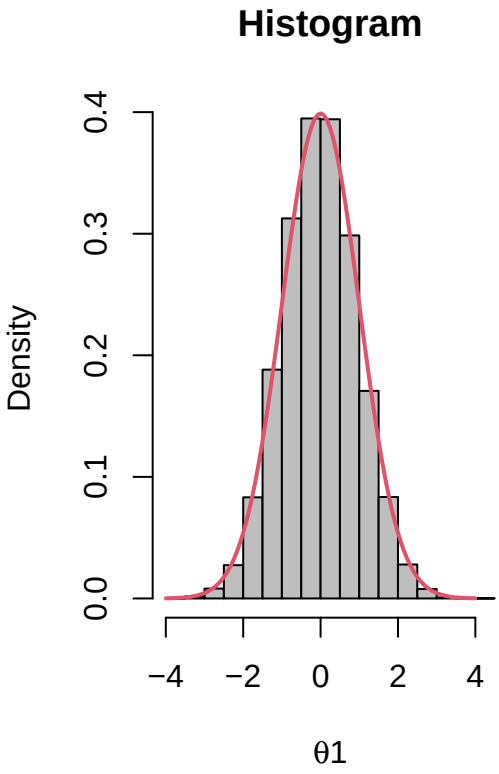

```
# histograms of marginal posteriors
rmeanplot(t1, lwd = 2, main = expression(paste("Run mean for", theta, "1")), mar = c(2,
    2, 1.5, 1) + 0.1)
```

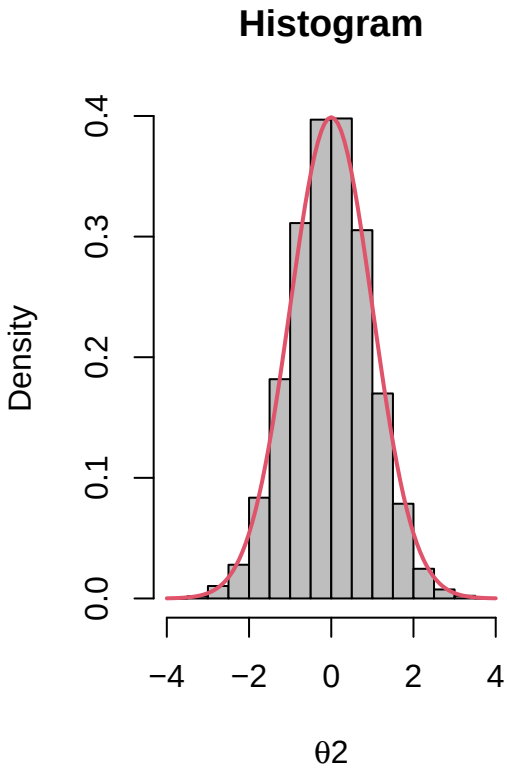



```
rmeanplot(t2, lwd = 2, main = expression(paste("Run mean for", theta, "2")), mar = c(2,
    2, 1.5, 1) + 0.1)
```




```
par(mfrow = c(1, 2), mar = c(5, 5, 4, 2))
hist(t1, freq = F, col = "grey", main = "Histogram", xlab = expression(paste(theta,
    "1")))
curve(dnorm(x, 0, 1), from = -4, 4, add = T, lwd = 2, col = 2)
hist(t2, freq = F, col = "grey", main = "Histogram", xlab = expression(paste(theta,
    "2")))
curve(dnorm(x, 0, 1), from = -4, 4, add = T, lwd = 2, col = 2)
```


Histogram
Density
0.0
0.1
0.2
0.3
0.4
−4
−2
0
2
4
θ1


Histogram
Density
0.0
0.1
0.2
0.3
0.4
−4
−2
0
2
4
θ2