Exercise 9 - MCMC - Gibbs Sampling, Metropolis-Hastings Sampling

Bosse Behrens, st.id 12347333

2024W

```
set.seed(12347333)
```

Task 1

We implement a Gibbs sampler to sample from the provided distribution. To do so we first need to calculate the conditional distributions. With the provided bivariate and marginal distributions ,we obtain

$$
\begin{aligned}
p(\theta_1|\theta_2) &= \frac{p(\theta_1,\theta_2)}{p(\theta_2)} \\
&= \frac{1}{2\pi\sqrt{(1-\rho^2)}} \cdot e^{-\frac{1}{2(1-\rho^2)}(\theta_1^2-2\rho\theta_1\theta_2+\theta_2^2)} \cdot \sqrt{2\pi}e^{\frac{1}{2}\theta_2^2} \\
&= \frac{1}{\sqrt{2\pi(1-\rho^2)}} \cdot e^{-\frac{1}{2(1-\rho^2)}(\theta_1^2-2\rho\theta_1\theta_2+\theta_2^2)} \cdot e^{-\frac{-\theta_2^2+\rho^2\theta_2^2}{2(1-\rho^2)}} \\
&= \frac{1}{\sqrt{2\pi(1-\rho^2)}} \cdot e^{-\frac{1}{2(1-\rho^2)}(\theta_1^2-2\rho\theta_1\theta_2+\rho^2\theta_2^2)} \\
&= \frac{1}{\sqrt{2\pi(1-\rho^2)}} \cdot e^{-\frac{1}{2(1-\rho^2)}(\theta_1-\rho\theta_2)^2}
\end{aligned}
$$

We can observe this is again the densioty function of a normal distribution and can read the parameters. Therefore we get $\theta_1|\theta_2 \sim N(\rho\theta_2, 1-\rho^2)$. To obtain the other conditional distribution is similar since the marginal distr. are symmeterical, therefore $\theta_2|\theta_1 \sim N(\rho\theta_1, 1-\rho^2)$. No we can implement the Gibbs sampler. We include the burn-in cut-off and an option for thinning in the function, but for now set the default thinning parameter so it effectively does not exist. If the need arises we can change it. Also we set $\theta_1 = \theta_2 = 0$ as default so we have an unifromative prior-setting to initialise the algorithm.

```
gibbs_sampler <- function(t1=0, t2=0, rho=0.5, M=30000, burn_in=5000, thin=1){
  #preallocate vectors (was very uneffective without)
  t1_samples <- numeric(M)
  t2_samples <- numeric(M)

  #initializing
  t1_current <- t1
  t2_current <- t2

  sigma <- sqrt(1 - rho^2)

  for(i in 1:M) {
    #sample theta1 | theta2
    t1_new <- rnorm(1, mean = rho * t2_current, sd = sigma)
    t1_samples[i] <- t1_new

    #sample theta2 | theta1
```

```
    t2_new <- rnorm(1, mean = rho * t1_new, sd = sigma)
    t2_samples[i] <- t2_new

    #update values
    t1_current <- t1_new
    t2_current <- t2_new
  }

  #burn-in and potential thinning

  keep <- seq(from = burn_in + 1, to = M, by = thin)
  t1_final <- t1_samples[keep]
  t2_final <- t2_samples[keep]

  return(list(t1 = t1_final, t2 = t2_final))
}
```

Task 2

Now we want to write a Metropolis-Hastings algorithm with block-wise update to simulate. First we need to get the acceptance probability. Since the target distribution is just as the conditional ones normal and we have $p(\theta_1)p(\theta_1|\theta_2) = p(\theta_1, \theta_2)$ and similar $p(\theta_2)p(\theta_2|\theta_1) = p(\theta_1, \theta_2)$, we get for the acceptance probability $\alpha(\theta_1, \theta_2)$:

$$\begin{aligned}\alpha(\theta_1, \theta_2) &= min(\frac{p(\theta_2)p(\theta_2|\theta_1)}{p(\theta_1)p(\theta_1|\theta_2)}, 1) \\ &= min(1, 1) \\ &= 1\end{aligned}$$

This makes sense since the proposal distribution exactly matches the conditional distributions of the target. Now we implement the Metropolis-Hastings algorithm. The acceptance/rejection part we can now leave out since it will always be accepted.

```
MH_sampler <- function(t1=0, t2=0, rho=0.5, M=30000, burn_in=5000, thin=1) {
  #preallocate vectors (was very ineffective without)
  t1_samples <- numeric(M)
  t2_samples <- numeric(M)

  #initializing
  t1_samples[1] <- t1
  t2_samples[1] <- t2

  for(i in 2:M) {
    #update t1 given t2 from the previous step, tehrefore i-1 as index
    t1_samples[i] <- rnorm(1, mean = rho * t2_samples[i - 1], sd = sqrt(1 - rho^2))

    #update t2 given t1 froms ame step
    t2_samples[i] <- rnorm(1, mean = rho * t1_samples[i], sd = sqrt(1 - rho^2))
  }

  keep <- seq(from = burn_in + 1, to = M, by = thin)
  t1_final <- t1_samples[keep]
  t2_final <- t2_samples[keep]
```

```
  return(list(t1 = t1_final, t2 = t2_final))
}
```

Task 3

Now we want to perform chain diagnostics on the resulting chain. First we use the Gibbs-sampler. We also set the ρ and the chain size.

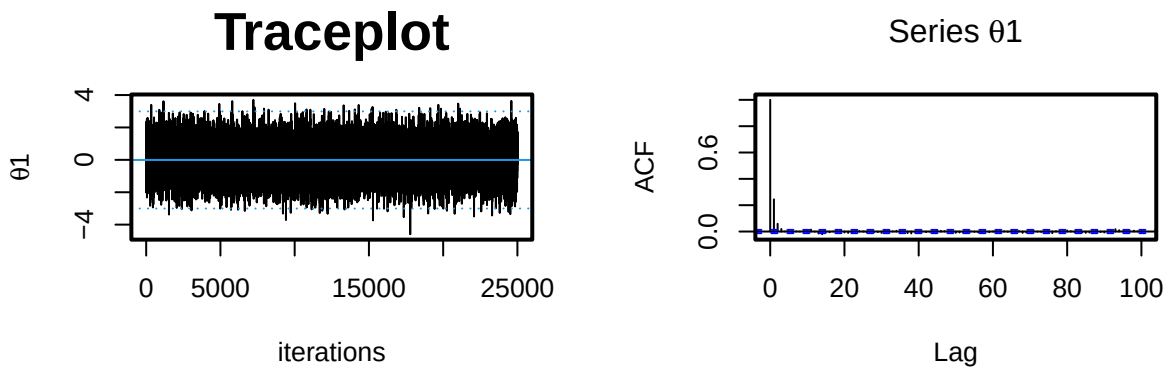
```
burn_in <- 5000
gibbs <- gibbs_sampler(t1=0, t2=0, burn_in=burn_in)
t1 <- gibbs$t1
t2 <- gibbs$t2

rho <- 0.5
M <- 30000 - burn_in
```

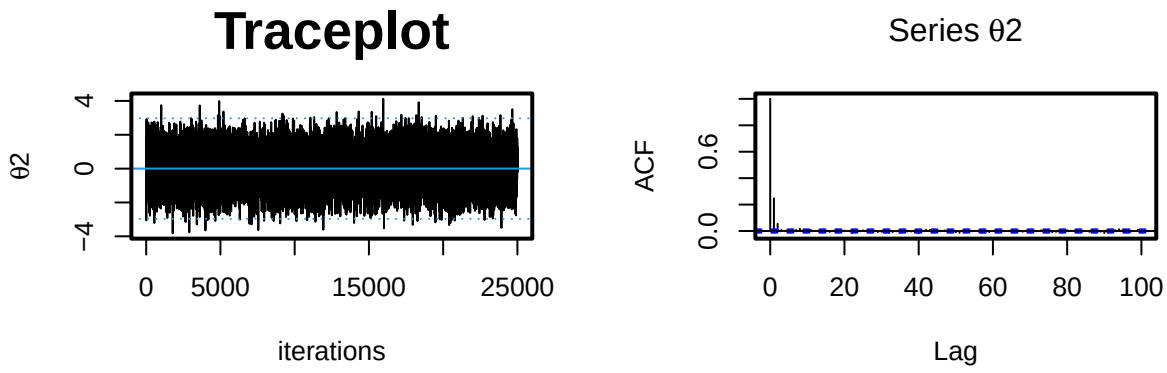
Now we plot the chain diagnostics.

```
library(mcmcplots)
```

```
## Lade nötiges Paket: coda
```

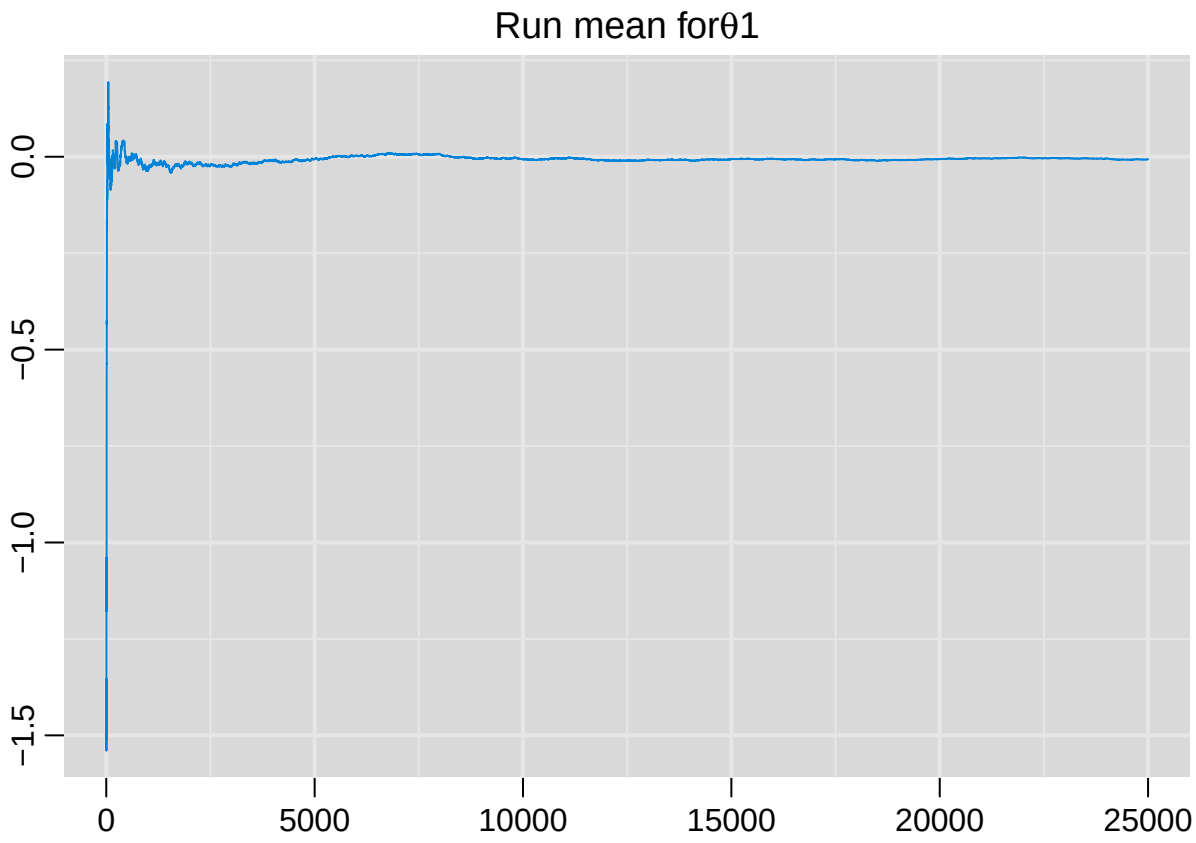
```
# trace plots
par(mfrow=c(2,2),mar=c(5,5,4,2))
plot(1:M,t1,type="l",ylim=c(min(t1,mean(t1)-3*sd(t1)),
      max(t1,mean(t1)+3*sd(t1))),xlab="iterations",
     ylab=expression(paste(theta,"1")),cex.main=2,main="Traceplot")
abline(h=mean(t1),lty=1,col=4);
abline(h=mean(t1)+3*sd(t1),lty=3,col=4)
abline(h=mean(t1)-3*sd(t1),lty=3,col=4); 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(theta,"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)
```





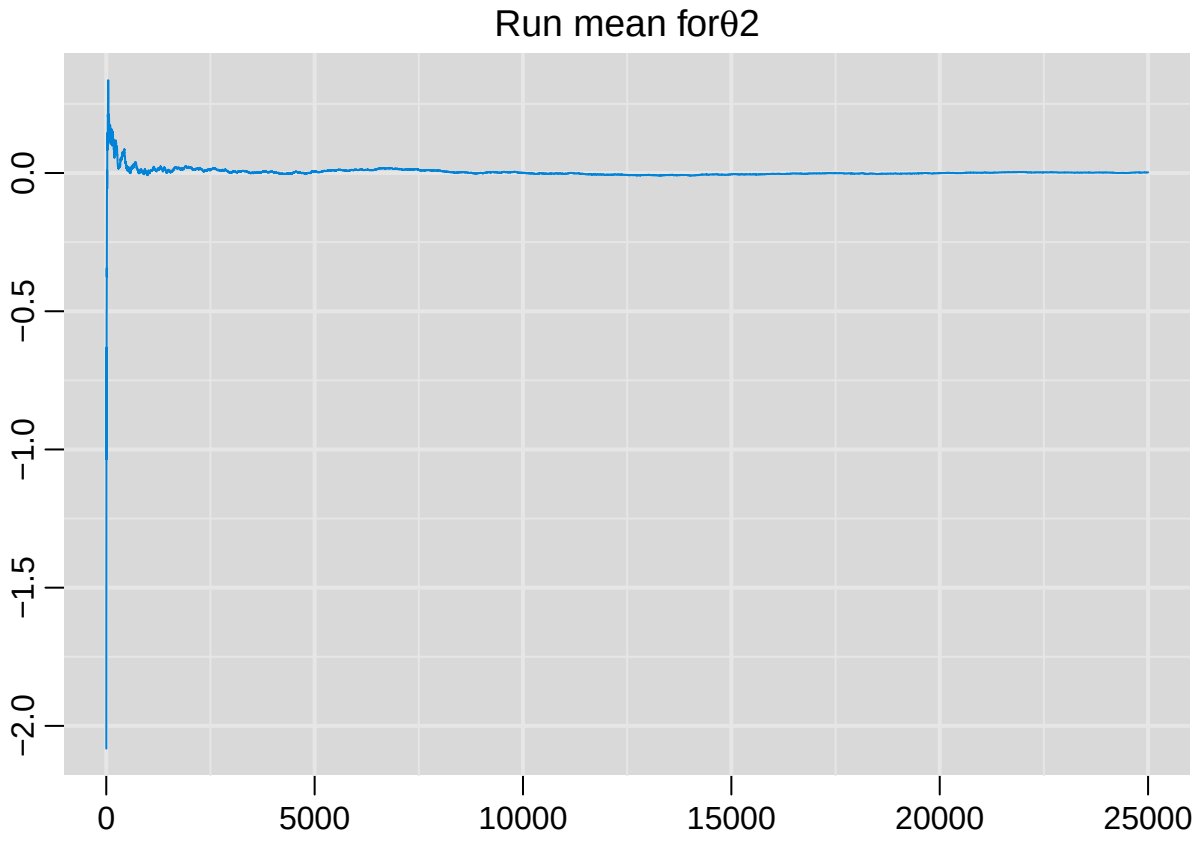

```
# histrograms of marginal posteriors
rmeanplot(t1,lwd=2,main=expression(paste("Run mean for",theta,"1")),mar=c(2,2,1.5,1)+ 0.1)
```

```
## Warning in rmeanplot(t1, lwd = 2, main = expression(paste("Run mean for", :
## Argument 'mcmcout' did not have valid variable names, so names have been
## created for you.
```

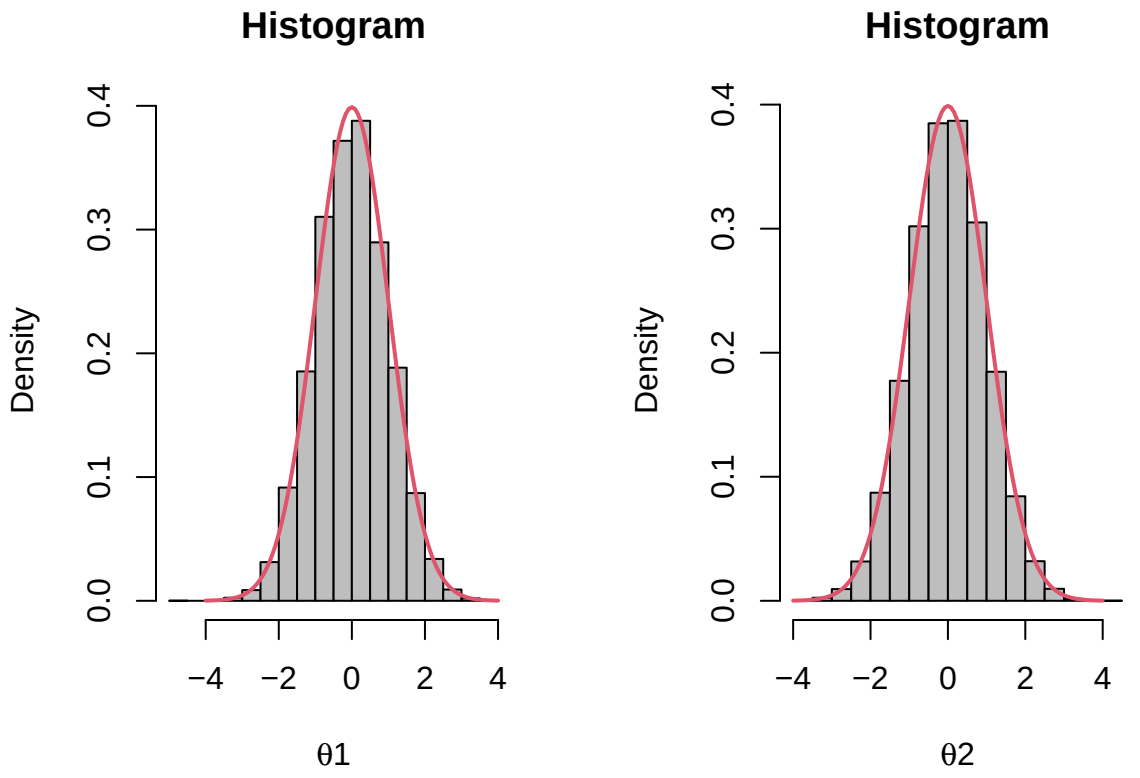



```
rmeanplot(t2,lwd=2,main=expression(paste("Run mean for",theta,"2")),mar=c(2,2,1.5,1)+ 0.1)
```

```
## Warning in rmeanplot(t2, lwd = 2, main = expression(paste("Run mean for", :
## Argument 'mcmcout' did not have valid variable names, so names have been
## created for you.
```




```
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)
```

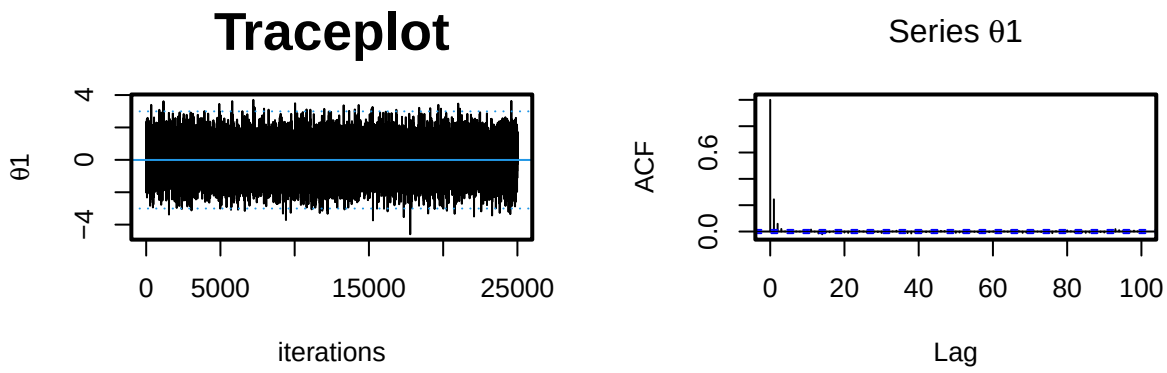


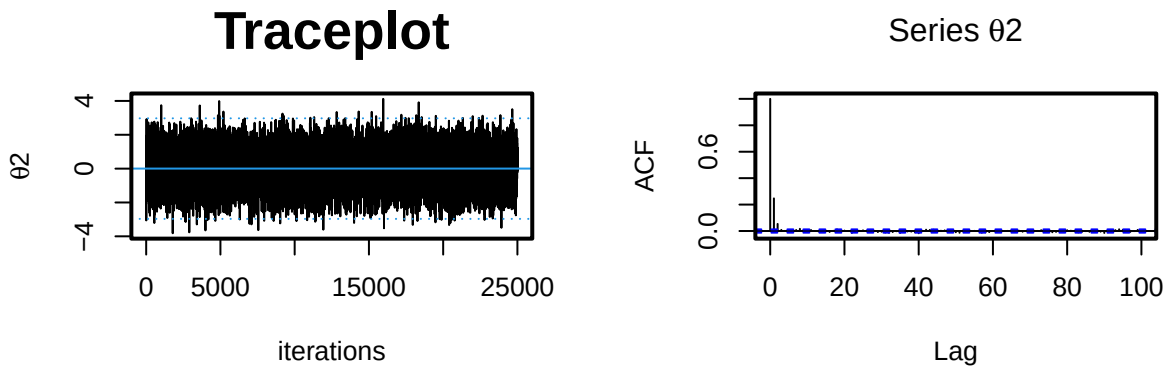
We generate samples with the Metropolis-Hastings sampler.

```
MH <- MH_sampler(t1=0, t2=0, burn_in=burn_in)
t1 <- gibbs$t1
t2 <- gibbs$t2
```

Finally the chain diagnostics are plotted for the MH samples as well.

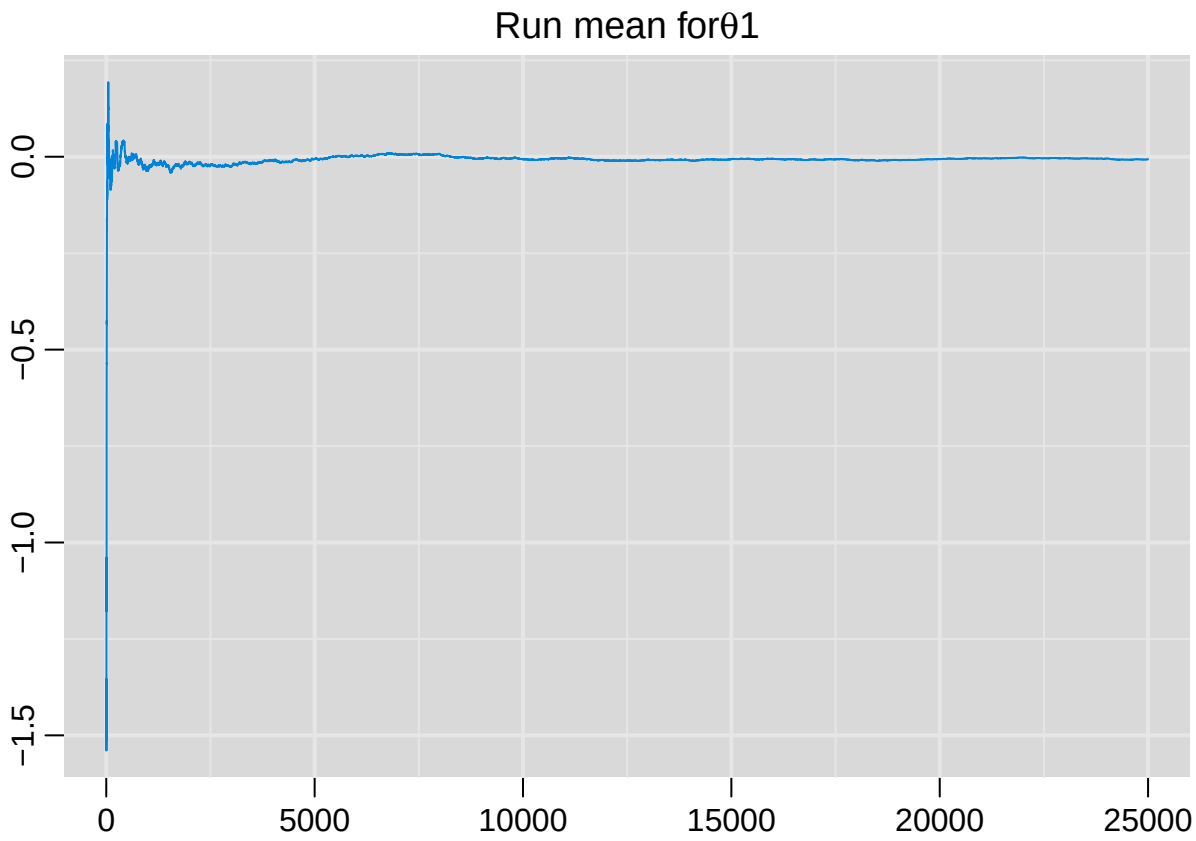
```
# trace plots
par(mfrow=c(2,2),mar=c(5,5,4,2))
plot(1:M,t1,type="l",ylim=c(min(t1,mean(t1)-3*sd(t1)),
      max(t1,mean(t1)+3*sd(t1))),xlab="iterations",
     ylab=expression(paste(theta,"1")),cex.main=2,main="Traceplot")
abline(h=mean(t1),lty=1,col=4);
abline(h=mean(t1)+3*sd(t1),lty=3,col=4)
abline(h=mean(t1)-3*sd(t1),lty=3,col=4); 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(theta,"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)
```





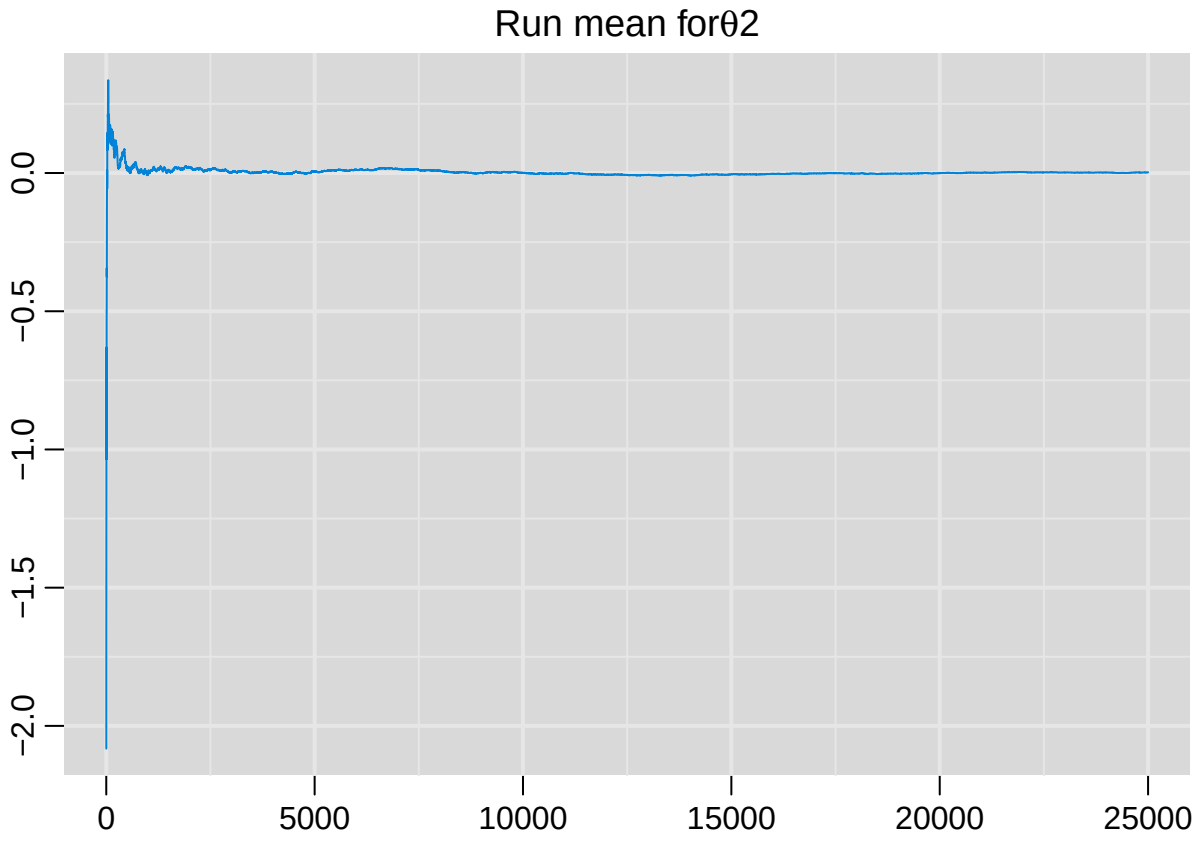

```
# histrograms of marginal posteriors
rmeanplot(t1,lwd=2,main=expression(paste("Run mean for",theta,"1")),mar=c(2,2,1.5,1)+ 0.1)
```

```
## Warning in rmeanplot(t1, lwd = 2, main = expression(paste("Run mean for", :
## Argument 'mcmcout' did not have valid variable names, so names have been
## created for you.
```

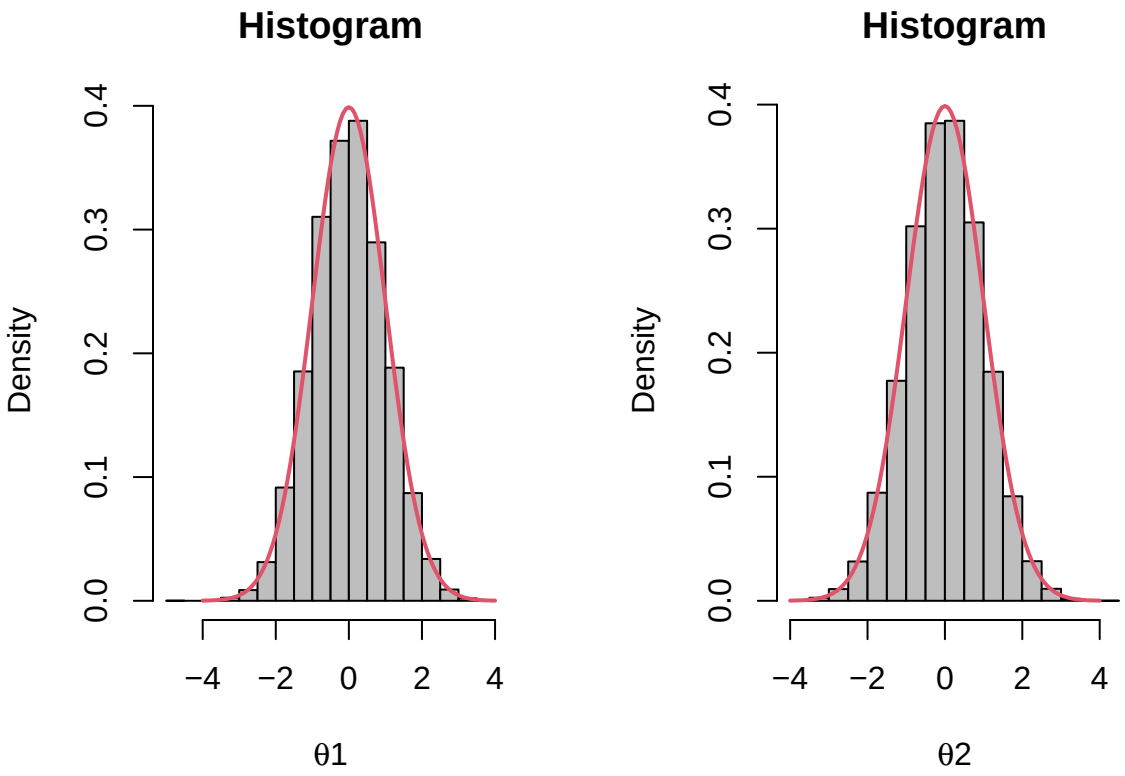



```
rmeanplot(t2,lwd=2,main=expression(paste("Run mean for",theta,"2")),mar=c(2,2,1.5,1)+ 0.1)
```

```
## Warning in rmeanplot(t2, lwd = 2, main = expression(paste("Run mean for", :
## Argument 'mcmcout' did not have valid variable names, so names have been
## created for you.
```




```
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)
```



As we can observe, for both the Gibbs and MH samplers the traceplots look good with a consistent range and efficient parameter space exploration. For the autocorrelation plots we can also see for both that the ACF decays very quickly in under 10 lags which indicates independent chain moves between successive iterations. In the run mean plot of both we also see that after the burn-in period the running mean quickly stabilizes and the plots show a stable flat line from then on. For the marginal posterior distributions we also see for both samplers that they are smooth and match the target distribution very well, e.g. a standard normal distribution.Overall it can be therefore said that both samplers worked well.