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 = 0.1
# 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

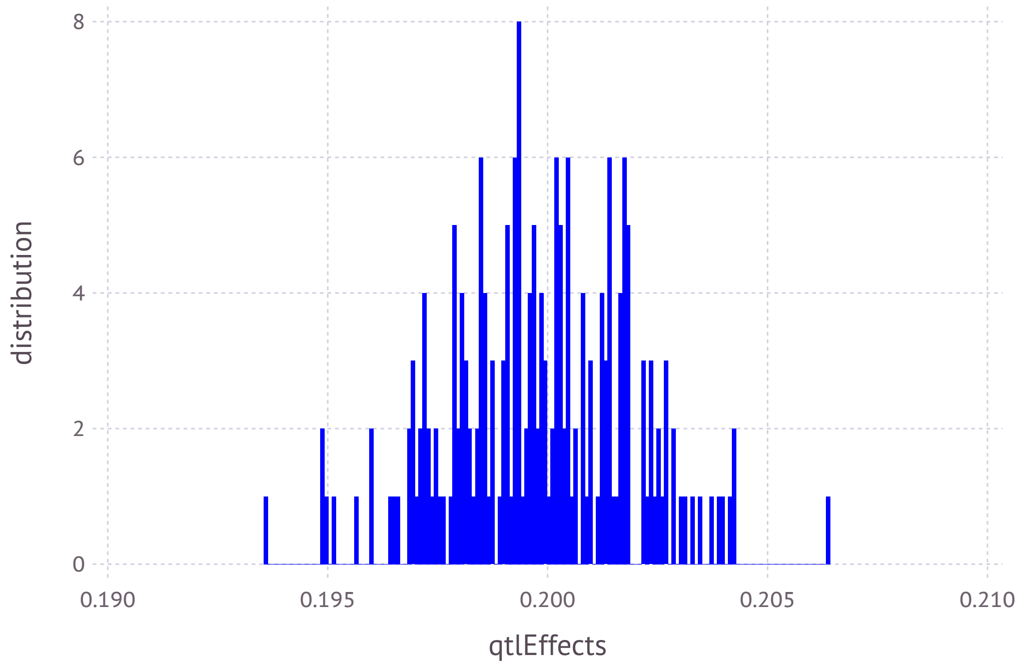
In [6]:
```
numChr    = 10
chrLength = 0.01
numLoci   = 20
nQTL      = 5
mutRate   = 0.0
locusInt  = chrLength/numLoci
mapPos    = collect(locusInt/2:locusInt:chrLength)
geneFreq  = fill(0.5,numLoci)
QTL = sample(1:numLoci,nQTL,replace=false)
qtlMarker = fill(false,numLoci)
qtlMarker[QTL] = true
mu = 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 = 0.115
XSim.init(numChr,numLoci,chrLength,geneFreq,mapPos,qtlMarker,qtlEffects,mutRate)
```

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

```
Out[7]: 200-element Array{Float64,1}:
         0.201779
         0.201652
         0.199301
         0.199022
         0.199224
         0.198763
         0.198298
         0.197873
         0.202871
         0.202701
         0.200368
         0.19836
         0.19707
         ⋮
         0.197344
         0.198441
         0.196908
         0.197155
         0.200435
         0.199718
         0.19791
         0.199728
         0.199382
         0.199581
         0.197946
         0.198503
```

```
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]: 0.19980190125607336
```

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

Out[11]: 4.32302413895192e-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
```

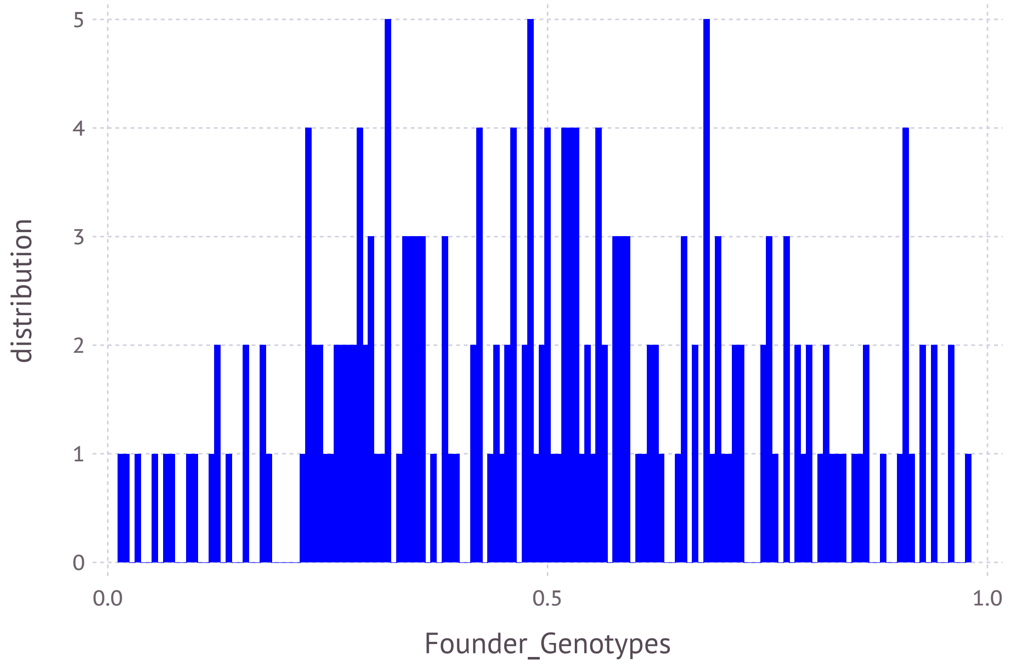
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
         gSPDam = XSim.getOurGenotypes(popSP[2])
         gSP = [gSPSire;gSPSire];
```

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

Out[20]: 1x200 Array{Float64,2}:
          0.06725  0.8355  0.288125  0.942  0.817625  …  0.386625  0.904125  0.532375
```

```
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);
```

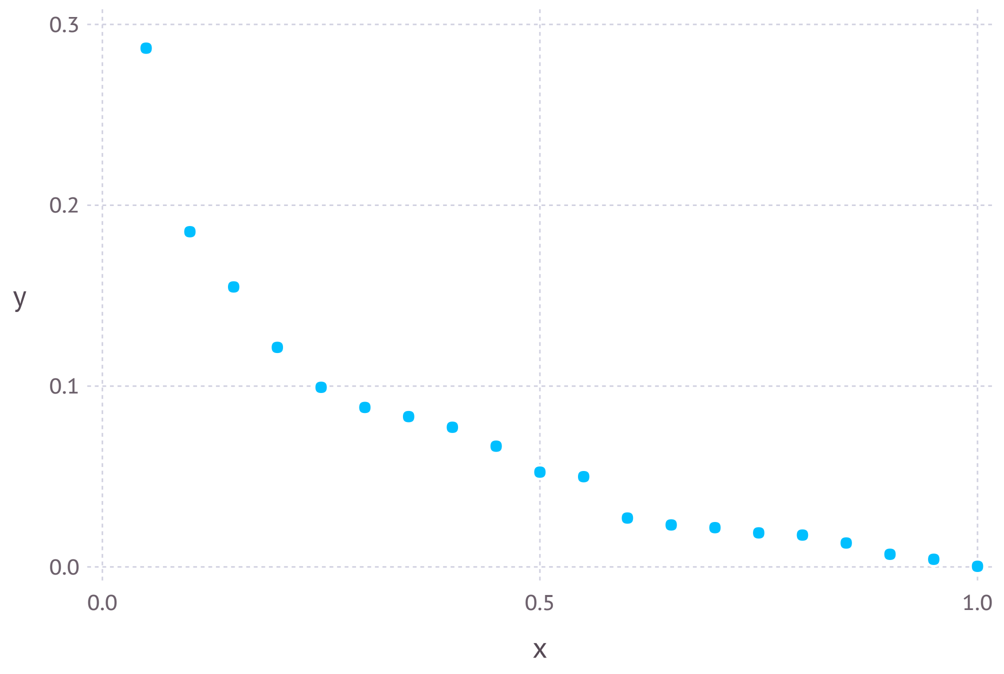
In [23]:
```
for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

Out[24]:
```
1x20 Array{Float64,2}:
 0.000338771  0.00423232  0.0070215  …  0.154833  0.18539  0.286848
```

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]:
```
9.452028023464276
```

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

Out[31]:
```
0.8711064548172242
```

In [32]:
```
XSim.common.varRes = 9*varGen    #heritability = 0.1
```

Out[32]:
```
7.839958093355017
```

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

Out[33]:
```
7.839958093355017
```

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]: 11.195627531517506

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

Out[36]: 11.194031626267199

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

Out[37]: 0.7971101621095362

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

Out[38]: 0.8378297272839439
```

Pedigree: All animals

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

Out[39]: 48000x3 Array{Int64,2}:
          40722  34133  37664
          40723  33080  38232
          40724  36702  39968
          40725  34112  37677
          40726  35843  38722
          40727  35681  36879
          40728  33218  39116
          40729  34223  38838
          40730  36314  38913
          40731  34588  38454
          40732  33545  37696
          40733  33888  37977
          40734  35655  39897
              ⋮
          88710  73921  77196
          88711  75575  79569
          88712  74933  78920
          88713  74591  77921
          88714  75948  78168
          88715  76224  80523
          88716  75461  80334
          88717  73586  79632
          88718  75453  76924
          88719  73454  80653
          88720  73643  78835
          88721  73747  79920
```

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

```
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}:
 1  2  0  2  2  1  1  1  1  1  1  1  1  …  2  0  1  2  0  2  1  1  1  0  2  2
 1  2  0  2  2  0  0  0  2  0  0  1  0     2  1  1  1  1  2  0  1  1  0  2  1
 0  2  1  2  1  0  0  0  2  0  0  2  1     1  1  2  1  1  1  1  1  1  1  2  1
 0  2  0  2  2  0  1  1  2  0  0  2  2     2  2  2  0  2  2  0  2  2  0  2  0
 0  1  1  1  1  0  1  1  1  1  1  1  2     2  1  1  1  2  2  0  1  1  1  2  0
 0  1  1  1  1  0  1  1  1  1  1  0  1  …  0  0  2  2  0  0  2  0  0  2  2  2
 0  2  0  2  2  0  0  0  2  0  0  2  2     2  1  2  2  2  2  0  1  1  1  2  1
 0  2  0  2  2  1  1  1  1  1  1  1  2     1  1  2  1  1  1  1  1  1  1  2  1
 0  2  0  2  2  0  1  0  2  0  0  0  0     2  1  2  1  2  2  0  1  1  1  1  1
 0  2  0  2  2  1  1  1  1  1  1  1  2     2  2  2  1  2  2  0  1  1  1  2  0
 0  2  0  2  2  0  0  0  2  0  0  1  1  …  2  2  2  1  2  2  0  1  1  1  2  0
 0  2  0  2  2  0  0  0  2  0  0  2  2     1  0  2  2  1  1  1  1  1  1  2  1
 0  2  0  2  2  1  1  1  1  1  1  1  2     2  1  1  1  1  2  0  1  1  0  2  1
 ⋮              ⋮              ⋮        ⋱        ⋮              ⋮
 0  1  1  1  1  0  1  1  1  1  1  0  1     2  0  1  2  2  2  0  0  0  2  1  1
 0  1  1  1  2  0  2  2  1  1  1  0  0     2  2  2  0  2  2  0  1  1  1  1  1
 0  1  1  2  2  1  1  1  1  1  1  0  1  …  2  0  1  1  1  2  1  0  0  2  0  2
 0  1  1  2  2  1  1  1  1  1  1  1  2     2  0  1  2  1  2  0  0  0  1  1  2
 0  2  0  2  2  0  0  0  2  0  0  0  0     2  1  2  2  2  2  0  0  0  2  1  1
 1  2  0  2  2  1  1  1  1  1  1  1  2     1  0  2  2  2  1  1  1  1  1  2  1
 0  2  1  2  1  1  1  1  1  1  1  1  1     2  1  2  1  2  2  0  1  1  1  1  1
 0  2  1  2  1  0  0  0  2  0  0  1  1  …  1  0  1  2  0  1  1  0  0  1  2  2
 0  2  0  2  2  0  1  0  2  0  0  1  1     2  1  2  2  2  2  0  0  0  1  1  1
 0  1  1  2  2  0  1  1  1  1  1  0  0     2  1  2  1  2  2  0  1  1  1  1  1
 0  2  0  2  2  0  1  0  2  0  0  2  2     2  1  1  1  1  1  1  1  1  1  2  1
 0  2  1  2  1  0  1  1  2  0  0  2  1     2  2  2  0  2  2  0  2  2  0  2  0
```

```
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}:
          42423
          44010
          42537
          43159
          42112
          42308
          41678
          42264
          43911
          41867
          41678
          42954
          44099
              ⋮
          73921
          75575
          74933
          74591
          75948
          76224
          75461
          73586
          75453
          73454
          73643
          73747
```

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

```
Out[53]: 1000-element Array{Int64,1}:
          42423
          44010
          42537
          43159
          42112
          42308
          41678
          42264
          43911
          41867
          42954
          44099
          43613
              ⋮
          74589
          73786
          76483
          74017
          76412
          74821
          74756
          74992
          74683
          74199
          73404
          76348
```

```
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}:
          42423
          44010
          42537
          43159
          42112
          42308
          41678
          42264
          43911
          41867
          42954
          44099
          43613
              ⋮
          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  12.169  11.359
          40723   9.801   9.985
          40724  11.578  11.151
          40725  11.049   9.763
          40726  11.703  10.359
          40727   8.053   8.175
          40728   6.834   9.178
          40729   9.79    9.558
          40730  10.223  10.173
          40731   9.576  10.743
          40732  10.401   9.762
          40733  11.783  10.759
          40734   9.798  10.179
               ⋮
          88710   9.145  10.974
          88711  10.082  10.361
          88712   9.673  10.558
          88713  13.106  11.356
          88714  18.308  10.373
          88715   9.161  12.563
          88716  13.924  11.751
          88717   7.093  10.359
          88718   6.733  12.752
          88719   9.206  10.969
          88720   7.803  11.564
          88721  11.729  11.563

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}:
            1
            2
            6
           13
           18
           21
           22
           26
           33
           38
           41
           42
           46
            ⋮
          153
          158
          161
          162
          166
          173
          178
          181
          182
          186
          193
          198
```

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

Out[83]:
```
150-element Array{Int64,1}:
   3
   4
   5
   7
   8
   9
  10
  11
  12
  14
  15
  16
  17
   ⋮
 187
 188
 189
 190
 191
 192
 194
 195
 196
 197
 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}:
 1  2  0  2  2  1  1  1  1  1  1  1  1  …  2  0  1  2  0  2  1  1  1  0  2  2
 0  1  1  1  1  0  1  1  1  1  1  1  2     2  1  1  1  2  2  0  1  1  1  2  0
 0  2  0  2  2  0  0  0  2  0  0  2  2     1  0  2  2  1  1  1  1  1  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  2  2     2  1  2  2  1  2  0  0  0  1  1  1
 0  2  0  2  2  0  1  1  1  1  1  1  2     0  0  2  2  1  1  1  0  0  1  2  2
 0  2  1  2  1  1  1  1  1  1  0  1  1  …  2  1  2  1  1  2  0  1  1  0  1  1
 0  2  0  2  2  1  1  1  1  1  1  1  2     2  1  1  2  1  1  1  0  0  1  2  2
 1  2  0  2  2  2  2  2  0  2  2  0  2     1  1  2  2  1  1  1  0  0  2  2  2
 0  2  1  2  1  0  0  0  2  0  0  2  1     2  1  2  1  2  1  1  1  1  1  2  1
 0  1  1  2  2  2  2  2  0  2  2  0  2     2  0  1  2  1  1  1  0  0  1  2  2
 0  2  0  2  2  0  1  1  2  0  0  2  2  …  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  0  2  2  1  2  2  1  1  1  1  2     2  2  2  1  2  2  0  1  1  0  2  0
 0  2  0  2  2  0  1  1  2  0  0  1  1     2  2  2  0  2  2  0  2  2  0  2  0
 ⋮              ⋮              ⋮        ⋱        ⋮              ⋮
 0  1  1  1  1  0  1  1  1  1  1  0  1     2  0  1  2  2  2  0  0  0  2  1  1
 0  1  1  1  2  0  2  2  1  1  1  0  0     2  2  2  0  2  2  0  1  1  1  1  1
 0  1  1  2  2  1  1  1  1  1  1  0  1  …  2  0  1  1  1  2  1  0  0  2  0  2
 0  1  1  2  2  1  1  1  1  1  1  1  2     2  0  1  2  1  2  0  0  0  1  1  2
 0  2  0  2  2  0  0  0  2  0  0  0  0     2  1  2  2  2  2  0  0  0  2  1  1
 1  2  0  2  2  1  1  1  1  1  1  1  2     1  0  2  2  2  1  1  1  1  1  2  1
 0  2  1  2  1  1  1  1  1  1  1  1  1     2  1  2  1  2  2  0  1  1  1  1  1
 0  2  1  2  1  0  0  0  2  0  0  1  1  …  1  0  1  2  0  1  1  0  0  1  2  2
 0  2  0  2  2  0  1  0  2  0  0  1  1     2  1  2  2  2  2  0  0  0  1  1  1
 0  1  1  2  2  0  1  1  1  1  1  0  0     2  1  2  1  2  2  0  1  1  1  1  1
 0  2  0  2  2  0  1  0  2  0  0  2  2     2  1  1  1  1  1  1  1  1  1  2  1
 0  2  1  2  1  0  1  1  2  0  0  2  1     2  2  2  0  2  2  0  2  2  0  2  0
```

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

```
Out[86]: 9000x50 Array{Int64,2}:
  1  2  1  1  0  1  1  0  1  1  1  1  2  …  1  2  2  0  2  0  1  1  1  0  0  0
  0  1  0  2  1  2  1  0  1  2  1  1  1     2  1  2  0  2  0  1  2  1  1  2  1
  0  2  0  2  1  2  1  0  1  2  2  0  0     2  2  2  0  0  0  2  2  0  1  1  1
  0  2  0  2  1  2  2  0  2  2  2  0  1     2  1  2  0  1  0  1  1  1  1  1  1
  0  2  0  2  1  1  0  0  1  1  0  2  2     2  0  1  0  2  0  2  2  0  0  1  1
  0  2  1  1  0  2  2  0  2  2  1  2  2  …  2  1  1  1  1  0  2  2  0  2  1  0
  0  2  1  2  1  2  1  1  2  1  2  1  2     2  2  2  1  0  0  1  1  2  1  1  1
  1  2  2  2  0  2  2  1  1  2  0  2  1     1  1  0  2  1  0  0  2  0  1  1  2
  0  2  0  1  2  1  1  0  2  1  0  2  2     1  2  2  0  2  0  0  2  1  2  2  1
  0  1  2  2  0  2  1  0  1  2  1  1  2     1  1  1  1  2  1  1  2  1  2  1  1
  0  2  0  2  1  2  0  0  1  2  2  1  1  …  2  2  0  1  1  0  2  2  0  1  1  0
  0  2  1  2  1  2  0  0  0  2  2  0  0     2  2  2  0  2  0  2  1  1  2  2  0
  0  2  0  1  1  1  2  0  2  2  1  1  0     2  2  0  1  1  0  1  2  0  2  2  0
  ⋮              ⋮              ⋮        ⋱        ⋮              ⋮
  0  1  0  1  0  1  2  1  2  1  1  2  2     1  1  1  1  1  0  2  2  2  1  2  2
  0  1  0  0  1  1  1  1  1  0  0  2  2     2  2  2  0  1  0  1  1  1  1  2  1
  0  1  1  1  0  2  1  1  1  2  1  1  0  …  2  2  1  1  1  0  1  2  1  0  1  2
  0  1  1  2  1  2  0  1  0  2  0  1  2     2  1  2  0  2  0  2  2  0  1  1  1
  0  2  0  0  0  2  0  0  0  2  1  1  1     1  1  2  1  0  0  2  2  0  2  2  2
  1  2  1  2  1  2  1  0  2  2  1  1  1     2  1  2  0  0  0  2  2  1  1  2  1
  0  2  1  1  0  2  2  0  2  2  2  0  2     2  2  2  0  2  0  0  2  1  2  2  1
  0  2  0  1  0  1  1  0  1  1  1  1  2  …  1  2  0  0  2  0  2  2  0  0  0  1
  0  2  0  1  1  2  2  0  2  2  1  1  2     1  2  2  0  0  1  1  1  2  2  2  1
  0  1  0  0  0  2  1  0  1  2  2  0  0     2  1  1  1  1  0  1  2  1  2  2  1
  0  2  0  2  2  2  2  1  1  2  0  2  2     2  2  0  2  1  0  0  2  1  1  1  1
  0  2  0  1  1  2  1  1  1  2  2  0  1     2  0  2  0  2  0  1  2  0  2  2  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]:
```
0.14317890647778117
```

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

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

```
Out[100]: 0.3740538471781798
```

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

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

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

```
Out[102]: 50-element Array{Float64,1}:
           0.201779
           0.201652
           0.198763
           0.19707
           0.197193
           0.199313
           0.199271
           0.198483
           0.199378
           0.199365
           0.199877
           0.203093
           0.197218
           ⋮
           0.199114
           0.202658
           0.203063
           0.19862
           0.199992
           0.19927
           0.201623
           0.201832
           0.198531
           0.204223
           0.200435
           0.199581
```

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

Out[103]:
```
48000-element Array{Float64,1}:
 11.4148 
 10.0088 
 11.1998 
  9.83233
 10.3948 
  8.20405
  9.20403
  9.6107 
 10.2218 
 10.7969 
  9.81656
 10.8104 
 10.2166 
  ⋮      
 10.9933 
 10.405  
 10.6128 
 11.4002 
 10.4117 
 12.6076 
 11.8096 
 10.401  
 12.8122 
 11.0029 
 11.5826 
 11.5817 
```

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

Out[104]:
```
9.488768740333251
```

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

Out[105]:
```
10.149502704338898
```

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

Out[106]:
```
10.42919780220928
```

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

Out[107]: 10.6852057309698

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

Out[108]: 10.95602415864258

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

Out[109]: 11.238249218937293

In [110]: EAlphaG=onlyQTL*QTLo

Out[110]: 9000-element Array{Float64,1}:
           11.4148
           10.3948
           10.8104
           10.9951
           11.195
           11.2077
           12.6021
           11.2007
           11.6117
           12.0018
           10.4073
           11.2159
           10.6067
             ⋮
           10.9933
           10.405
           10.6128
           11.4002
           10.4117
           12.6076
           11.8096
           10.401
           12.8122
           11.0029
           11.5826
           11.5817
```

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

Out[111]: 11.214941664924627

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

Out[112]: 1.726172924591376

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

Out[113]: 10.805048689655164

In [114]: meanBreedingValueSelectedG0=meanEAlphaS0-meanEAlphaG0

Out[114]: 1.3162799493219133

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

Out[115]: 10.679284464273355

In [116]: meanBreedingValueSelectedG1=meanEAlphaS1-meanEAlphaG0

Out[116]: 1.1905157239401039

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

Out[117]: 10.957080472440587

In [118]: meanBreedingValueSelectedG2=meanEAlphaS2-meanEAlphaG0

Out[118]: 1.4683117321073365

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

Out[119]: 11.203544607788853

In [120]: meanBreedingValueSelectedG3=meanEAlphaS3-meanEAlphaG0

Out[120]: 1.714775867455602
```

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

Out[121]: 11.497447929958703

In [122]: meanBreedingValueSelectedG4=meanEAlphaS4-meanEAlphaG0

Out[122]: 2.008679189625452

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

Out[123]: 11.238249218937293

In [124]: meanBreedingValueSelectedG5=meanEAlphaS5-meanEAlphaG0

Out[124]: 1.7494804786040419
```