In [1]:
```
# Founders: real haplotype data (ch1to10.200SNP)
# "Sample animals for sire and dam candidates" & "Dams and Sires": dam size = N of dams
# One Dam -> 1 male & 1 female (1 dam -> 2 offsprings)
# selection: increase
# 5 generation selection: increase
# heritability = 0.3
# Phenotypes : all animals in G0 to G4
# Genotypes  : all progeny in G5 and all sires in each generation
# Change muAlpha = 0.2
# 10 chromosomes; 20 loci per chromosome = > 200 Loci (50 QTL & 150 Markers)
```

In [2]:
```
include("/home/nicole/Jupyter/XSimSel.jl")
```

Out[2]: XSim

In [3]:
```
using DataFrames
```

In [4]:
```
using Distributions
```

In [5]:
```
using(Gadfly)
```

# Initialize XSim

In [6]:
```
numChr    = 10
chrLength = 0.01
numLoci   = 20
nQTL      = 5
mutRate   = 0.0
locusInt  = chrLength/numLoci
mapPos    = collect(locusInt/2:locusInt:chrLength)
geneFreq  = fill(0.5,numLoci)
QTL = sample(1:numLoci,nQTL,replace=false)
qtlMarker = fill(false,numLoci)
qtlMarker[QTL] = true
mu = 100                                  #  alpha ~ N(100,1)
Va = nQTL*numChr*0.5*mu*mu                # Va= nQTL*2pq*mean(alpha)^2
qtlEffects= rand(Normal(mu/sqrt(Va), sqrt(1/Va)),numLoci*numChr)  # Let alpha mu = 0.2
XSim.init(numChr,numLoci,chrLength,geneFreq,mapPos,qtlMarker,qtlEffects,mutRate)
```

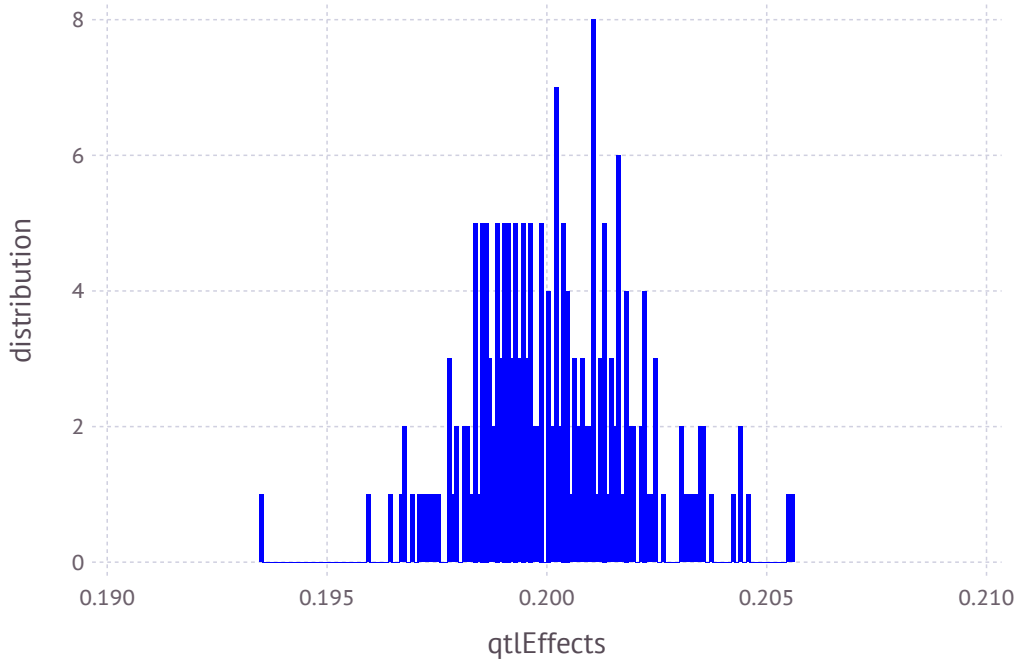
In [7]:
```
qtlEffects
```

Out[7]:
```
200-element Array{Float64,1}:
 0.198554
 0.201061
 0.199481
 0.201486
 0.203387
 0.197112
 0.199904
 0.199639
 0.198117
 0.198545
 0.197845
 0.20199
 0.197745
 ⋮
 0.199539
 0.199042
 0.199885
 0.198335
 0.201246
 0.196437
 0.20115
 0.198394
 0.202678
 0.199144
 0.201048
 0.201904
```

In [8]:
```
writedlm("qtlEffects",qtlEffects)
```

In [9]:
```
plot(x=qtlEffects, Geom.histogram, Guide.XLabel("qtlEffects"), Guide.YLabel("distribution"), Theme(defaul
```

Out[9]:




In [10]:
```
mean(qtlEffects)
```

Out[10]:
```
0.20018102444872266
```

In [11]:
```
var(qtlEffects)
```

Out[11]:
```
3.466864475074167e-6
```

In [12]:
```
# Base Population
gen=0
nGenBase    = 5
popSizeBase = 8000

# Sample 20 sire and 400 dams
popSizeSP = 8000

# Purbred Populations - mating
popSize = 8000
nGener  = 5
nSires  = 200
nDams   = 4000
npop = 1
;
```

In [13]:
```
# Animals with genotypes
posOFF0S = 1
posOFF0E = popSizeSP
posOFF1S = posOFF0E + 1
posOFF1E = posOFF0E + popSize
posOFF2S = posOFF1E + 1
posOFF2E = posOFF1E + popSize
posOFF3S = posOFF2E + 1
posOFF3E = posOFF2E + popSize
posOFF4S = posOFF3E + 1
posOFF4E = posOFF3E + popSize
posOFF5S = posOFF4E + 1
posOFF5E = posOFF4E + popSize
println("posOFF1S: ",posOFF1S,"; posOFF1E: ",posOFF1E)
println("posOFF2S: ",posOFF2S,"; posOFF2E: ",posOFF2E)
println("posOFF3S: ",posOFF3S,"; posOFF3E: ",posOFF3E)
println("posOFF4S: ",posOFF4S,"; posOFF4E: ",posOFF4E)
println("posOFF5S: ",posOFF5S,"; posOFF5E: ",posOFF5E)

# FileName :
PedAll = "PedAll.txt"           # pedigree  file with all animals
PheAll = "PheAll.txt"           # phenotype file with all animsla
GenAll = "GenAll.txt"           # genotype  file with all animals

Gen = "Gen.txt"                 # genotype file with all progeny in G5 and all sires in each generation
Phe = "Phe.txt"                 # phenotype file with all animals in G1 to G4
QTL = "QTL.txt"                 # QTL with with all progeny in G5 and all sires in each generation
Mar = "Mar.txt"                 # Marker with with all progeny in G5 and all sires in each generation
GenNF = "GenNF.txt"             # remove fixed genes from genotype file
QTLNF = "QTLNF.txt"             # remove fixed genes from QTL file
MarNF = "MarNF.txt"             # remove fixed genes from Marker file
;
```

```
posOFF1S: 8001; posOFF1E: 16000
posOFF2S: 16001; posOFF2E: 24000
posOFF3S: 24001; posOFF3E: 32000
posOFF4S: 32001; posOFF4E: 40000
posOFF5S: 40001; posOFF5E: 48000
```

In [14]:
```
fileName = "SS"
pedText = fileName * ".ped"
genText = fileName * ".gen"
pheText = fileName * ".phe"
;
```

In [15]:
```
;rm $pedText $genText $pheText
```

```
rm: cannot remove 'SS.ped': No such file or directory
rm: cannot remove 'SS.gen': No such file or directory
rm: cannot remove 'SS.phe': No such file or directory
```

# Sampling Founders

In [16]:
```
sires = XSim.sampleFounders("/home/nicole/Jupyter/HERdata/data/Clean/newdata/ch1to10.200SNP.group1")
dams  = XSim.sampleFounders("/home/nicole/Jupyter/HERdata/data/Clean/newdata/ch1to10.200SNP.group2");
```

```
Sampling 360 animals into base population.
Sampling 361 animals into base population.
```

# Base population

In [17]:
```
baseSires, baseDams, baseGen = XSim.sampleRan(popSizeBase, nGenBase, sires, dams, gen=gen);
```

```
Generation     1: sampling  4000 males and  4000 females
Generation     2: sampling  4000 males and  4000 females
Generation     3: sampling  4000 males and  4000 females
Generation     4: sampling  4000 males and  4000 females
Generation     5: sampling  4000 males and  4000 females
```

# Sample animals for sire and dam candidates

In [18]:
```
popSP = XSim.sampleRan(popSizeSP, 1, baseSires, baseDams, gen=baseGen,fileName=fileName);
```

```
Generation     6: sampling  4000 males and  4000 females
```

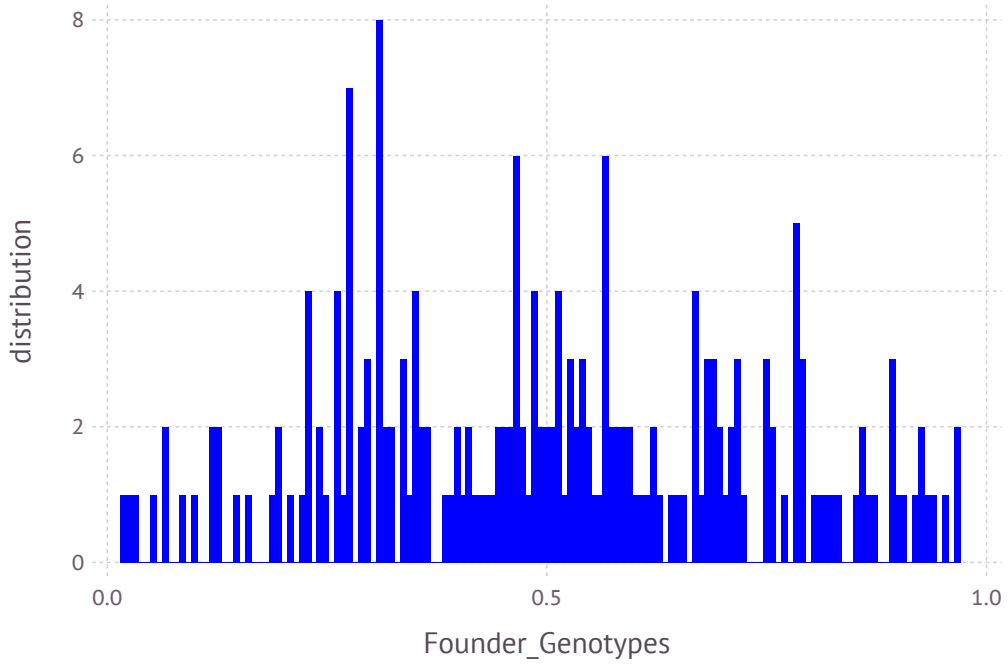
In [19]:
```
gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

In [20]:
```
FCM = mean(gSP/2,1)
```

Out[20]:
```
1x200 Array{Float64,2}:
 0.063  0.831625  0.288375  0.94125  …  0.35075  0.3945  0.89275  0.568875
```

In [21]:
```
plot(x=FCM, Geom.histogram, Guide.XLabel("Founder_Genotypes"), Guide.YLabel("distribution"), Theme(defaul
```

Out[21]:




In [22]:
```
V=var(gSP,1)
Mark=gSP[:,V.>0]
corMat=cor(Mark)
nRows =size(corMat,1)
LDMat =zeros(nRows-1,20);
```

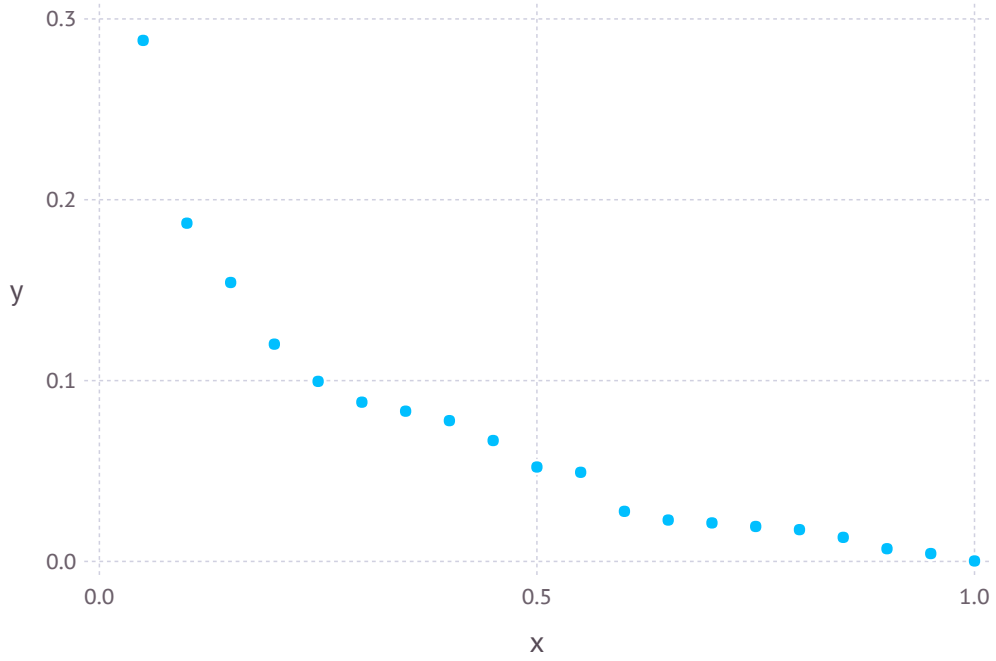
In [23]:
```
for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

In [24]:
```
y=mean(LDMat,1)
sort(y,2)
```

Out[24]:
```
1x20 Array{Float64,2}:
 0.000252855  0.00438695  0.00703225  …  0.154263  0.187048  0.288099
```

In [25]:
```
plot(x=(1:20)/20*1,y=y)
```

Out[25]:




In [26]:
```
FCMstream = open("SNPCMF.txt", "w")
```

Out[26]:
```
IOStream(<file SNPCMF.txt>)
```

In [27]:
```
for i in 1:size(FCM,2)
    @printf(FCMstream, "%6.4f ", FCM[1,i])
end
```

In [28]:
```
close(FCMstream)
```

# Selection - increase

In [29]:
```
aSPSire = XSim.getOurGenVals(popSP[1])
aSPDam = XSim.getOurGenVals(popSP[2])
aSP = [aSPSire;aSPDam];
```

In [30]:
```
mean(aSP)
```

Out[30]:
```
10.878614839689178
```

In [31]:
```
varGen=var(aSP)
```

Out[31]:
```
0.6345603381799936
```

In [32]:
```
XSim.common.varRes = (7*varGen)/3    #heritability = 0.3
```

Out[32]:
```
1.4806407890866515
```

In [33]:
```
varRes = XSim.common.varRes
```

Out[33]:
```
1.4806407890866515
```

In [34]:
```
popRMP = XSim.sampleSelNoLap(nSires, nDams, nGener,popSP[1],popSP[2],gen=popSP[3],fileName=fileName,direc
```

```
Generation     7: sampling  4000 males and  4000 females
Generation     8: sampling  4000 males and  4000 females
Generation     9: sampling  4000 males and  4000 females
Generation    10: sampling  4000 males and  4000 females
Generation    11: sampling  4000 males and  4000 females
```

In [35]:
```
ymRMP = XSim.getOurGenVals(popRMP[1])    # for males: pop[1]
mean(ymRMP)
```

Out[35]: 13.105562155221044

In [36]:
```
yfRMP = XSim.getOurGenVals(popRMP[2])    # for females: pop[2]
mean(yfRMP)
```

Out[36]: 13.120614552293832

In [37]:
```
amRMP = XSim.getOurGenVals(popRMP[1])
var(amRMP)
```

Out[37]: 0.6032751600606551

In [38]:
```
afRMP = XSim.getOurGenVals(popRMP[2])
var(afRMP)
```

Out[38]: 0.6176354750475195

# Pedigree: All animals

In [39]:
```
PED = convert(Array,readtable(pedText,separator=' ',header=false))
```

Out[39]:
```
48000x3 Array{Int64,2}:
 40722  33349  37716
 40723  33433  38407
 40724  36662  37365
 40725  36451  39262
 40726  33753  40405
 40727  33162  40721
 40728  32883  38114
 40729  35371  38065
 40730  35577  40626
 40731  36032  37205
 40732  33677  37399
 40733  35303  37639
 40734  35997  36838
     ⋮
 88710  74944  77717
 88711  76603  78935
 88712  75801  79425
 88713  76665  80711
 88714  72969  79244
 88715  75308  77485
 88716  72934  79070
 88717  75970  78896
 88718  76580  77363
 88719  75434  80596
 88720  76715  79442
 88721  74069  80591
```

In [40]:
```
PEDstream = open(PedAll, "w")
```

Out[40]:
```
IOStream(<file PedAll.txt>)
```

In [41]:
```
for i in 1:size(PED,1)
    @printf(PEDstream, "%19d%19d%19d \n", PED[i,1], PED[i,2], PED[i,3])
end
```

In [42]:
```
close(PEDstream)
```

# Create matrix of `genotype" covariates for all animals

In [43]:
```
nObs = countlines(pedText)
```

Out[43]: 48000

In [44]:
```
nMarker = numChr*numLoci
```

Out[44]: 200

In [45]:
```
GT = convert(Array,readtable(genText,separator=' ',header=false))
```

Out[45]: 48000x201 Array{Int64,2}:

| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 40722 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | … | 2 | 1 | 0 | 2 | 0 | 1 | 2 | 0 | 0 | 2 | 1 | 2 |
| 40723 | 0 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 2 | 0 | | 1 | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 1 | 2 | 2 |
| 40724 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 2 | 2 | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 2 |
| 40725 | 0 | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | | 2 | 2 | 2 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 2 | 1 |
| 40726 | 0 | 1 | 1 | 1 | 1 | 0 | 2 | 2 | 1 | 1 | | 2 | 2 | 2 | 1 | 2 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 40727 | 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | … | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 0 |
| 40728 | 0 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 1 | 2 | 2 |
| 40729 | 0 | 0 | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | | 2 | 1 | 1 | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 40730 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 40731 | 1 | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 |
| 40732 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | … | 1 | 0 | 2 | 2 | 2 | 1 | 1 | 0 | 0 | 2 | 1 | 2 |
| 40733 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 2 | 2 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 1 | 1 |
| 40734 | 1 | 1 | 1 | 2 | 2 | 1 | 2 | 2 | 0 | 2 | | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 1 | 2 | 0 |
| ⋮ | | | | ⋮ | | | | ⋮ | ⋱ | | | ⋮ | | | | ⋮ | | | | | ⋮ | | |
| 88710 | 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 2 | 2 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 88711 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 2 | 1 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 2 | 0 |
| 88712 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 1 | … | 2 | 0 | 0 | 2 | 1 | 2 | 1 | 0 | 0 | 2 | 1 | 1 |
| 88713 | 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | | 2 | 0 | 1 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 88714 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 2 | 1 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 0 | 2 | 2 |
| 88715 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 2 | 1 | 2 | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 1 | 1 |
| 88716 | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 0 | 2 | | 2 | 0 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 1 | 2 | 1 |
| 88717 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | … | 1 | 0 | 1 | 2 | 1 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 88718 | 0 | 2 | 1 | 2 | 1 | 0 | 0 | 0 | 2 | 0 | | 2 | 1 | 0 | 2 | 1 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 88719 | 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | | 1 | 0 | 1 | 2 | 1 | 1 | 0 | 1 | 1 | 0 | 2 | 1 |
| 88720 | 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | | 2 | 2 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 88721 | 0 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 2 | 0 | | 1 | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 1 | 2 | 2 |

In [46]:
```
allID = GT[:,1];
```

In [47]:
```
GTM = GT[:,2:end];
```

# Create marker file for all animals

In [48]:
```
M = GTM     # maker file for Julia
```

Out[48]:
```
48000x200 Array{Int64,2}:
 0  2  0  2  2  0  0  0  2  0  0  1  1  …  2  1  0  2  0  1  2  0  0  2  1  2
 0  2  0  2  2  0  1  1  2  0  0  2  2     1  0  1  2  0  1  1  0  0  1  2  2
 0  1  1  2  2  1  1  1  1  1  1  0  1     0  0  2  2  0  0  1  0  0  2  1  2
 0  2  1  2  1  1  1  1  1  1  1  1  1     2  2  2  1  2  2  0  1  1  1  2  1
 0  1  1  1  1  0  2  2  1  1  1  1  2     2  2  2  1  2  2  0  1  1  0  2  1
 0  2  0  2  2  1  1  1  1  1  1  1  2  …  2  2  2  0  2  2  0  2  2  0  2  0
 0  1  2  2  1  1  1  1  1  1  1  1  1     1  1  1  2  1  1  1  0  0  1  2  2
 0  0  2  2  2  0  2  2  0  2  2  0  0     2  1  1  1  1  2  0  1  1  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  0  1     1  0  2  2  2  1  1  1  1  0  1  1
 1  2  1  2  1  1  1  1  1  1  1  1  1     1  1  2  1  1  1  1  1  1  1  2  1
 0  1  1  1  1  0  1  1  1  1  1  0  1  …  1  0  2  2  2  1  1  0  0  2  1  2
 0  2  0  2  2  0  0  0  2  0  0  1  1     2  2  1  1  2  2  0  1  1  1  1  1
 1  1  1  2  2  1  2  2  0  2  2  0  1     2  2  2  0  2  2  0  1  1  1  2  0
 ⋮              ⋮              ⋮        ⋱        ⋮              ⋮
 0  2  0  2  2  1  1  1  1  1  1  0  1     0  0  2  2  1  1  0  1  1  1  1  1
 0  2  0  2  2  0  0  0  2  0  0  0  0     2  1  1  1  2  2  0  1  1  1  2  0
 0  1  1  1  2  0  1  1  1  1  1  1  1  …  2  0  0  2  1  2  1  0  0  2  1  1
 0  2  0  2  2  1  1  1  1  1  1  1  2     2  0  1  2  0  1  0  0  0  0  2  2
 0  2  0  2  2  0  0  0  2  0  0  1  1     2  1  0  2  1  2  0  0  0  0  2  2
 0  2  0  2  2  0  0  0  2  0  0  0  0     2  1  2  1  1  2  0  1  1  0  1  1
 0  1  1  2  2  2  2  2  0  2  2  0  2     2  0  0  2  1  2  0  0  0  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  …  1  0  1  2  1  2  0  1  1  0  2  1
 0  2  1  2  1  0  0  0  2  0  0  1  0     2  1  0  2  1  2  0  1  1  0  2  1
 0  2  0  2  2  1  1  1  1  1  0  1  1     1  0  1  2  1  1  0  1  1  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  0  0     2  2  1  1  2  2  0  1  1  0  2  1
 0  2  0  2  2  0  1  1  2  0  0  1  1     1  0  1  2  0  1  1  0  0  1  2  2
```

In [49]:
```
Mstream = open(GenAll, "w")
```

Out[49]:
```
IOStream(<file GenAll.txt>)
```

In [50]:
```
for i in 1:size(M,1)
    @printf(Mstream, "%19d", allID[i])
    for j in 1:size(M,2)
        @printf(Mstream, "%3d", M[i,j])
    end
    @printf(Mstream, "\n")
end
```

In [51]:
```
close(Mstream)
```

# Genotypes - all sires and all offsprings in G5

In [52]:
```
AllSire = PED[posOFF1S:posOFF5E,2]
```

Out[52]:
```
40000-element Array{Int64,1}:
 42719
 40748
 43434
 41389
 41401
 40780
 41554
 43289
 41599
 41279
 43637
 44110
 41882
     ⋮
 74944
 76603
 75801
 76665
 72969
 75308
 72934
 75970
 76580
 75434
 76715
 74069
```

In [53]:
```
SireID = unique(AllSire)
```

Out[53]:
```
1000-element Array{Int64,1}:
 42719
 40748
 43434
 41389
 41401
 40780
 41554
 43289
 41599
 41279
 43637
 44110
 41882
     ⋮
 75638
 76614
 75903
 75545
 73925
 76271
 73885
 76065
 75905
 75113
 76336
 76432
```

In [54]:
```
OFF5 = PED[posOFF5S:posOFF5E,1]
```

Out[54]:
```
8000-element Array{Int64,1}:
 80722
 80723
 80724
 80725
 80726
 80727
 80728
 80729
 80730
 80731
 80732
 80733
 80734
     ⋮
 88710
 88711
 88712
 88713
 88714
 88715
 88716
 88717
 88718
 88719
 88720
 88721
```

In [55]:
```
SireOFF5ID = [SireID;OFF5]
```

Out[55]:
```
9000-element Array{Int64,1}:
 42719
 40748
 43434
 41389
 41401
 40780
 41554
 43289
 41599
 41279
 43637
 44110
 41882
     ⋮
 88710
 88711
 88712
 88713
 88714
 88715
 88716
 88717
 88718
 88719
 88720
 88721
```

In [56]:
```
SOFF5ID= DataFrame()
SOFF5ID[:ID] = SireOFF5ID;
```

In [57]:
```
typeof(SOFF5ID)
```

Out[57]:
```
DataFrames.DataFrame
```

In [58]:
```
MT = readtable(GenAll,separator=' ',header=false);
```

In [59]:
```
rename!(MT,:x1,:ID);
```

In [60]:
```
GSOFF5 = join(SOFF5ID, MT, on = :ID, kind = :inner);
```

In [61]:
```
size(GSOFF5)
```

Out[61]: (9000,201)

In [62]:
```
GSOFF5Row = size(GSOFF5,1)
```

Out[62]: 9000

In [63]:
```
GSOFF5Col = size(GSOFF5,2)
```

Out[63]: 201

In [64]:
```
GSOFF5stream = open(Gen, "w")
```

Out[64]: IOStream(<file Gen.txt>)

In [65]:
```
for i in 1:size(GSOFF5,1)
    for j in 1
        @printf(GSOFF5stream, "%19d", GSOFF5[i,j])
    end
    for k in 2:size(GSOFF5,2)
        @printf(GSOFF5stream, "%3d", GSOFF5[i,k])
    end
    @printf(GSOFF5stream, "\n")
end
```

In [66]:
```
close(GSOFF5stream)
```

# Phenotypes - All animals

In [67]:
```
PBV = convert(Array,readtable(pheText,separator=' ',header=false))
```

Out[67]:
```
48000x3 Array{Real,2}:
 40722  10.689  10.269
 40723  13.177  11.672
 40724  10.743  10.455
 40725   8.207  10.639
 40726   9.807  10.245
 40727  10.03   10.842
 40728  12.573  11.452
 40729  10.887  10.039
 40730  12.282  12.256
 40731  10.715  11.642
 40732  12.959  11.849
 40733  10.984  10.641
 40734   9.206   9.439
     ⋮
 88710  13.521  14.266
 88711  14.252  14.256
 88712  12.232  13.465
 88713  14.718  14.281
 88714  13.724  15.482
 88715  12.562  13.846
 88716  13.534  14.469
 88717  13.426  14.66
 88718  16.227  13.661
 88719  15.55   14.867
 88720  11.769  13.66
 88721  12.629  13.269
```

In [68]:
```
PBVstream = open(PheAll, "w")
```

Out[68]:
```
IOStream(<file PheAll.txt>)
```

In [69]:
```
for i in 1:size(PBV,1)
    @printf(PBVstream, "%19d %10.3f %10.3f \n", PBV[i,1], PBV[i,2], PBV[i,3])
end
```

In [70]:
```
close(PBVstream )
```

# Phenotypes - all animnls from G0 to G4

In [71]:
```
OFFG0toG4 = PED[posOFF0S:posOFF4E,1]
```

Out[71]:
```
40000-element Array{Int64,1}:
 40722
 40723
 40724
 40725
 40726
 40727
 40728
 40729
 40730
 40731
 40732
 40733
 40734
     ⋮
 80710
 80711
 80712
 80713
 80714
 80715
 80716
 80717
 80718
 80719
 80720
 80721
```

In [72]:
```
OFFG0toG4ID= DataFrame()
OFFG0toG4ID[:ID] = OFFG0toG4;
```

In [73]:
```
typeof(OFFG0toG4ID)
```

Out[73]:
```
DataFrames.DataFrame
```

In [74]:
```
AllPBV= readtable(pheText,separator=' ',header=false);
```

In [75]:
```
rename!(AllPBV,:x1,:ID)
head(AllPBV);
```

In [76]:
```
OFFG0toG4PBV = join(OFFG0toG4ID, AllPBV, on = :ID, kind = :inner);
```

In [77]:
```
Row = size(OFFG0toG4PBV,1)
```

Out[77]:
```
40000
```

In [78]:
```
Col = size(OFFG0toG4PBV,2)
```

Out[78]:
```
3
```

In [79]:
```
Phestream = open(Phe, "w")
```

Out[79]:
```
IOStream(<file Phe.txt>)
```

In [80]:
```
for i in 1:size(OFFG0toG4PBV,1)
    @printf(Phestream, "%19d %10.3f %10.3f \n", OFFG0toG4PBV[i,1], OFFG0toG4PBV[i,2], OFFG0toG4PBV[i,3])
end
```

In [81]:
```
close(Phestream)
```

# Get files with QTL only or Markers only

## QTL file and Markers file

In [82]:
```
QTLPos = XSim.common.G.qtl_index
```

Out[82]:
```
50-element Array{Int64,1}:
   2
   5
  12
  14
  17
  22
  25
  32
  34
  37
  42
  45
  52
   ⋮
 154
 157
 162
 165
 172
 174
 177
 182
 185
 192
 194
 197
```

In [83]:
```
k = size(M,2)
MarkerPos = deleteat!(collect(1:k),sort(QTLPos))
```

Out[83]:
```
150-element Array{Int64,1}:
   1
   3
   4
   6
   7
   8
   9
  10
  11
  13
  15
  16
  18
   ⋮
 186
 187
 188
 189
 190
 191
 193
 195
 196
 198
 199
 200
```

## Genotype codes: 0, 1, 2

In [84]:
```
IDgen = convert(Array,readtable(Gen,separator=' ',header=false));
```

In [85]:
```
onlyID = IDgen[:,1]
QTLMarker = IDgen[:, 2:end]
```

Out[85]: 9000x200 Array{Int64,2}:

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 2 | 0 | 0 | 2 | 2 | … | 1 | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 1 | 2 | 2 |
| 0 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | | 2 | 0 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 1 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | | 2 | 0 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 1 | 2 | 1 |
| 1 | 2 | 0 | 2 | 2 | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | | 1 | 0 | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 2 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 2 | | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 0 |
| 0 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | … | 2 | 2 | 0 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 2 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | 0 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 2 | | 1 | 2 | 1 | 2 | 0 | 2 | 1 | 0 | 0 | 2 | 1 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | | 0 | 0 | 2 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 2 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | | 0 | 0 | 2 | 2 | 0 | 0 | 2 | 0 | 0 | 2 | 2 | 2 |
| 0 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | … | 2 | 2 | 0 | 2 | 1 | 1 | 1 | 0 | 0 | 2 | 1 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | | 2 | 2 | 1 | 2 | 2 | 2 | 0 | 0 | 0 | 1 | 2 | 2 |
| ⋮ | | | | ⋮ | | | | | ⋮ | | | | ⋱ | | | ⋮ | | | | | ⋮ | | | | |
| 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | | 0 | 0 | 2 | 2 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | | 2 | 1 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 1 | 2 | 0 |
| 0 | 1 | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | … | 2 | 0 | 0 | 2 | 1 | 2 | 1 | 0 | 0 | 2 | 1 | 1 |
| 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | | 2 | 0 | 1 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | | 2 | 1 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 0 | 2 | 2 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | | 2 | 1 | 2 | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 0 | 2 | 2 | 0 | 2 | | 2 | 0 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 1 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | … | 1 | 0 | 1 | 2 | 1 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 0 | 2 | 1 | 2 | 1 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | | 2 | 1 | 0 | 2 | 1 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | | 1 | 0 | 1 | 2 | 1 | 1 | 0 | 1 | 1 | 0 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | | 2 | 2 | 1 | 1 | 2 | 2 | 0 | 1 | 1 | 0 | 2 | 1 |
| 0 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 2 | 0 | 0 | 1 | 1 | | 1 | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 1 | 2 | 2 |

In [86]:
```
onlyQTL = QTLMarker[:,QTLPos]
```

Out[86]:
```
9000x50 Array{Int64,2}:
 2  2  2  1  0  0  1  2  2  0  1  1  2  …  1  2  0  0  2  2  1  0  2  2  1  0
 1  2  1  0  1  1  1  0  0  0  1  2  1     1  2  0  0  1  2  1  2  2  2  2  0
 2  2  0  2  1  0  1  2  2  1  2  0  2     0  2  0  0  1  1  2  1  2  2  2  0
 2  2  0  2  1  1  2  1  1  0  2  1  1     1  2  0  0  1  2  1  1  2  2  1  0
 2  2  2  0  0  0  1  2  2  1  2  1  2     1  2  0  0  1  2  1  0  0  0  2  2
 1  2  0  2  1  1  1  1  1  1  0  1  2  …  1  2  0  0  1  2  1  2  1  2  2  0
 2  2  1  2  1  2  2  1  1  1  1  0  2     0  2  1  0  1  1  2  0  0  0  2  2
 2  2  2  0  1  1  1  1  1  1  0  1  1     1  2  1  0  2  2  2  1  1  2  2  0
 2  2  1  1  1  1  2  1  1  0  0  0  2     0  2  0  0  2  2  0  0  2  2  0  0
 2  2  1  1  1  2  2  1  1  0  1  1  2     1  2  0  0  2  2  0  0  2  2  0  0
 1  2  0  2  2  1  2  2  2  1  1  1  2  …  2  2  0  0  2  2  1  0  1  2  1  0
 2  2  0  2  1  1  1  2  2  1  1  0  2     2  2  1  0  1  2  1  0  1  1  1  1
 2  2  0  2  2  1  1  1  1  1  1  2  1     2  2  1  0  2  2  1  1  2  2  2  0
 ⋮              ⋮              ⋮          ⋱        ⋮              ⋮
 2  2  0  2  2  1  1  1  1  1  1  1  2     2  2  1  1  2  2  0  0  2  2  1  1
 2  2  0  2  2  1  1  1  1  0  2  2  1     1  2  2  0  1  2  2  1  1  1  2  1
 1  2  1  1  1  0  0  1  1  1  1  1  2  …  1  2  1  1  1  2  1  2  2  2  2  0
 2  2  1  1  0  2  2  0  1  1  2  1  1     2  2  0  1  1  2  1  2  2  2  1  0
 2  2  1  1  1  2  2  1  2  0  1  2  1     1  2  1  2  2  2  2  2  2  2  2  0
 2  2  0  2  1  2  1  0  1  2  2  2  0     1  2  2  0  2  2  2  1  1  1  2  1
 1  2  0  1  1  1  2  1  1  1  2  2  0     1  2  2  2  2  2  2  2  2  2  2  0
 2  2  1  1  2  1  2  2  2  1  2  1  1  …  1  2  2  0  2  2  2  0  2  2  2  1
 2  1  1  2  1  0  1  1  1  1  2  1  1     1  2  1  1  2  2  1  1  2  2  2  1
 2  2  1  1  1  1  2  2  1  2  1  1  1     2  2  0  2  1  2  2  1  2  2  1  1
 2  2  0  2  2  1  1  1  1  1  1  2  2     1  2  0  0  2  2  0  1  1  1  2  1
 2  2  1  1  1  1  1  1  1  0  2  2  0     1  2  0  1  1  2  1  0  2  2  1  0
```

In [87]:
```
onlyMar = QTLMarker[:,MarkerPos];
```

In [88]:
```
QTLstream = open(QTL, "w")
Marstream = open(Mar, "w");
```

In [89]:
```
for i in 1:size(onlyID,1)
    @printf(QTLstream, "%19d", onlyID[i])
    for j in 1:size(onlyQTL,2)
        @printf(QTLstream, "%3d", onlyQTL[i,j])
    end
    @printf(QTLstream, "\n")
end
```

In [90]:
```
for i in 1:size(onlyID,1)
    @printf(Marstream, "%19d", onlyID[i])
    for j in 1:size(onlyMar,2)
        @printf(Marstream, "%3d", onlyMar[i,j])
    end
    @printf(Marstream, "\n")
end
```

In [91]:
```
close(QTLstream)
close(Marstream)
```

# Remove Fixed Gene from the panel

In [92]:
```
VQM = var(QTLMarker,1)
QMnoFixed = QTLMarker[:,VQM .> 0]
VQ = var(onlyQTL,1)
QnoFixed = onlyQTL[:,VQ .> 0]
VM = var(onlyMar,1)
MnoFixed = onlyMar[:,VM .> 0];
```

In [93]:
```
GenNFstream = open(GenNF, "w")
QTLNFstream = open(QTLNF, "w")
MarNFstream = open(MarNF, "w");
```

In [94]:
```
for i in 1:size(onlyID,1)
    @printf(GenNFstream, "%19d", onlyID[i])
    for j in 1:size(QMnoFixed,2)
        @printf(GenNFstream, "%3d", QMnoFixed[i,j])
    end
    @printf(GenNFstream, "\n")
end
```

In [95]:
```
for i in 1:size(onlyID,1)
    @printf(QTLNFstream, "%19d", onlyID[i])
    for j in 1:size(QnoFixed,2)
        @printf(QTLNFstream, "%3d", QnoFixed[i,j])
    end
    @printf(QTLNFstream, "\n")
end
```

In [96]:
```
for i in 1:size(onlyID,1)
    @printf(MarNFstream, "%19d", onlyID[i])
    for j in 1:size(MnoFixed,2)
        @printf(MarNFstream, "%3d", MnoFixed[i,j])
    end
    @printf(MarNFstream, "\n")
end
```

In [97]:
```
close(GenNFstream)
close(QTLNFstream)
close(MarNFstream)
```

# Check heritability

In [98]:
```
P = AllPBV[:,2]
BV = AllPBV[:,3];
```

In [99]:
```
VP = var(P)
VBV = var(BV)
H = VBV/VP
```

Out[99]: 0.4488126834461731

In [100]:
```
cor=cor(P,BV)
```

```
WARNING: imported binding for cor overwritten in module Main
```

Out[100]:
```
0.67086997274362
```

In [101]:
```
QTLAll = M[:,QTLPos]
```

Out[101]:
```
48000x50 Array{Int64,2}:
 2  2  1  1  1  1  2  1  1  0  0  0  2  …  0  1  0  0  2  2  0  2  2  2  1  0
 2  2  2  1  0  0  1  2  2  0  1  1  2     1  2  0  0  2  2  1  0  2  2  1  0
 1  2  0  1  1  1  1  1  1  1  2  0  1     0  2  0  0  2  2  1  0  2  2  0  0
 2  1  1  2  1  1  1  1  1  0  1  1  1     1  2  0  0  0  1  2  0  1  1  2  1
 1  1  1  1  0  1  2  1  2  0  1  0  1     0  2  1  0  1  2  0  1  0  1  2  1
 2  2  1  1  0  0  1  1  1  1  1  0  2  …  1  2  1  0  2  2  1  0  0  0  2  2
 1  1  1  1  0  1  2  0  0  1  1  0  2     0  2  0  1  1  2  2  1  2  2  1  0
 0  2  0  1  1  0  0  2  2  0  0  0  2     0  2  0  0  0  1  2  0  1  1  2  1
 2  2  0  2  1  1  1  1  1  0  1  1  1     1  2  0  0  1  1  1  0  2  2  1  1
 2  1  1  0  0  0  2  0  0  2  2  1  1     1  2  0  0  2  2  1  0  1  1  1  1
 1  1  0  2  1  2  2  1  1  0  2  1  0  …  1  2  1  0  1  2  2  0  1  2  1  0
 2  2  1  1  1  2  2  0  0  0  2  1  1     0  2  1  1  0  2  1  0  0  1  2  1
 1  2  0  1  1  1  2  0  0  1  1  1  2     1  2  1  0  0  2  0  0  0  0  2  1
 ⋮              ⋮              ⋮        ⋱        ⋮              ⋮
 2  2  0  2  2  1  1  1  1  1  1  1  2     2  2  1  1  2  2  0  0  2  2  1  1
 2  2  0  2  2  1  1  1  1  0  2  2  1     1  2  2  0  1  2  2  1  1  1  2  1
 1  2  1  1  1  0  0  1  1  1  1  1  2  …  1  2  1  1  1  2  1  2  2  2  2  0
 2  2  1  1  0  2  2  0  1  1  2  1  1     2  2  0  1  1  2  1  2  2  2  1  0
 2  2  1  1  1  2  2  1  2  0  1  2  1     1  2  1  2  2  2  2  2  2  2  2  0
 2  2  0  2  1  2  1  0  1  2  2  2  0     1  2  2  0  2  2  2  1  1  1  2  1
 1  2  0  1  1  1  2  1  1  1  2  2  0     1  2  2  2  2  2  2  2  2  2  2  0
 2  2  1  1  2  1  2  2  2  1  2  1  1  …  1  2  2  0  2  2  2  0  2  2  2  1
 2  1  1  2  1  0  1  1  1  1  2  1  1     1  2  1  1  2  2  1  1  2  2  2  1
 2  2  1  1  1  1  2  2  1  2  1  1  1     2  2  0  2  1  2  2  1  2  2  1  1
 2  2  0  2  2  1  1  1  1  1  1  2  2     1  2  0  0  2  2  0  1  1  1  2  1
 2  2  1  1  1  1  1  1  1  0  2  2  0     1  2  0  1  1  2  1  0  2  2  1  0
```

In [102]:
```
QTLo=qtlEffects[QTLPos]
```

Out[102]:
```
50-element Array{Float64,1}:
 0.201061
 0.203387
 0.20199
 0.200407
 0.197343
 0.201207
 0.203563
 0.203694
 0.201339
 0.198529
 0.199915
 0.200385
 0.201765
 ⋮
 0.199313
 0.198165
 0.19909
 0.19982
 0.200038
 0.198461
 0.199302
 0.201571
 0.199277
 0.198335
 0.196437
 0.202678
```

In [103]:
```
EAlpha=QTLAll*QTLo
```

Out[103]:
```
48000-element Array{Float64,1}:
 10.2306
 11.6396
 10.4279
 10.6194
 10.2404
 10.8359
 11.4268
 10.0254
 12.2273
 11.6192
 11.8215
 10.6119
  9.43118
  ⋮
 14.2325
 14.2218
 13.4314
 14.2338
 15.4407
 13.8086
 14.4311
 14.632
 13.6218
 14.8416
 13.6224
 13.2337
```

In [104]:
```
meanEAlphaG0=mean(EAlpha[1:8000])   # our mu_g
```

Out[104]:
```
10.854784170073152
```

In [105]:
```
meanEAlphaG1=mean(EAlpha[8001:16000])
```

Out[105]:
```
11.367365088775948
```

In [106]:
```
meanEAlphaG2=mean(EAlpha[16001:24000])
```

Out[106]:
```
11.765536618031671
```

```
In [107]: meanEAlphaG3=mean(EAlpha[24001:32000])
```

```
Out[107]: 12.15845753943986
```

```
In [108]: meanEAlphaG4=mean(EAlpha[32001:40000])
```

```
Out[108]: 12.60500527833524
```

```
In [109]: meanEAlphaG5=mean(EAlpha[40001:48000])
```

```
Out[109]: 13.080648263437663
```

```
In [110]: EAlphaG=onlyQTL*QTLo
```

```
Out[110]: 9000-element Array{Float64,1}:
           11.6396
           12.6161
           11.8305
           12.8393
           11.8429
           12.4281
           11.6346
           12.2241
           11.6343
           12.0387
           12.0221
           13.0385
           13.0145
            ⋮
           14.2325
           14.2218
           13.4314
           14.2338
           15.4407
           13.8086
           14.4311
           14.632
           13.6218
           14.8416
           13.6224
           13.2337
```

In [111]:
```
meanEAlphaG=mean(EAlphaG)
```

Out[111]: 13.031584480725426

In [112]:
```
meanBreedingValueSelected=meanEAlphaG-meanEAlphaG0   # Legarra mu_g
```

Out[112]: 2.1768003106522738

In [113]:
```
meanEAlphaS0=mean(EAlphaG[1:200])
```

Out[113]: 11.879679389587745

In [114]:
```
meanBreedingValueSelectedG0=meanEAlphaS0-meanEAlphaG0
```

Out[114]: 1.0248952195145922

In [115]:
```
meanEAlphaS1=mean(EAlphaG[201:400])
```

Out[115]: 12.169143865307856

In [116]:
```
meanBreedingValueSelectedG1=meanEAlphaS1-meanEAlphaG0
```

Out[116]: 1.3143596952347032

In [117]:
```
meanEAlphaS2=mean(EAlphaG[401:600])
```

Out[117]: 12.548742119027125

In [118]:
```
meanBreedingValueSelectedG2=meanEAlphaS2-meanEAlphaG0
```

Out[118]: 1.6939579489539724

In [119]:
```
meanEAlphaS3=mean(EAlphaG[601:800])
```

Out[119]: 13.06608982904885

In [120]:
```
meanBreedingValueSelectedG3=meanEAlphaS3-meanEAlphaG0
```

Out[120]: 2.2113056589756983

In [121]:
```
meanEAlphaS4=mean(EAlphaG[801:1000])
```

Out[121]: 13.531715892166083

In [122]:
```
meanBreedingValueSelectedG4=meanEAlphaS4-meanEAlphaG0
```

Out[122]: 2.676931722092931

In [123]:
```
meanEAlphaS5=mean(EAlphaG[1001:9000])
```

Out[123]: 13.080648263437663

In [124]:
```
meanBreedingValueSelectedG5=meanEAlphaS5-meanEAlphaG0
```

Out[124]: 2.2258640933645104