```
data = read.table("M1D.txt",h=T) #data file of training
population

M1 = as.matrix(data[,-(1:4)]) #file of markers of training
population

# Separating the subpopulations of the training population
# phenotypes vector of training population
phen = cbind(1:1000, data[,4])
colnames(phen) = c("id","phen")
y = phen[,2]

# phenotypes vector of the training population in descending
order
order = phen[order(phen[,2], decreasing=TRUE),]

# Subpopulation 1 with the highest phenotypes
g1 = order[1:(0.5*nrow(data)),] #phenotypes
M1_new = M1[g1[,1],] #genotypes

# Subpopulation 2 with the smallest phenotypes
g2 = order[(0.5*nrow(data)+1):nrow(data),] #phenotypes
M2_new = M1[g2[,1],] #genotypes

# Allele frequencies of the total population, subpopulation
1 and subpopulation 2
p10 = matrix(0,ncol(M1_new),1)
p20 = matrix(0,ncol(M1_new),1)
p0 = matrix(0,ncol(M1_new),1)
for (i in 1:ncol(M1_new))
{
p10[i,]=(length(which(M1_new[,i]==1))+2*length(which(M1_new
[,i]==2)))/(2*nrow(M1_new))
p20[i,]=(length(which(M2_new[,i]==1))+2*length(which(M2_new
[,i]==2)))/(2*nrow(M2_new))
p0[i,]=(length(which(M1[,i]==1))+2*length(which(M1[,i]==2))
)/(2*nrow(M1))
}

# Matrix of dominance
Wp0 = matrix(0,nrow(M1),ncol(M1))
Mp0 = matrix(0,nrow(M1),ncol(M1))
for(j in 1:ncol(M1))
{
Mp0[,j] = M1[,j]-2*p0[j,] #additive matrix
for(i in 1:nrow(M1))
{
if(M1[i,j]==2)(Wp0[i,j]=-2*(1-p0[j,])^2)
if(M1[i,j]==1)(Wp0[i,j]=2*p0[j,]*(1-p0[j,]))
if(M1[i,j]==0)(Wp0[i,j]=-2*(p0[j,])^2)
}}
```

```
# Exclusion of markers by MAF
MAF = 0.05
maf = NULL
count = NULL
for(i in 1:length(p0))
{
maf[i] = min(p0[i,],1-p0[i,])
if(maf[i]>MAF){count[i]=i}
}

# Deletion of markers with deltap0 = 0
deltap0 = (p10 - p20) #difference between frequencies
count1 = NULL
for(i in 1:length(p0))
{
if(deltap0[i]!=0){count1[i]=i}
}

# Frequencies of markers that were not excluded by the control
int_snp=intersect(as.vector(na.omit(count)),as.vector(na.omit(count1)))
Mp = as.matrix(Mp0[,int_snp])
Wp = as.matrix(Wp0[,int_snp])
p1 = as.matrix(p10[int_snp,])
p2 = as.matrix(p20[int_snp,])
p = as.matrix(p0[int_snp,])


########### Additive Effects

deltap = (p1 - p2) #difference between frequencies
deltap_med = mean(abs(deltap)) #average difference

# Average bm
ad = data[,2]
h2a = var(ad)/var(y)
bm = (0.5*h2a)*(mean(g1[,2])-mean(g2[,2]))/(length(p)*2*deltap_med)

# Calculation of bi - additive effects of markers
bi = (deltap/deltap_med)*bm

########### Independent validation

# Data file of the validation population
data10 = read.table(paste("M",10,"D.txt",sep=""),h=T)
M10 = as.matrix(data10[,-c(1:4)]) #matrix of validation population markers
M100 = M10[,int_snp]
```

```
# Allele frequencies and additive incidence matrix of the
validation population
p10 = matrix(0,ncol(M100),1)
M10p = matrix(0,nrow(M100),ncol(M100))
for(i in 1:ncol(M100))
{
p10[i,]=(length(which(M100[,i]==1))+2*length(which(M100[,i]
==2)))/(2*nrow(M100))
M10p[,i] = M100[,i]-2*p10[i,] # Additive matrix
}

# Additive genomic value of individuals from the validation
population
gbv_p = M10p%*%bi

# Predictive ability, accuracy and prediction bias
ad10 = data10[,2]
y10 = data10[,4]
(res_p=cbind(cor(gbv_p,y10),cor(gbv_p,ad10),
cov(gbv_p,y10)/var(gbv_p)))

########### RR-BLUP / G-BLUP additive
library(rrBLUP)
rrblup = mixed.solve(y,Z=Mp)
gbv_g = M10p%*%rrblup$u
(res_g=cbind(cor(gbv_g,y10),cor(gbv_g,ad10),cov(gbv_g,y10)/
var(gbv_g)))

########### Additive index
cor_pp = cor(gbv_p,ad10)
cor_gg = cor(gbv_g,ad10)
deltap2 = var(gbv_p)/var(gbv_g)

b1=(1-(cor_pp^2)*(deltap2))/
(1-(cor_pp^2)*(cor_gg^2)*(deltap2))
b2 = (1-(cor_gg^2))/(1-(cor_pp^2)*(cor_gg^2)*(deltap2))
I=diag(b1[1],nrow(M10p))%*%as.matrix(gbv_g)+diag(b2[1],nrow
(M10p))%*%as.matrix(gbv_p)
num = (1-(cor_gg^2))*(1-(cor_pp^2)*deltap2)
den = 1-(cor_gg^2)*(cor_pp^2)*deltap2
rIa = (1-num/den)^(1/2)
ef = rIa/cor_gg

########### Dominance Effects
delta2pq = 2*p1*(1-p1) - 2*p2*(1-p2) #difference between 2pq
delta2pq_med = mean(abs(delta2pq)) #difference between 2pq
average

# bm médio
dom = data[,3]
h2d = var(dom)/var(y)
```

```
bm_2pq=(0.5*h2d)*(mean(g1[,2])-
mean(g2[,2]))/(length(p)*delta2pq_med)

# Calculation of bi_2pq – dominance markers effects
bi_2pq = (delta2pq/delta2pq_med)*bm_2pq

########### Independent validation

# Dominance incidence matrix of the validation population
W10p = matrix(0,nrow(M100),ncol(M100))
for(j in 1:ncol(M100))
{
for(i in 1:nrow(M100))
{
if(M100[i,j]==2)(W10p[i,j]=-2*(1-p10[j,])^2)
if(M100[i,j]==1)(W10p[i,j]=2*p10[j,]*(1-p10[j,]))
if(M100[i,j]==0)(W10p[i,j]=-2*(p10[j,])^2)
}}

# Dominance Genomic value of individuals in the validation
population
d_p = W10p%*%bi_2pq

# Predictive ability, accuracy and prediction bias
dom10 = data10[,3]
(res_d_p=cbind(cor(d_p,y10),cor(d_p,dom10),cov(d_p,y10)/var
(d_p)))


########### RR-BLUP / G-BLUP to dominance effects
rrblup_d = mixed.solve(phen[,2],Z=Wp)
d_g = W10p%*%rrblup_d$u
(res_d_g=cbind(cor(d_g,y10),cor(d_g,dom10),cov(d_g,dom10)/v
ar(d_g)))


########### Index to dominance effects

cord_pp = cor(d_p,dom10)
cord_gg = cor(d_g,dom10)

deltap2d = var(d_p)/var(d_g)
b1d=(1-(cord_pp^2)*(deltap2d))/(1-
cord_pp^2)*(cord_gg^2)*(deltap2d))
b2d =(1-(cord_gg^2))/(1-(cord_pp^2)*(cord_gg^2)*(deltap2d))
Id=diag(b1d[1],nrow(Wp))%*%as.matrix(d_g)+diag(b2d[1],nrow(
Wp))%*%as.matrix(d_p)
numd = (1-(cord_gg^2))*(1-(cord_pp^2)*deltap2d)
dend = 1-(cord_gg^2)*(cord_pp^2)*deltap2d
rId = (1-numd/dend)^(1/2)
efd = rId/cord_gg
```