Metropolis-Hastings

Seung Ah Ha, Jaymo Kim, Wonbin Song

Metropolis-Hastings

Our target distribution is Beta(6,4) and our proposal distribution is $\phi_{prop}|\phi_{old} \sim Beta(c\phi_{old},\ c(1-\phi_{old}))$, for some constant c. First, we implement several functions which will be used in calculating posterior distribution. Here, we define likelihood function which gives the log-likelihood value of the parameter, and also prior function that gives the log density value of the parameter. The density used in the likelihood function is Beta(6,4) since it is our target distribution, and the density used in the prior function is Uniform[0,1] since our start value is from Uniform[0,1]. In addition, since the parameters in Beta distribution have to be larger than 0, for $c > 0,\ \phi < 1$.

```r
a <- 6
b <- 4

likelihood <- function(param){
  singlelikelihoods = dbeta(param, a, b, log = T)
  return(singlelikelihoods)
}

prior <- function(param){
  pr = dunif(param, min=0, max=1, log = T)
  return(pr)
}

posterior <- function(param){
  return (exp(likelihood(param) + prior(param)))
}
```

Posterior function can be calculated by multiplying likelihood function and prior function. However, we used log in the likelihood function and the prior function, so we need to convert them into $exp(likelihood + prior)$ in the posterior function.

```r
metropolis_MCMC <- function(startvalue, iterations, c){
  chain = array(dim = c(iterations+1,1))
  chain[1,] = startvalue
  for (i in 1:iterations){
    proposal = rbeta(1, c*chain[i,], c*(1-chain[i,]))
    probab = (posterior(proposal)*dbeta(chain[i,],c*proposal,c*(1-proposal)))/
      (posterior(chain[i,])*dbeta(proposal, c*chain[i,], c*(1-chain[i,])))
    if (runif(1) < probab){
      chain[i+1,] = proposal
    }else{
      chain[i+1,] = chain[i,]
    }
  }
  return(chain)
}
```

After implementing all the functions that are required to obtain a chain, then we can finally define metropolis_MCMC function that can yield a chain. Note that

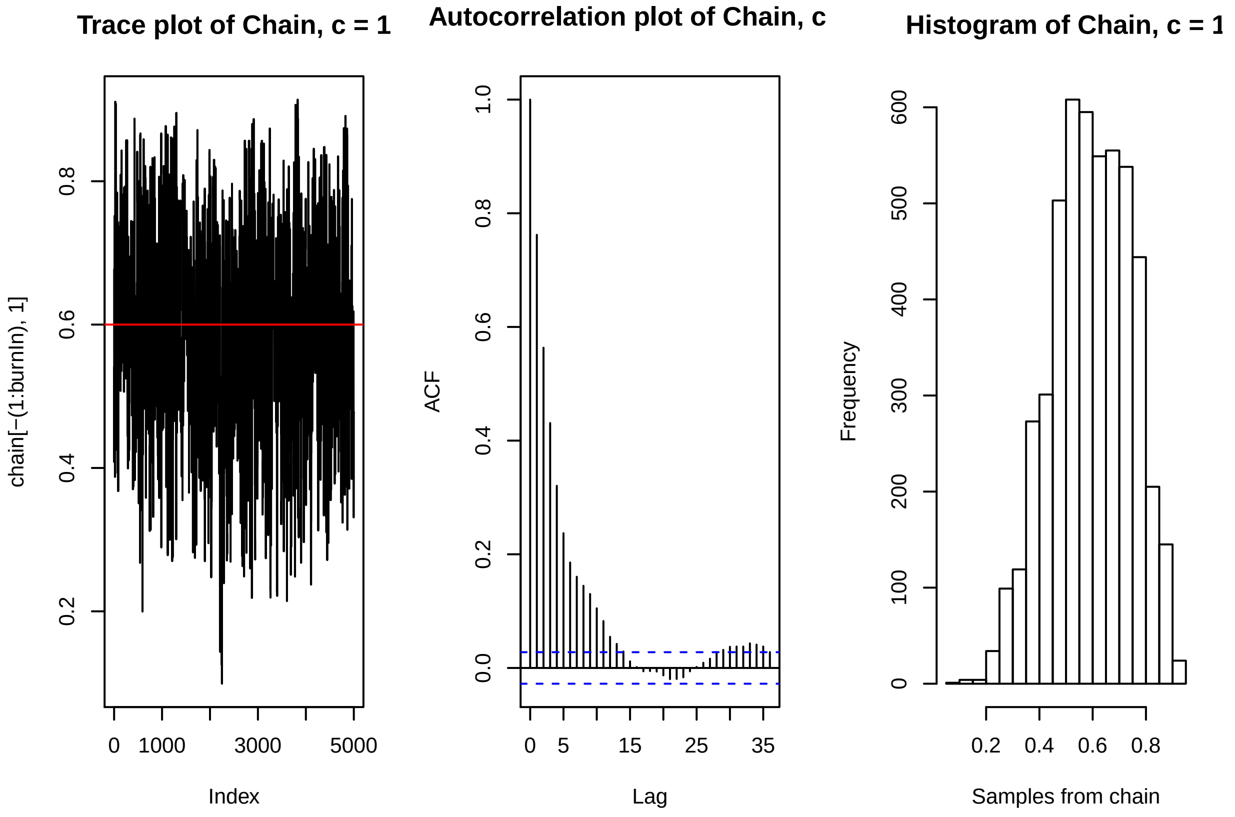
$$\phi_{new} = \phi_{proposal} \;\; with \;\; probability \;\; \alpha(\phi_{old}, \phi_{new})$$
$$= \phi_{old} \;\; with \;\; probaility \;\; 1 - \alpha(\phi_{old}, \phi_{new})$$

where $\alpha(\phi_{old}, \phi_{new}) = min(1, \frac{p(\phi_{new})q(\phi_{old}|\phi_{new})}{p(\phi_{old})q(\phi_{new}|\phi_{old})})$. Here, $p(\phi)$ is the posterior function, and $q(\phi_{new}|\phi_{old})$ is our proposal functions that follows $Beta(c\phi_{old}, \;\; c(1 - \phi_{old}))$. Since Beta distribution is asymmetric, the probability form cannot be simplified to the ratio of posterior functions. Hence, we need to calculate the probabiltiy by using posterior functions and proposal functions.

Next, we can finally get the chain using metropolis_MCMC function with the startvalue generated from Uniform(0,1). We also generate random values from Beta(6,4) for the comparison.

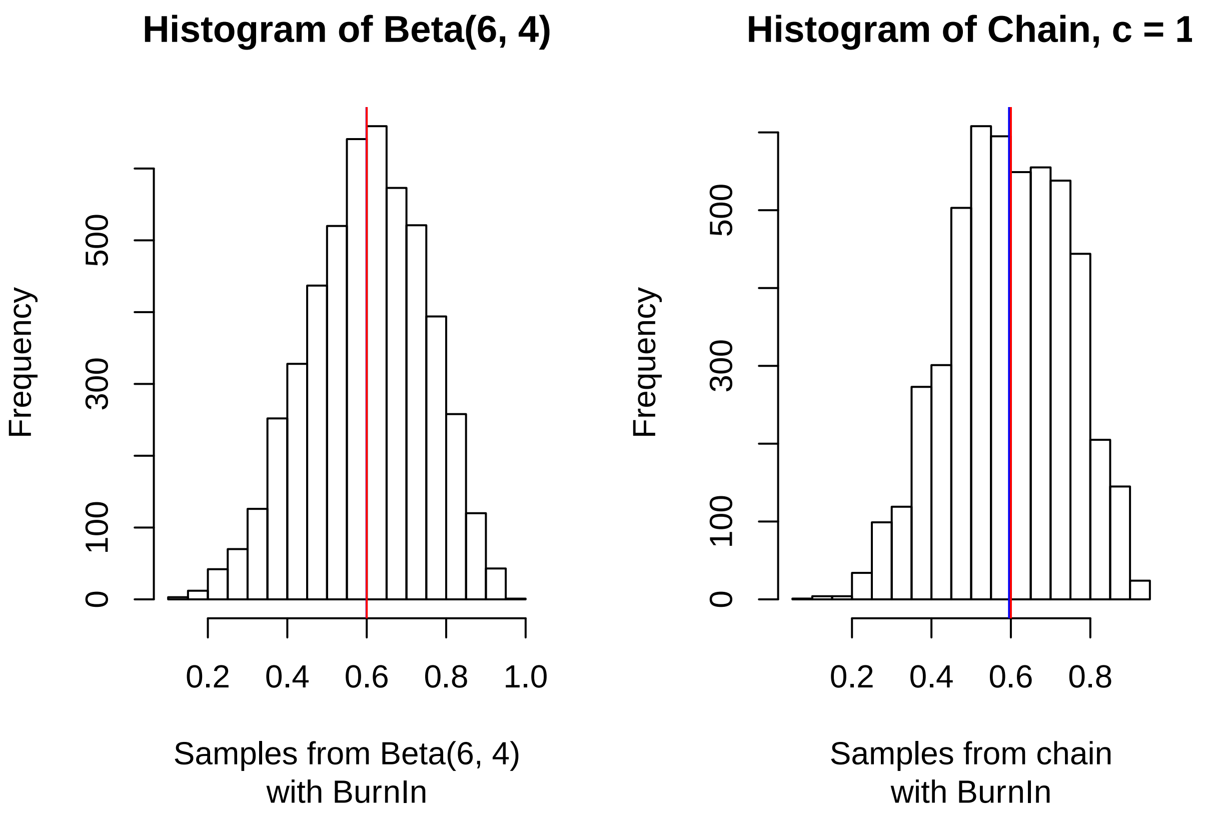
```r
startvalue <- runif(1, 0, 1)
chain=metropolis_MCMC(startvalue, 10000, c=1)
test <- rbeta(10000, 6, 4)
```

```r
# Performance of the sampler: c = 1 with BurnIn
par(mfrow=c(1,3))
test2 <- test[5001:10000]
burnIn = 5000
acceptance = 1-mean(duplicated(chain[-(1:burnIn),]))
plot(chain[-(1:burnIn),1], type='l', main = "Trace plot of Chain, c = 1")
abline(h=0.6, col="red")
acf(chain[-(1:burnIn),1], main = "Autocorrelation plot of Chain, c = 1")
hist(chain[-(1:burnIn),1], xlab = "Samples from chain", main = "Histogram of Chain, c = 1")
```



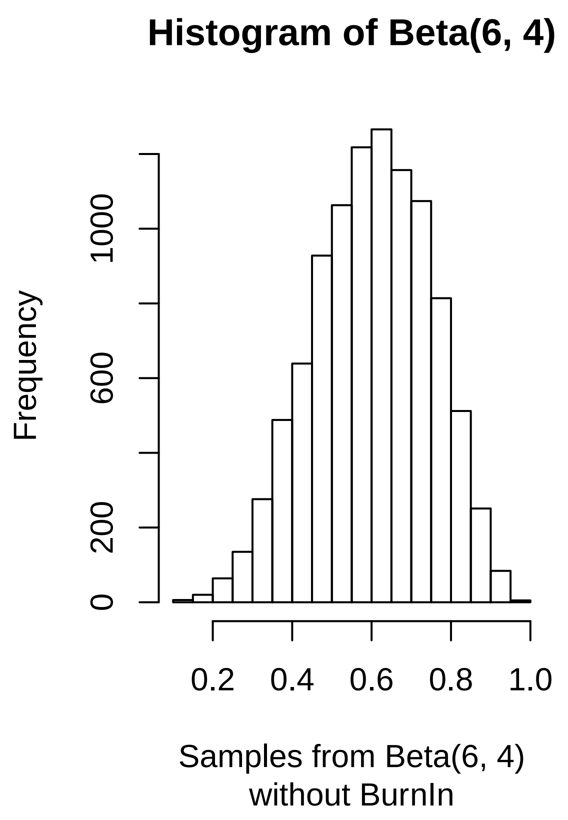
Here, we evaluate the performance of the sampler when c = 1 without burn in. The first plot is the trace plot, showing the trace of our chain. The specific explanation of those plots will be covered at the last part.

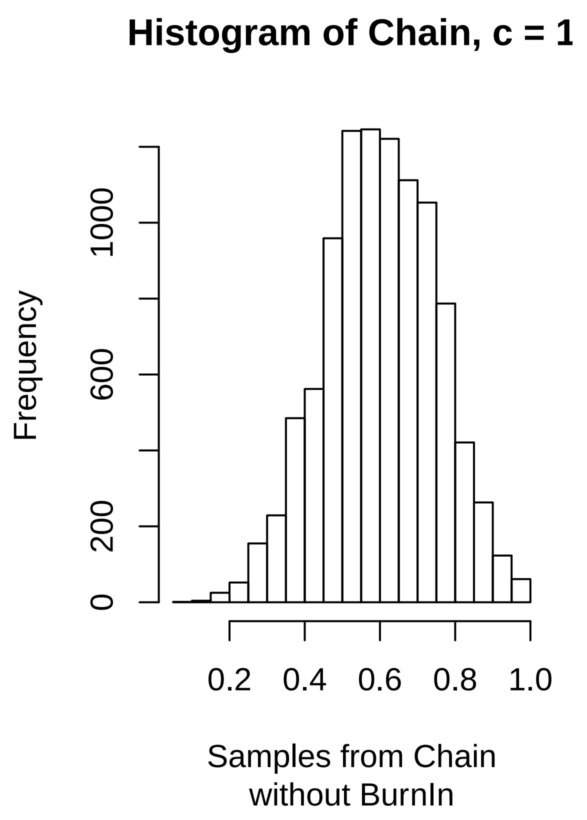
```r
# Histogram of the target distribution Beta(6,4) vs. the Chain with burnIn
par(mfrow=c(1,2))
hist(test2, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "with BurnIn")
abline(v=mean(test2), col="blue")
abline(v=0.6, col="red")
hist(chain[-(1:burnIn),1], xlab = "Samples from chain", main = "Histogram of Chain, c = 1"
     , sub = "with BurnIn")
abline(v = mean(chain[-(1:burnIn),1]), col="blue")
abline(v=0.6, col="red")
```



In addition, we can compare the target distribution Beta(6,4) with the histogram of the chain as we can see above. The red line indicates the real mean of the distribution, which is $\frac{6}{4+6} = 0.6$. We can check that both histogram looks similar and the mean of the chain is `0.5954081` which is very close to 0.6. This is after we remove some redundant moves from the chain using BurnIn. However, we can still compare the distribution of Beta(6,4) with the whole chain without BurnIn.

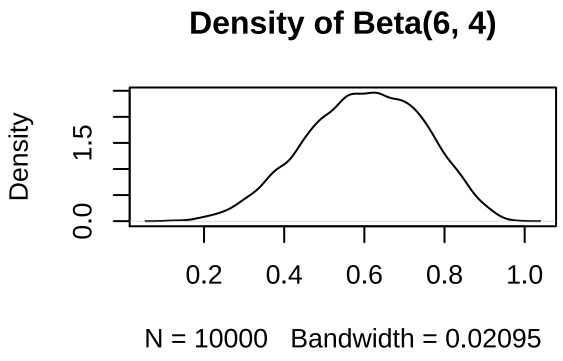
```r
# Histogram of the chain vs. the target distribution Beta(6,4) without BurnIn
par(mfrow=c(1,2))
hist(test, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "without BurnIn")

hist(chain, xlab = "Samples from Chain", main = "Histogram of Chain, c = 1",
     sub = "without BurnIn")
```

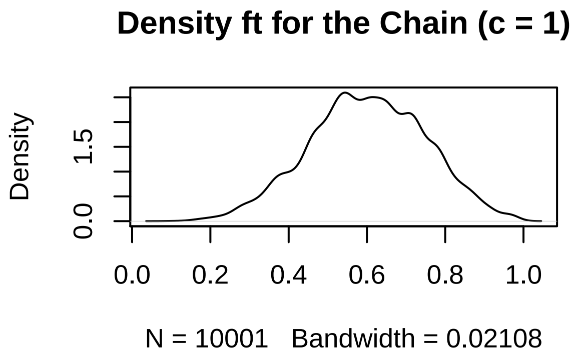


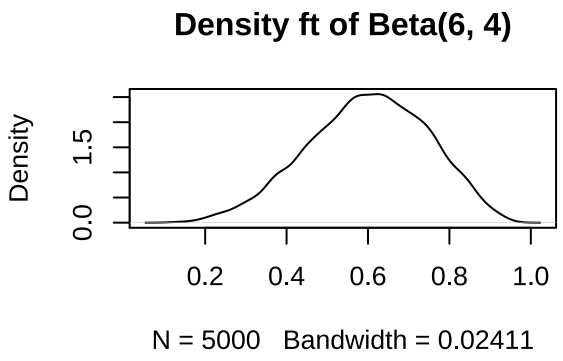


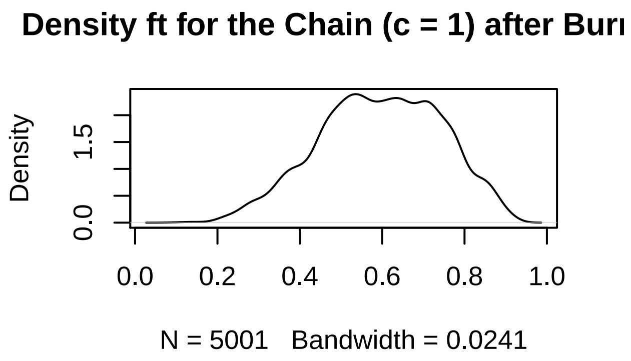
If we check the difference between the distributions with and without BurnIn. As we can see in the plot, the distribution after BurnIn better suits the target distribution.

```r
par(mfrow = c(2,2))
plot(density(test), main = "Density of Beta(6, 4)")
plot(density(chain), main = "Density ft for the Chain (c = 1)")
plot(density(test2), main = "Density ft of Beta(6, 4)")
plot(density(chain[-(1:burnIn),1]), main = "Density ft for the Chain (c = 1) after BurnIn")
```









Finally, we can implement Kolmogorov-Smirnov test. Accordking to the result, we cannot say that the chain follows Beta(6, 4) distribution since p-value is smaller than 0.05.

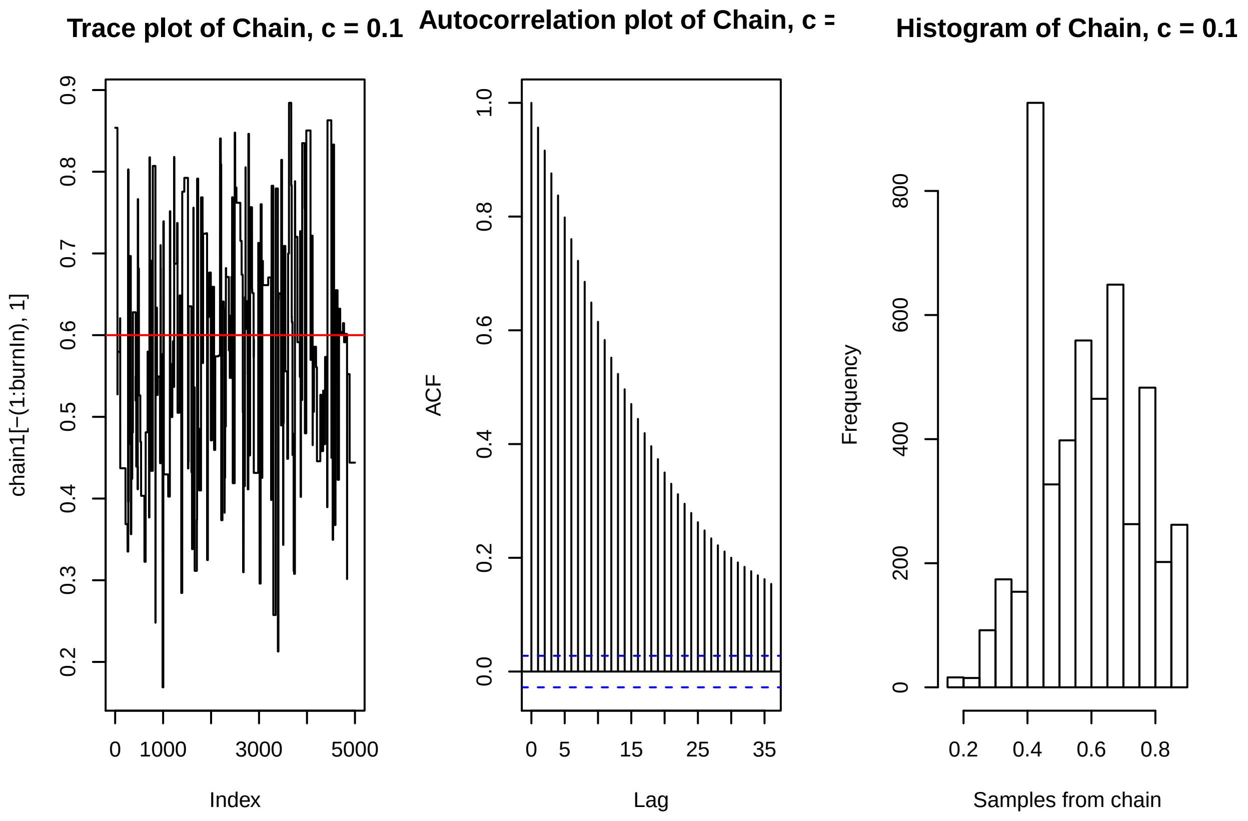
```
# Kolmogorov-Smirnov Test
chainks <- sort(chain[-(1:burnIn),1])
ks.test(chainks, "pbeta", 6, 4)
```

```
## Warning in ks.test(chainks, "pbeta", 6, 4): ties should not be present for
## the Kolmogorov-Smirnov test
```

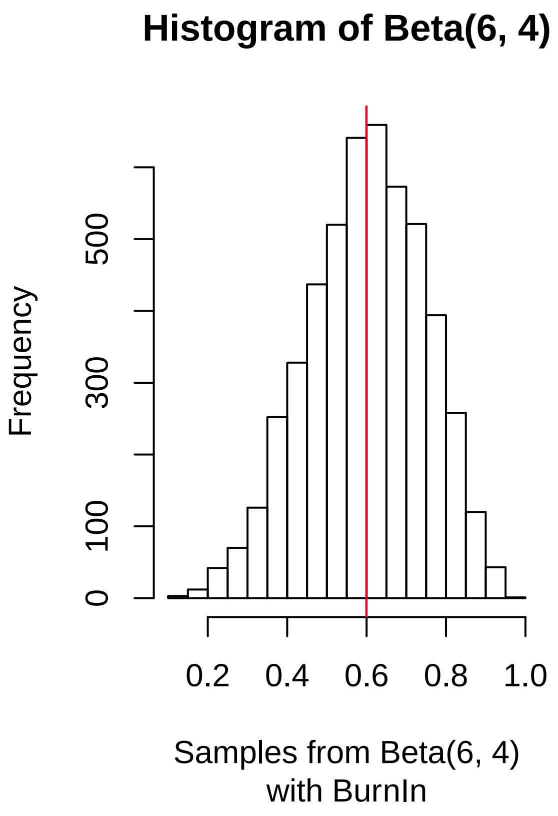
```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  chainks
## D = 0.035345, p-value = 7.491e-06
## alternative hypothesis: two-sided
```

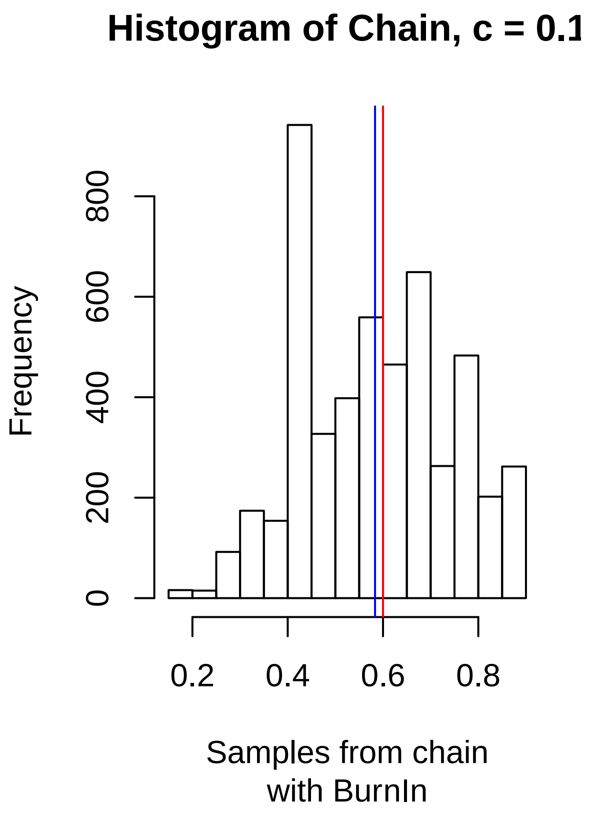
Next, we repeat the same procedure when c = 0.1, c = 2.5 and c = 10.

```
set.seed(1)
# Performance of the sampler: c = 0.1
chain1=metropolis_MCMC(startvalue, 10000, c=0.1)
par(mfrow=c(1,3))
acceptance1 = 1-mean(duplicated(chain1[-(1:burnIn),]))
plot(chain1[-(1:burnIn),1], type='l', main = "Trace plot of Chain, c = 0.1")
abline(h=0.6, col="red")
acf(chain1[-(1:burnIn),1], main = "Autocorrelation plot of Chain, c = 0.1")
hist(chain1[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 0.1")
```

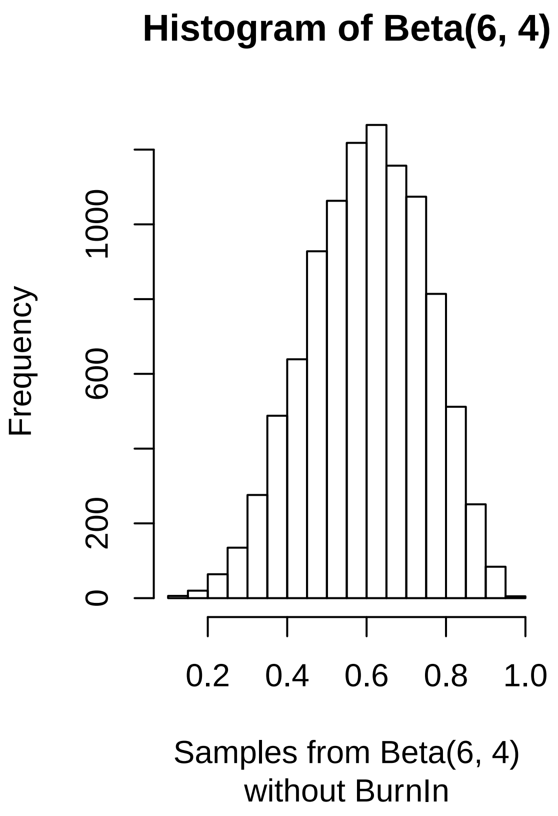


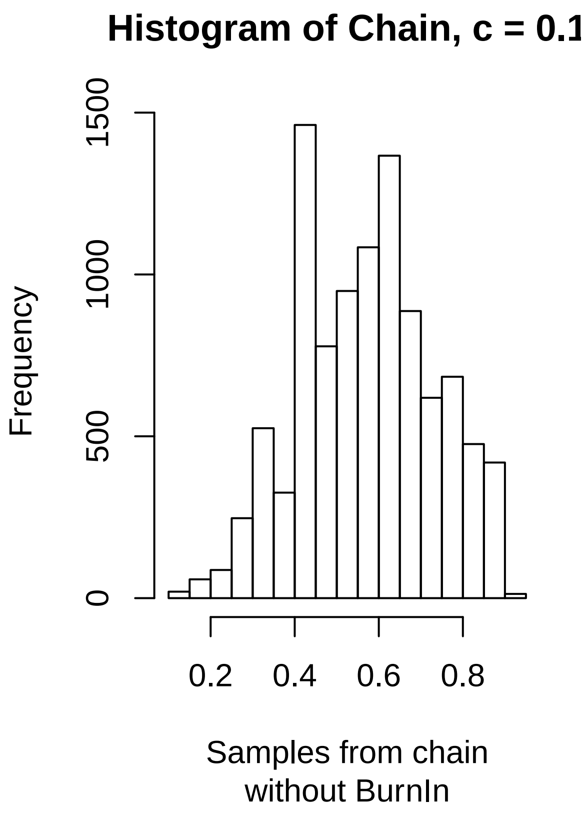
```r
# Histogram of the target distribution Beta(6,4) vs. the Chain after burnIn
par(mfrow=c(1,2))
hist(test2, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "with BurnIn")
abline(v=mean(test2), col="blue")
abline(v=0.6, col="red")
hist(chain1[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 0.1", sub = "with BurnIn")
abline(v = mean(chain1[-(1:burnIn),1]), col="blue")
abline(v=0.6, col="red")
```



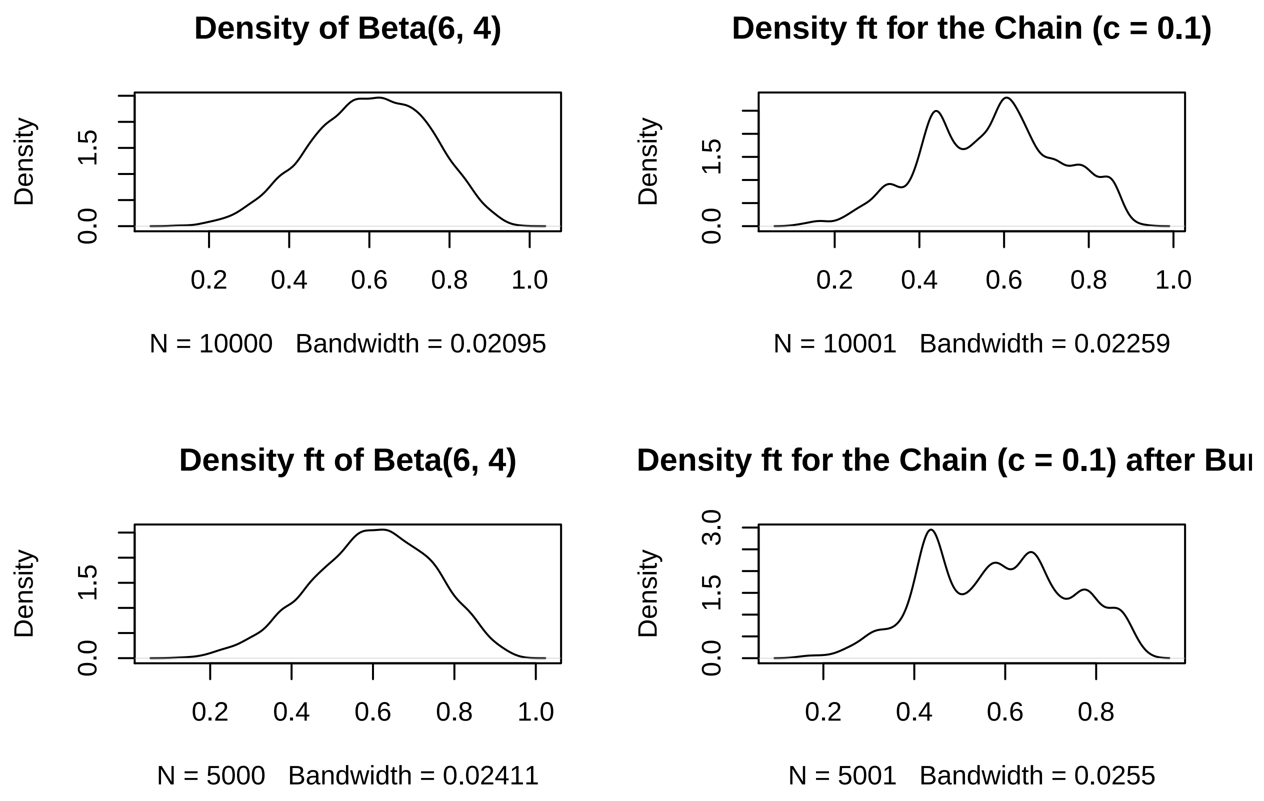


```
# Histogram of the chain vs. the target distribution Beta(6,4) without BurnIn
par(mfrow=c(1,2))
hist(test, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "without BurnIn")
hist(chain1, xlab = "Samples from chain", main = "Histogram of Chain, c = 0.1",
     sub = "without BurnIn")
```





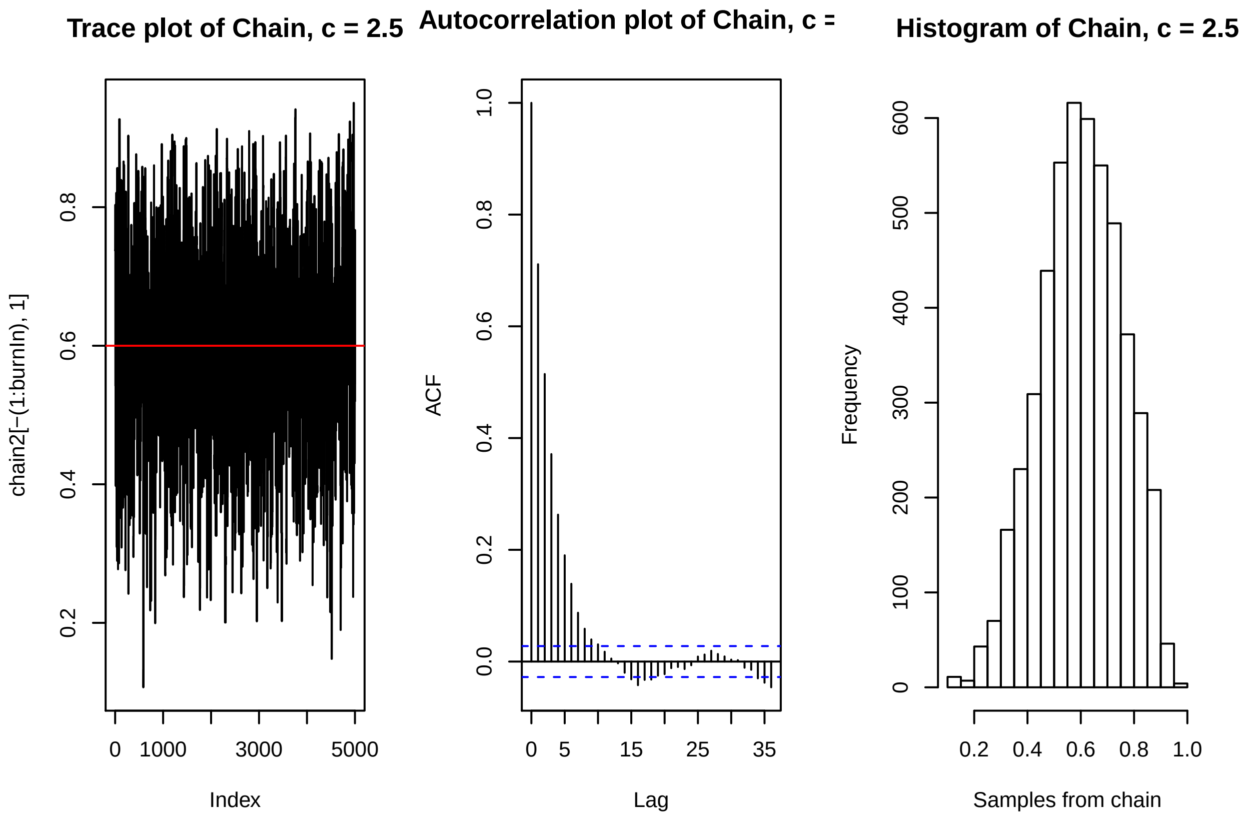
```
par(mfrow = c(2,2))
plot(density(test), main = "Density of Beta(6, 4)")
plot(density(chain1), main = "Density ft for the Chain (c = 0.1)")
plot(density(test2), main = "Density ft of Beta(6, 4)")
plot(density(chain1[-(1:burnIn),1]), main = "Density ft for the Chain (c = 0.1) after BurnIn")
```



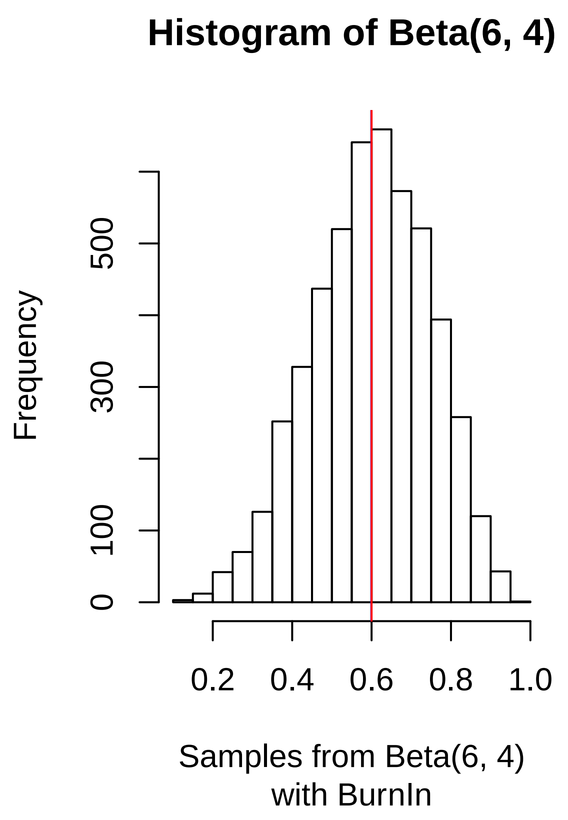
```
# Kolmogorov-Smirnov statistic
chain1ks <- sort(chain1[-(1:burnIn),1])
ks.test(chain1ks, "pbeta", 6, 4)
```

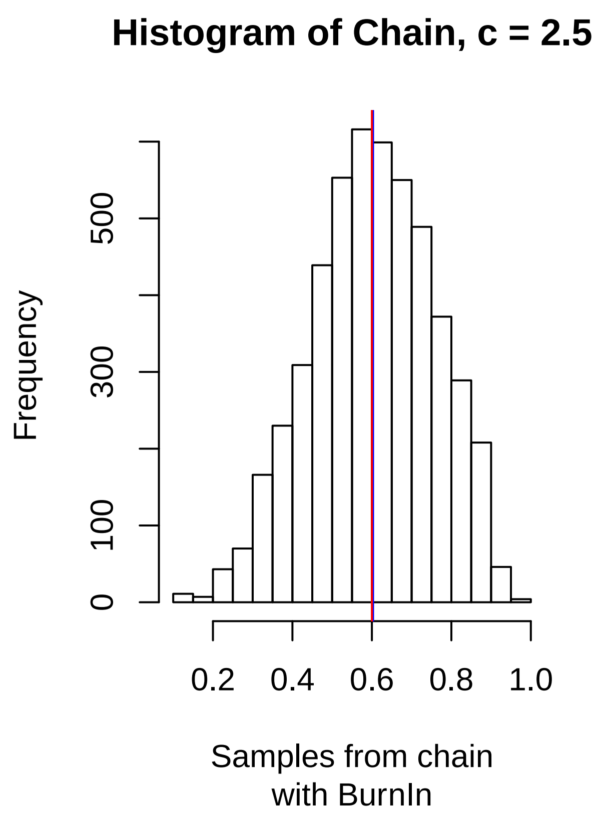
```
## Warning in ks.test(chain1ks, "pbeta", 6, 4): ties should not be present for
## the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  chain1ks
## D = 0.11379, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
set.seed(2)
# Performance of the sampler: c = 2.5
chain2=metropolis_MCMC(startvalue, 10000, c=2.5)
par(mfrow=c(1,3))
acceptance2 = 1-mean(duplicated(chain2[-(1:burnIn),]))
plot(chain2[-(1:burnIn),1], type='l', main = "Trace plot of Chain, c = 2.5")
abline(h=0.6, col="red")
acf(chain2[-(1:burnIn),1], main = "Autocorrelation plot of Chain, c = 2.5")
hist(chain2[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 2.5")
```

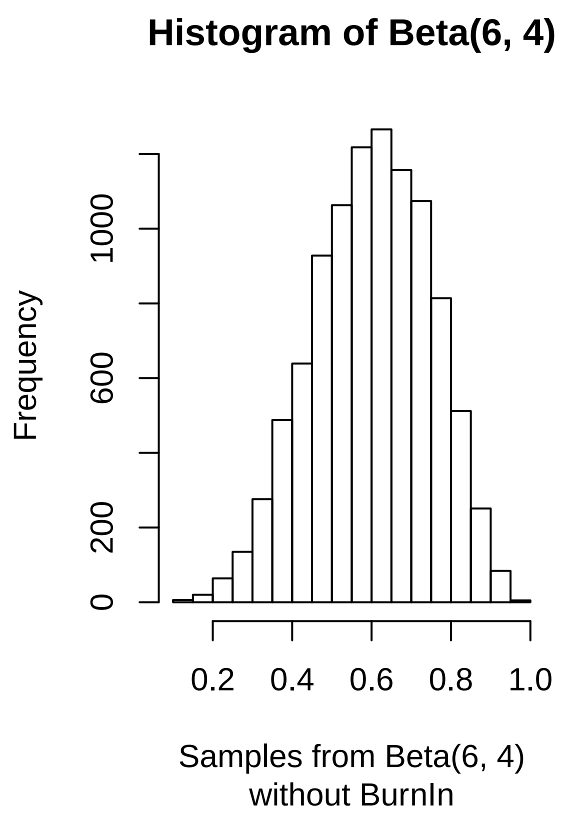


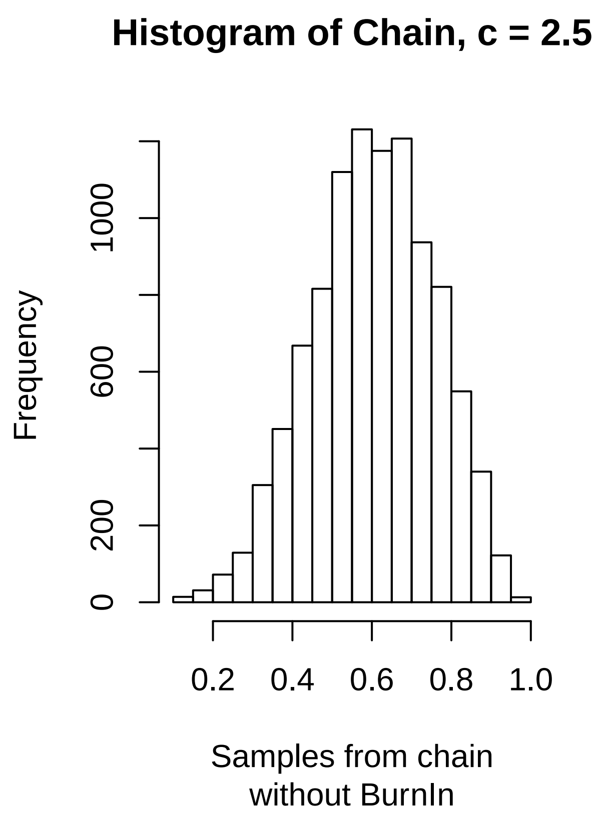
```r
# Histogram of the target distribution Beta(6,4) vs. the Chain after burnIn
par(mfrow=c(1,2))
hist(test2, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "with BurnIn")
abline(v=mean(test2), col="blue")
abline(v=0.6, col="red")
hist(chain2[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 2.5", sub = "with BurnIn")
abline(v = mean(chain2[-(1:burnIn),1]), col="blue")
abline(v=0.6, col="red")
```



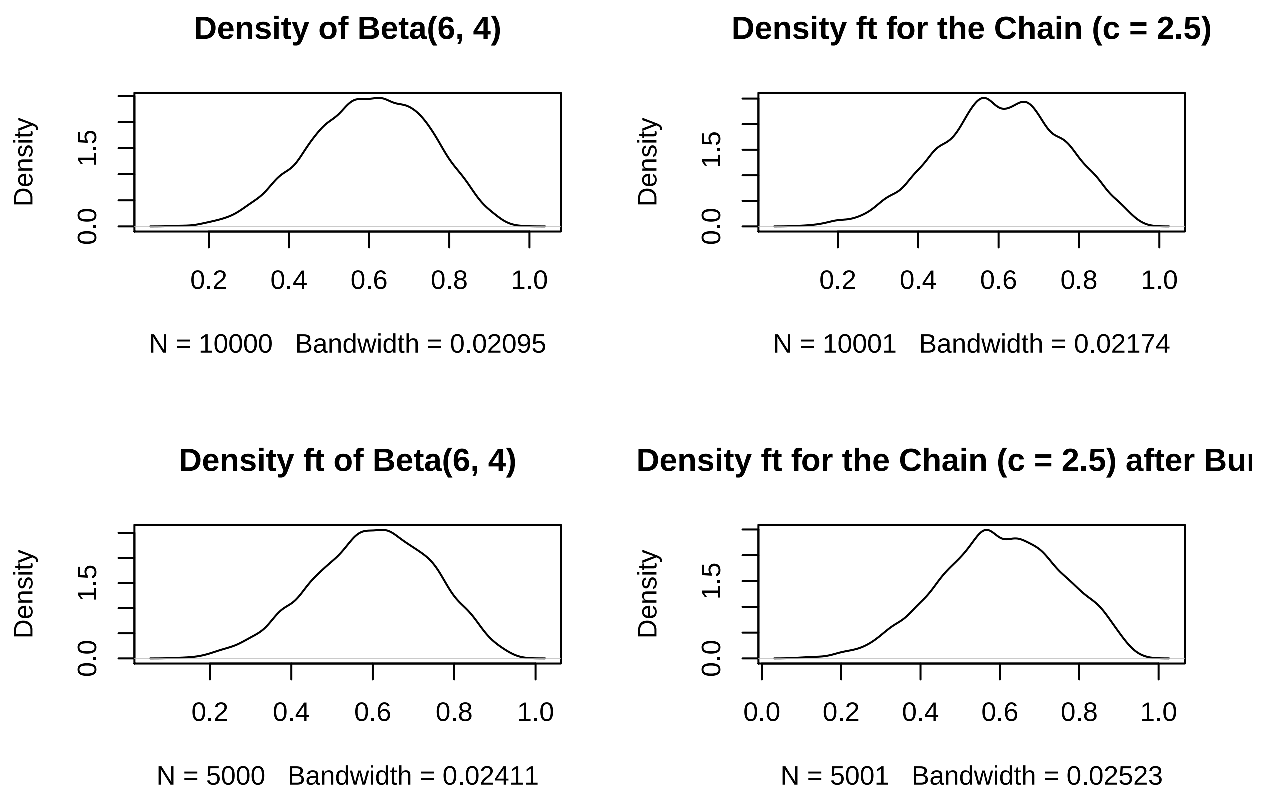


```r
# Histogram of the chain vs. the target distribution Beta(6,4) without BurnIn
hist(test, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "without BurnIn")
hist(chain2, xlab = "Samples from chain", main = "Histogram of Chain, c = 2.5",
     sub = "without BurnIn")
```





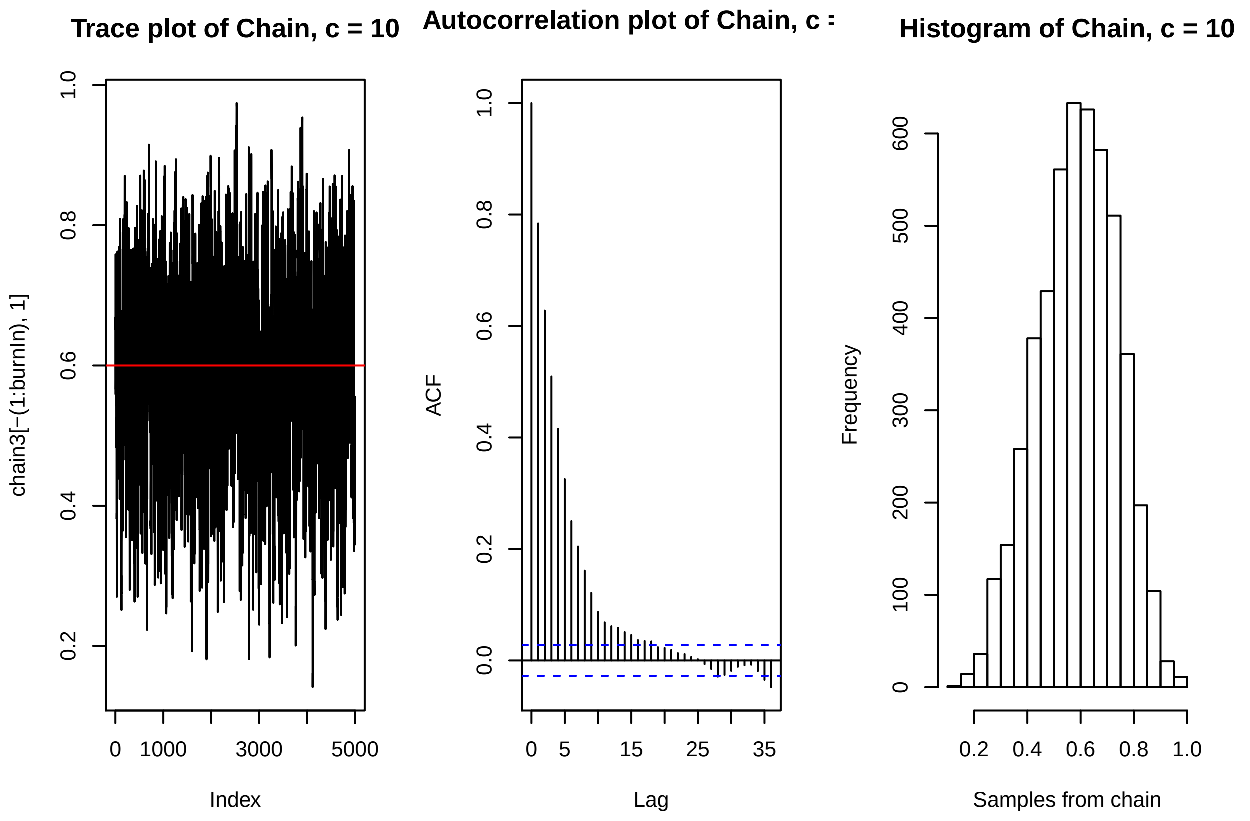
```
par(mfrow = c(2,2))
plot(density(test), main = "Density of Beta(6, 4)")
plot(density(chain2), main = "Density ft for the Chain (c = 2.5)")
plot(density(test2), main = "Density ft of Beta(6, 4)")
plot(density(chain2[-(1:burnIn),1]), main = "Density ft for the Chain (c = 2.5) after BurnIn")
```



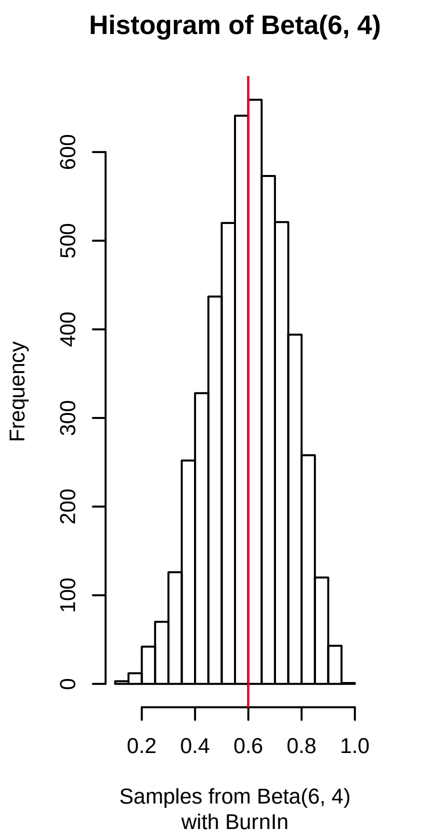
```r
# Kolmogorov-Smirnov statistic
chain2ks <- sort(chain2[-(1:burnIn),1])
ks.test(chain2ks, "pbeta", 6, 4)
```

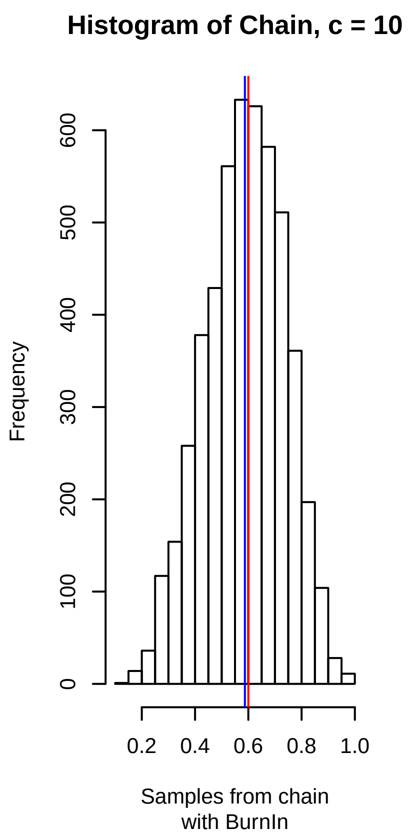
```
## Warning in ks.test(chain2ks, "pbeta", 6, 4): ties should not be present for
## the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  chain2ks
## D = 0.03107, p-value = 0.0001282
## alternative hypothesis: two-sided
```

```r
set.seed(3)
# Performance of the sampler: c = 10
chain3=metropolis_MCMC(startvalue, 10000, c=10)
par(mfrow=c(1,3))
acceptance3 = 1-mean(duplicated(chain3[-(1:burnIn),]))
plot(chain3[-(1:burnIn),1], type='l', main = "Trace plot of Chain, c = 10")
abline(h=0.6, col="red")
acf(chain3[-(1:burnIn),1], main = "Autocorrelation plot of Chain, c = 10")
hist(chain3[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 10")
```

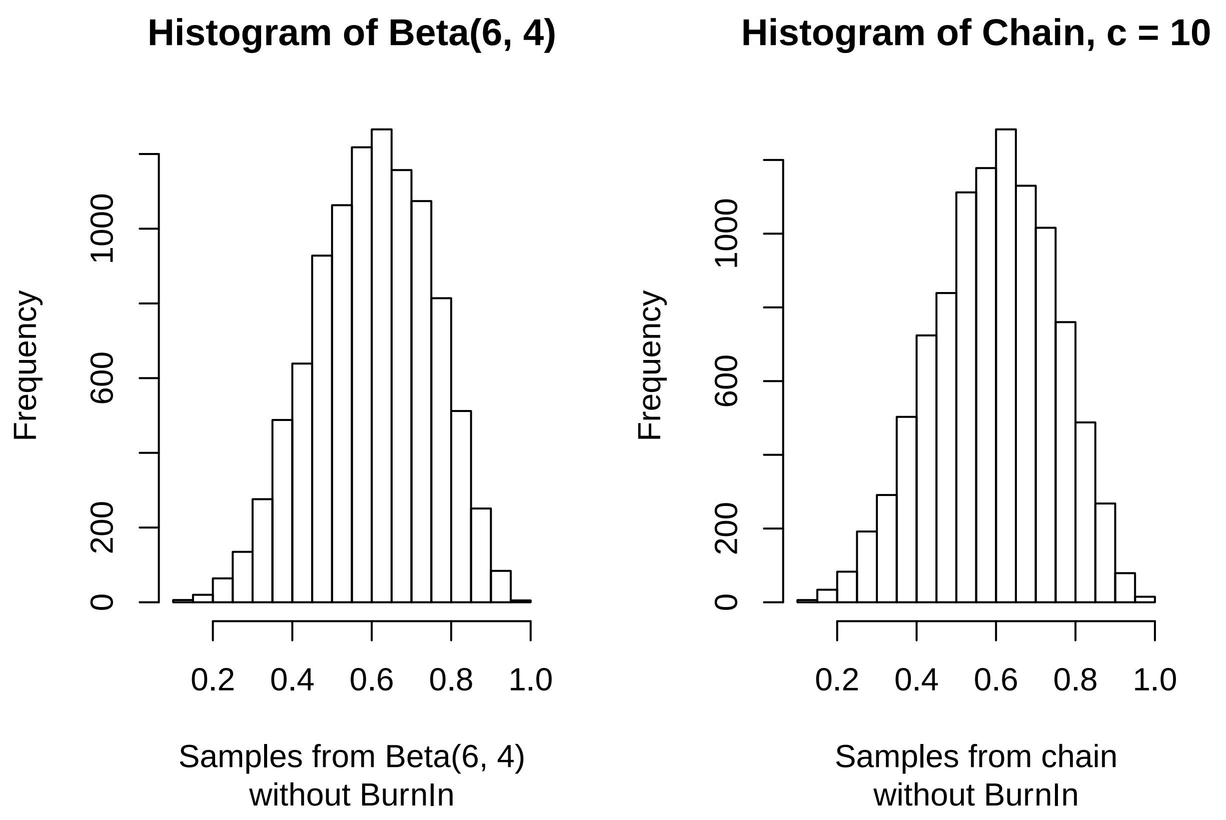


```
# Histogram of the target distribution Beta(6,4) vs. the Chain after burnIn
hist(test2, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "with BurnIn")
abline(v=mean(test2), col="blue")
abline(v=0.6, col="red")
hist(chain3[-(1:burnIn),1], xlab = "Samples from chain",
     main = "Histogram of Chain, c = 10", sub = "with BurnIn")
abline(v = mean(chain3[-(1:burnIn),1]), col="blue")
abline(v=0.6, col="red")

# Histogram of the chain vs. the target distribution Beta(6,4) without BurnIn
par(mfrow=c(1,2))
```

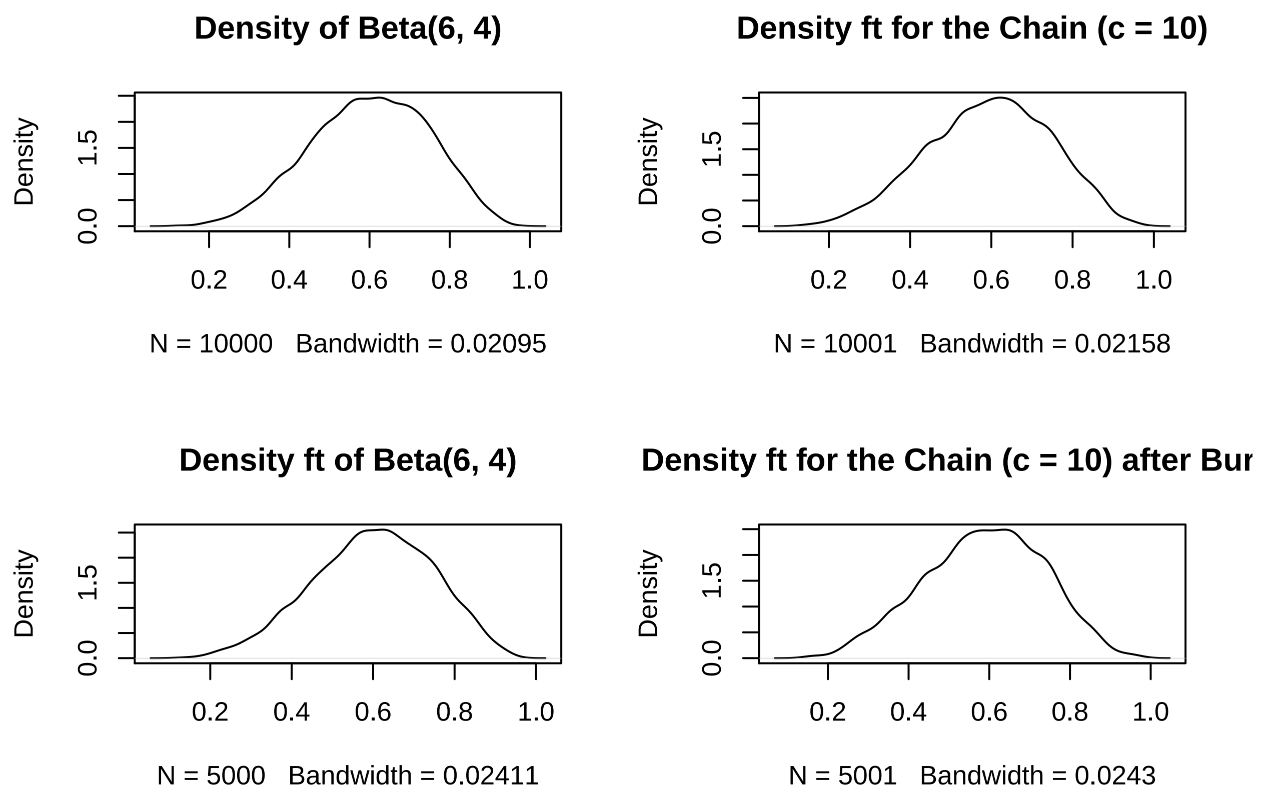




```
hist(test, xlab = "Samples from Beta(6, 4)", main = "Histogram of Beta(6, 4)",
     sub = "without BurnIn")
hist(chain3, xlab = "Samples from chain", main = "Histogram of Chain, c = 10",
     sub = "without BurnIn")
```



```r
par(mfrow = c(2,2))
plot(density(test), main = "Density of Beta(6, 4)")
plot(density(chain3), main = "Density ft for the Chain (c = 10)")
plot(density(test2), main = "Density ft of Beta(6, 4)")
plot(density(chain3[-(1:burnIn),1]), main = "Density ft for the Chain (c = 10) after BurnIn")
```



```
# Kolmogorov-Smirnov statistic
chain3ks <- sort(chain3[-(1:burnIn),1])
ks.test(chain3ks, "pbeta", 6, 4)
```

```
## Warning in ks.test(chain3ks, "pbeta", 6, 4): ties should not be present for
## the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  chain3ks
## D = 0.036098, p-value = 4.372e-06
## alternative hypothesis: two-sided
```

```
acceptance
```

```
## [1] 0.2591482
```

```
acceptance1
```

```
## [1] 0.04219156
```

```
acceptance2
```

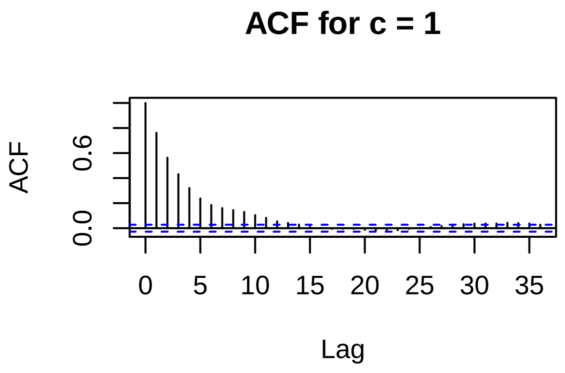
```
## [1] 0.4167167
```

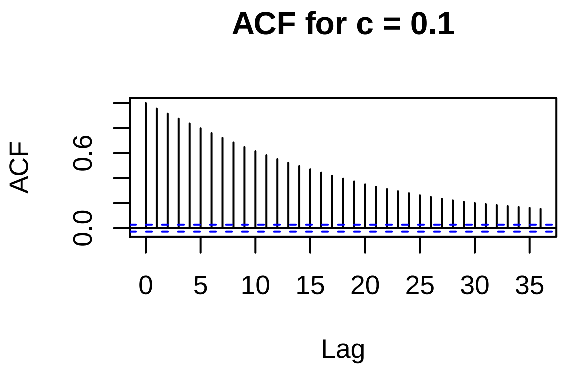
```
acceptance3
```

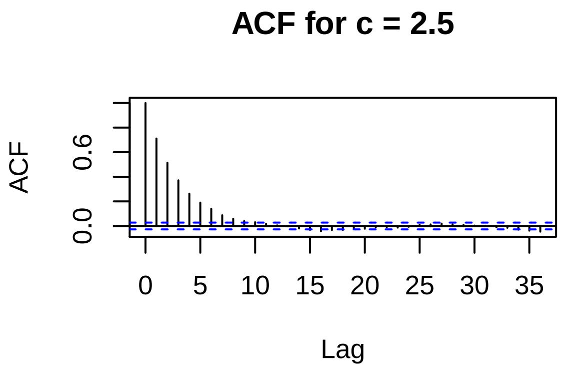
```
## [1] 0.6752649
```

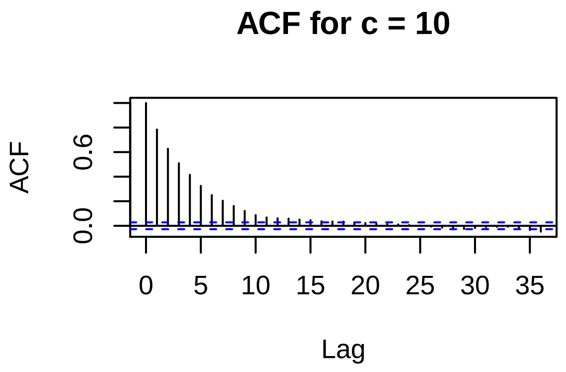
```
par(mfrow=c(2,2))
acf(chain[-(1:burnIn),1], main = "ACF for c = 1")
acf(chain1[-(1:burnIn),1], main = "ACF for c = 0.1")
```

```r
acf(chain2[-(1:burnIn),1], main = "ACF for c = 2.5")
acf(chain3[-(1:burnIn),1], main = "ACF for c = 10")
```









As we can see in the result, as c increases, acceptance rate increases. This is because the proposal distribution $\phi_{prop}|\phi_{old} \sim Beta(c\phi_{old},\ c(1-\phi_{old}))$ is getting narrower compare to the target distribution as c increases. That is, for c = 0.1, the proposal distribution is wider than the target distribution. In addition, according to the trace plot for c = 0.1, we can check that the chain tends to stay at same place more than when c = 2.5 and c = 10. This implies that c has lower acceptance rate. Next, if we look at the autocorrelation plot, we can check that there exists high autocorrelation in the chain, and that causes the chain to stay where it was. For c = 2.5 and c = 10, we can see in the autocorrelation has dropped faster than in c = 0.1, so it implies that the chain tends to move faster as we can check in the trace plot. This indicates that the acceptance rate is higher than c = 0.1 for both cases. However, as we can see in the acceptance rates of 3 cases, the acceptance rate when c = 10 is the highest, implying that the chain for c = 10 moves frequently than when c = 2.5. However, very high acceptance rate is not preferrable, and optimal acceptance rate is 20~30%, as we obtain in c = 1 (which is `0.2591482`). If we compare among c = 0.1, 2.5, 10, the case when c = 2.5 seems to be the most optimal. This is comparable to the Kolmogorov-Smirnov test p-value, since among those three cases, p-value in c = 2.5 is the highest, implying that it is less likely to reject the null hypothesis that the chain distribution has the same distribution as the target distribution, Beta(6,4). (However, the results of all cases show that they reject the null hypothesis.)