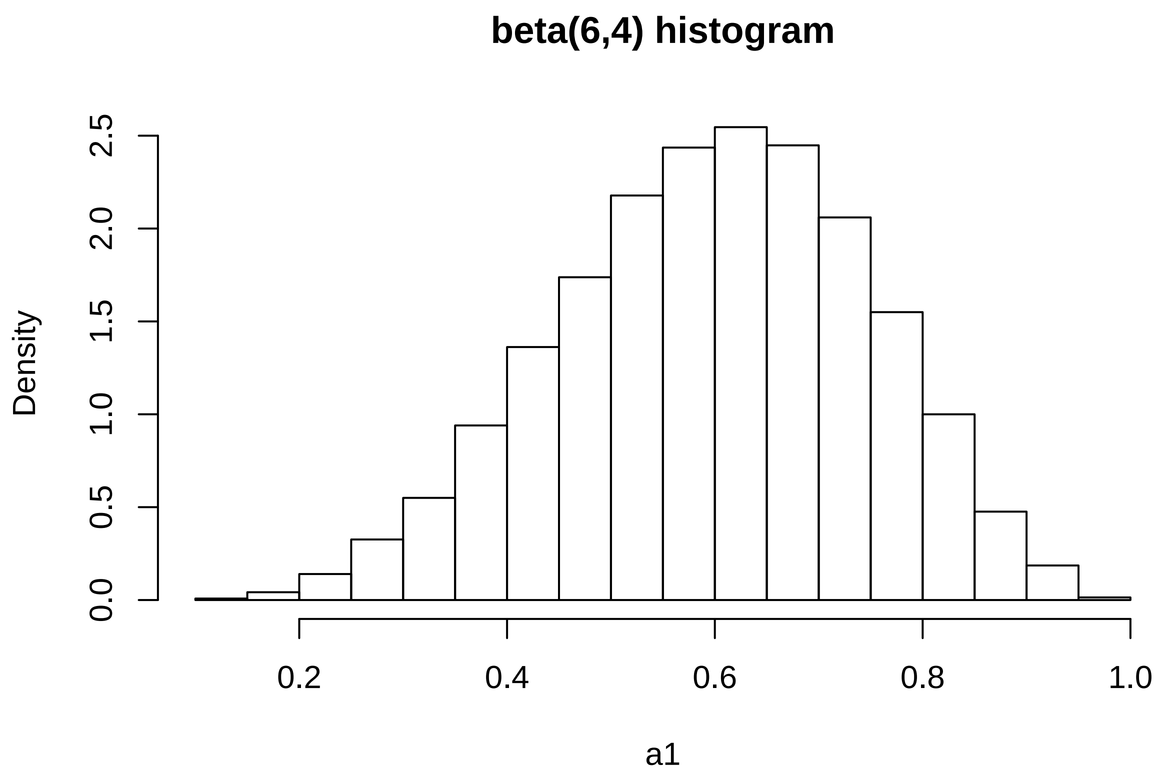
Metropolis-Hastings for Beta distribution

We write a Metropolis-Hastings algoritm to generate X~beta(6,4). We select a r.v. from unif(0,1) as a starting value and a Beta jumping distribution
where c is a parameter in our function. Later on we can test MH algoritm with differnt c's 10 percent of iterations will be treated as burn-in

```
MHalg<-function(iter,c){

  phi<-  c()
  phi[1]<-runif(1,0,1)#generate a starting value from unif(0,1)

  new_phi<- function(phi) {
    proposal_phi <- rbeta(1, shape1= c*phi , shape2= c*(1-phi) ) #giving jumping dist.
    # ratio of accepting proposal_phi
    r <- (dbeta(proposal_phi, shape1 = 6, shape2 = 4)/
          dbeta(proposal_phi, shape1= c*phi , shape2=c*(1-phi)))/
         (dbeta(phi, shape1 = 6, shape2 = 4)/
          dbeta(phi, shape1= c*proposal_phi, shape2=c*(1-proposal_phi)))

    if (runif(1) <= r)
      return(proposal_phi)
    else
      return(phi)
  }

  for (i in 1:iter)
    phi[i+1]= new_phi(phi[i])
  phi=phi[-(iter/10)] #truncate burn-in

}
```

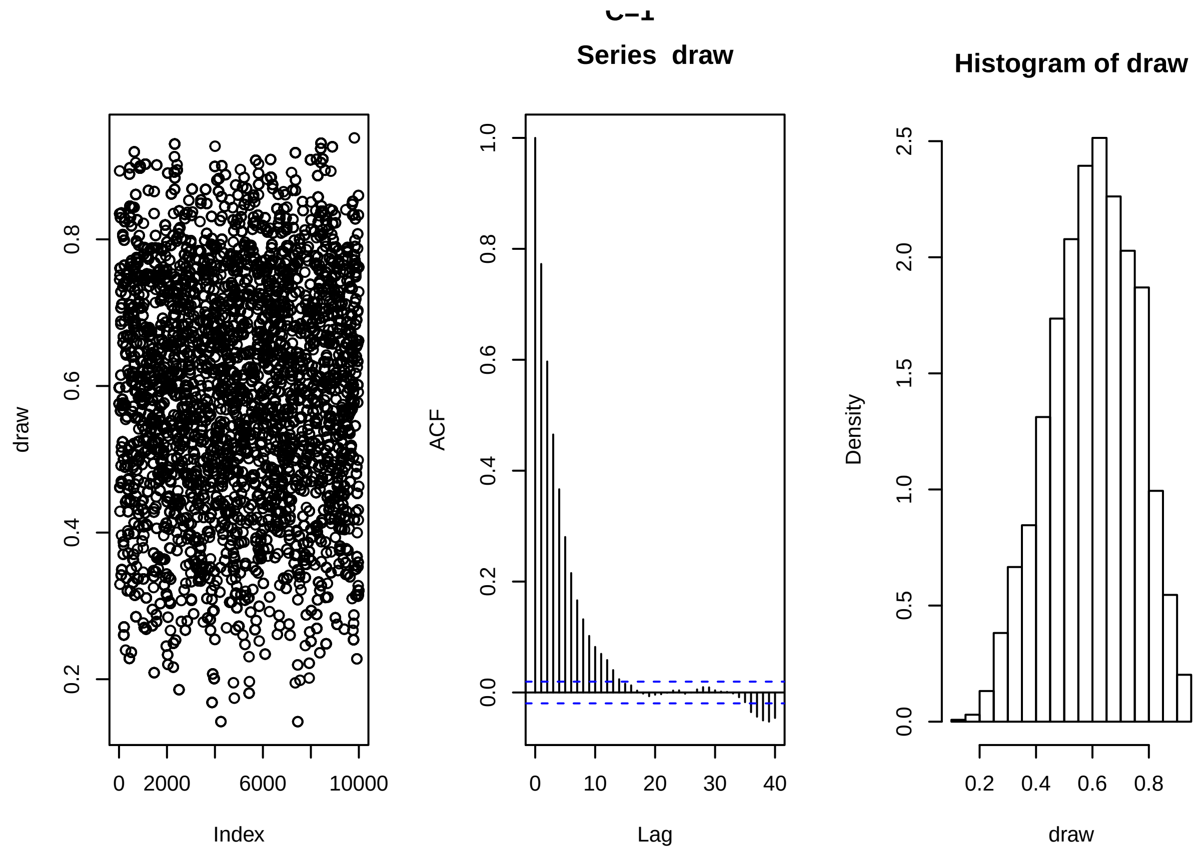
For comparion, generate a histogram with random variables~beta(6,4)

```
a1=rbeta(10000,6,4)
hist(a1,freq=FALSE,main="beta(6,4) histogram")
```



We first start a simulation with c=1, and 10000 draws. Perform KS test to determine whether our generated random variables are numerically approximately follow Beta(6,4)

```
draw=MHalg(10000,1)
par(mfrow=c(1,3))
plot(draw); acf(draw); hist(draw,freq=FALSE)
title("C=1", outer=TRUE)
```




```
ks.test(draw,a1)
```

```
## Warning in ks.test(draw, a1): p-value will be approximate in the presence
## of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  draw and a1
## D = 0.0235, p-value = 0.007992
## alternative hypothesis: two-sided
```

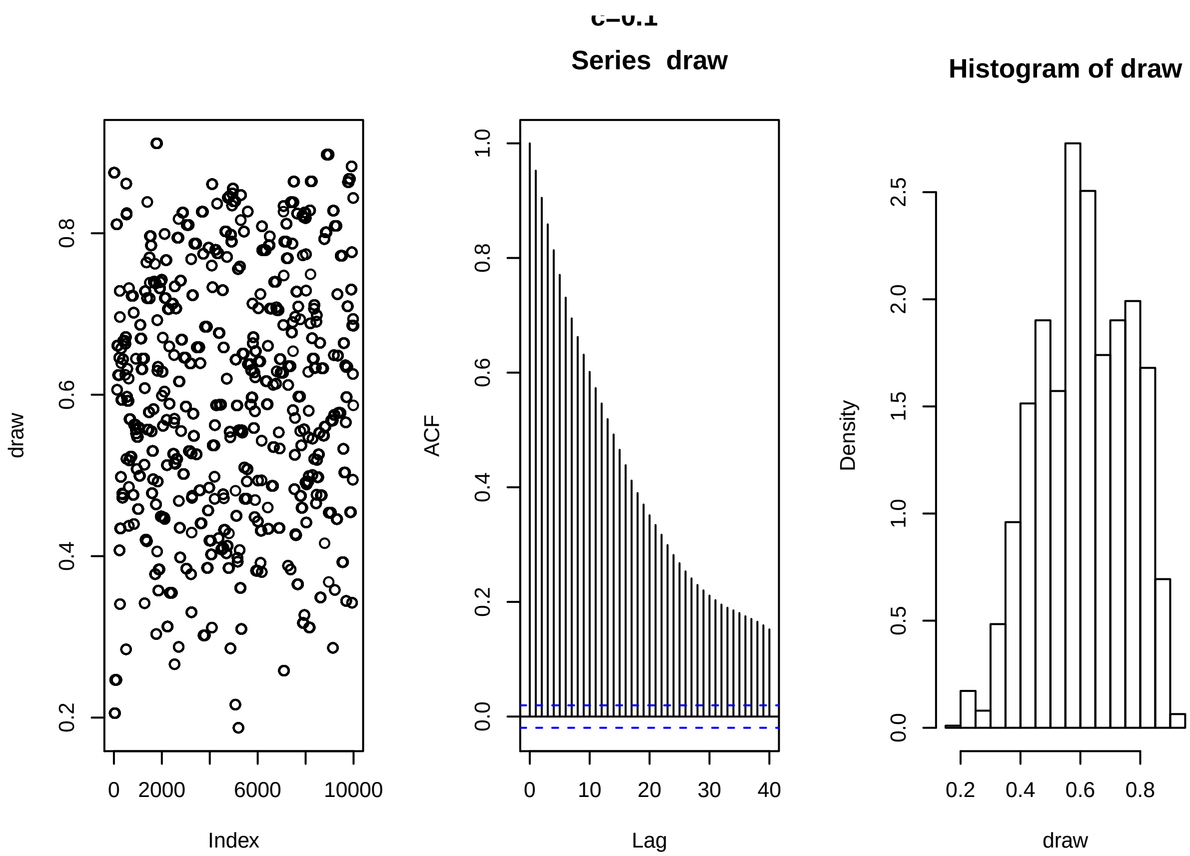
Now we simulate with c=0.1, 2.5, 10.

```
draw=MHalg(10000,0.1)
par(mfrow=c(1,3))
plot(draw); acf(draw); hist(draw,freq=FALSE)
ks.test(draw,a1)
```

```
## Warning in ks.test(draw, a1): p-value will be approximate in the presence
## of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  draw and a1
## D = 0.0824, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```
title("c=0.1", outer=TRUE)
```

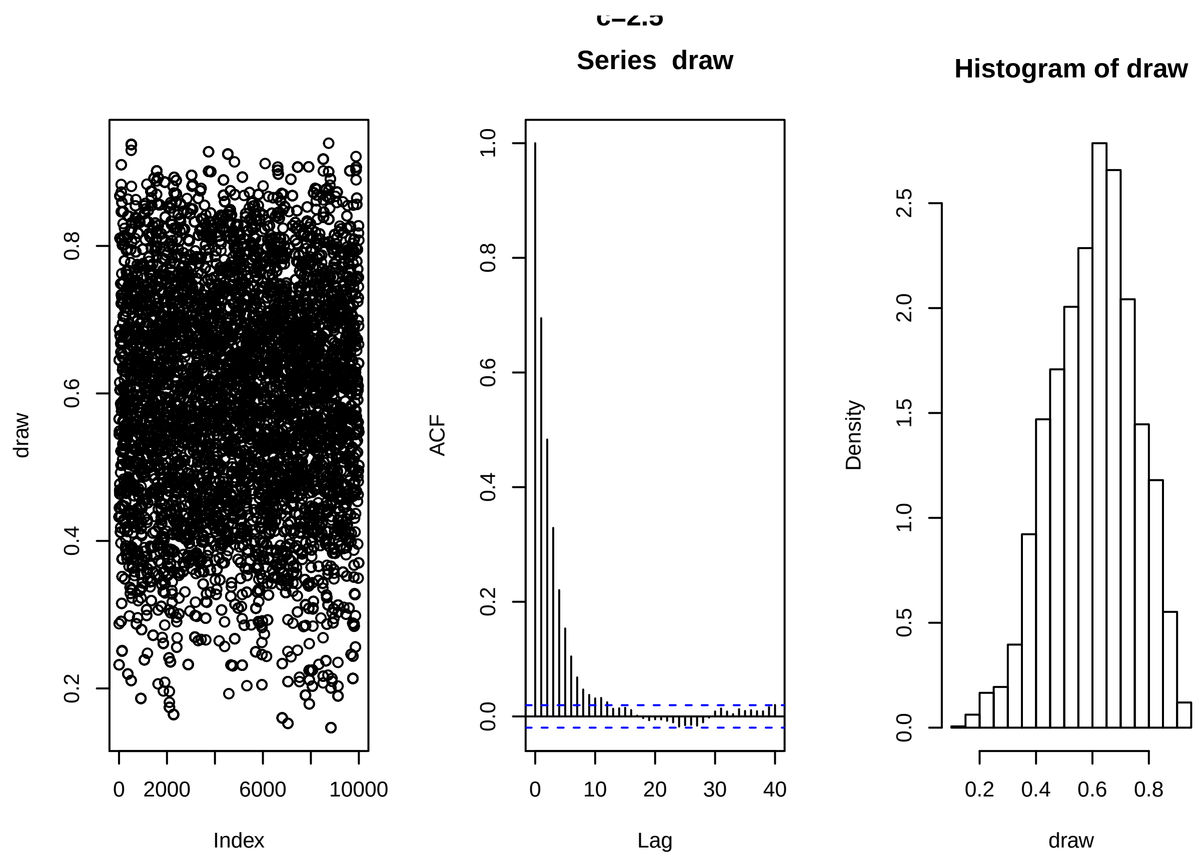


```r
draw=MHalg(10000,2.5)
par(mfrow=c(1,3))
plot(draw); acf(draw); hist(draw,freq=FALSE)
ks.test(draw,a1)
```

```
## Warning in ks.test(draw, a1): p-value will be approximate in the presence
## of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  draw and a1
## D = 0.0289, p-value = 0.0004718
## alternative hypothesis: two-sided
```

```r
title("c=2.5", outer=TRUE)
```

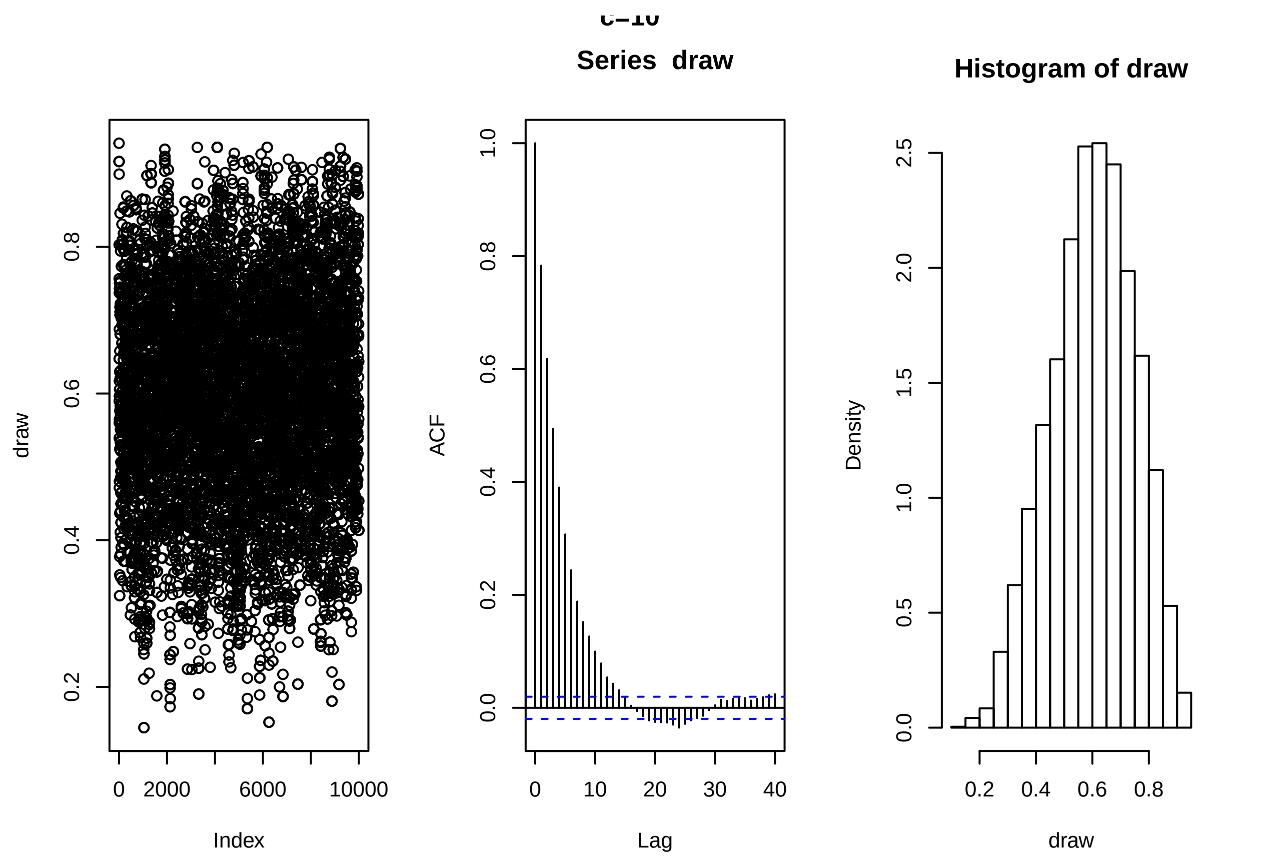


```r
draw=MHalg(10000,10)
par(mfrow=c(1,3))
plot(draw); acf(draw); hist(draw,freq=FALSE)
ks.test(draw,a1)
```

```
## Warning in ks.test(draw, a1): p-value will be approximate in the presence
## of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  draw and a1
## D = 0.0152, p-value = 0.1982
## alternative hypothesis: two-sided
```

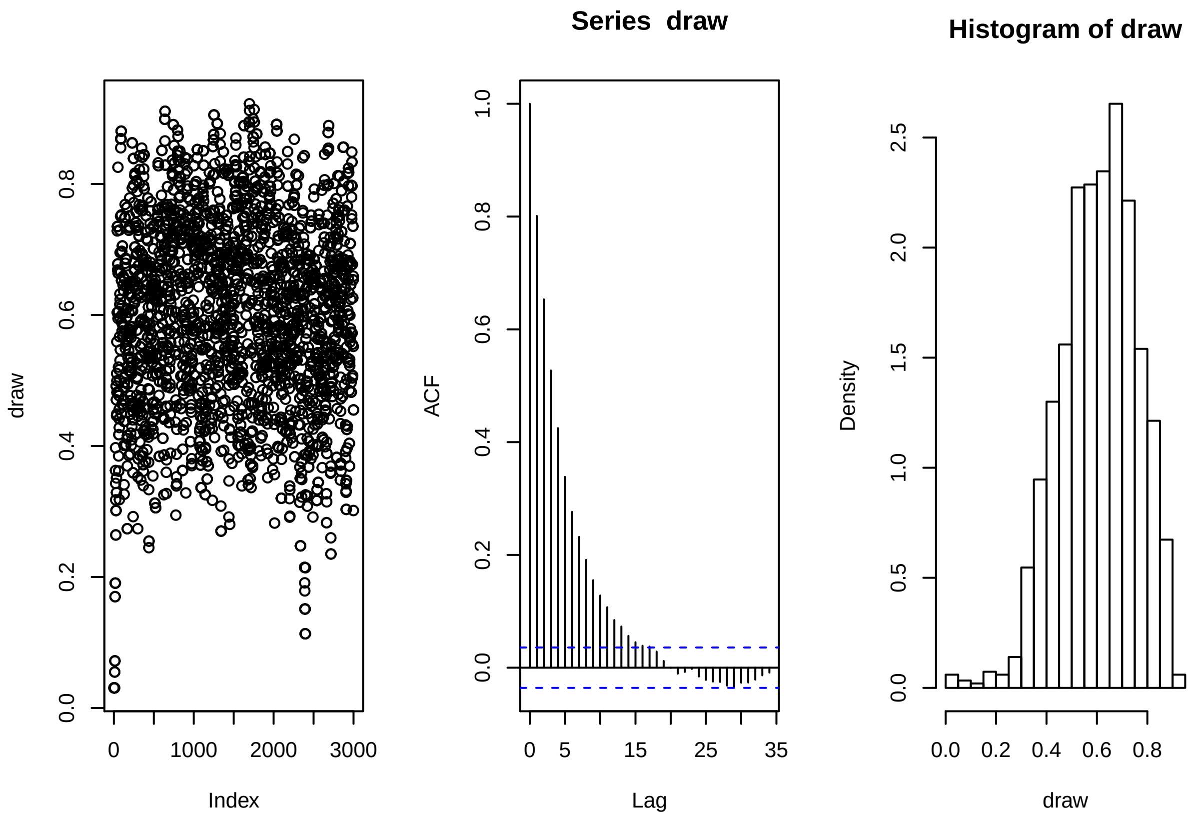
```r
title("c=10", outer=TRUE)
```



Comparing histograms and Kolmogorov–Smirnov statistic, c=10 is the most effective.

Now we lower the iteration to 3000. The result is already very close to beta(6,4)

```r
draw=MHalg(3000,10)
par(mfrow=c(1,3))
plot(draw); acf(draw); hist(draw,freq=FALSE)
```




```
a1=rbeta(3000,6,4)
ks.test(draw,a1)
```

```
## Warning in ks.test(draw, a1): p-value will be approximate in the presence
## of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  draw and a1
## D = 0.0493, p-value = 0.001349
## alternative hypothesis: two-sided
```