

```
In [1]: # Founders: real haplotype data (chlto10.200SNP)
# "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                                     #  $\alpha \sim N(100,1)$ 
        Va = nQTL*numChr*0.5*mu*mu                  #  $Va = nQTL * 2pq * \text{mean}(\alpha)^2$ 
        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)
```

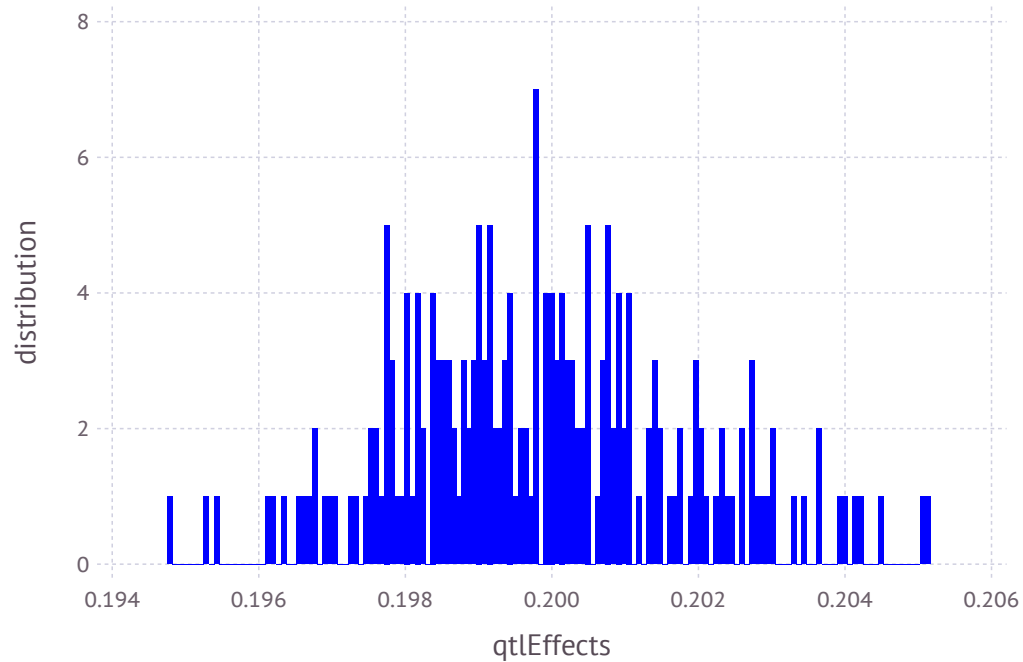
```
In [7]: qtlEffects
```

```
Out[7]: 200-element Array{Float64,1}:  
 0.197279  
 0.19757  
 0.200444  
 0.200741  
 0.201354  
 0.196106  
 0.196316  
 0.197715  
 0.202846  
 0.198392  
 0.199771  
 0.202747  
 0.202058  
  ⋮  
 0.199777  
 0.199417  
 0.199441  
 0.200514  
 0.202283  
 0.200735  
 0.198506  
 0.197762  
 0.196586  
 0.198053  
 0.199088  
 0.200105
```

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

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

```
Out[9]:
```



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

```
Out[10]: 0.19985347811041687
```

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

```
Out[11]: 3.824736978729179e-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 animals
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
```

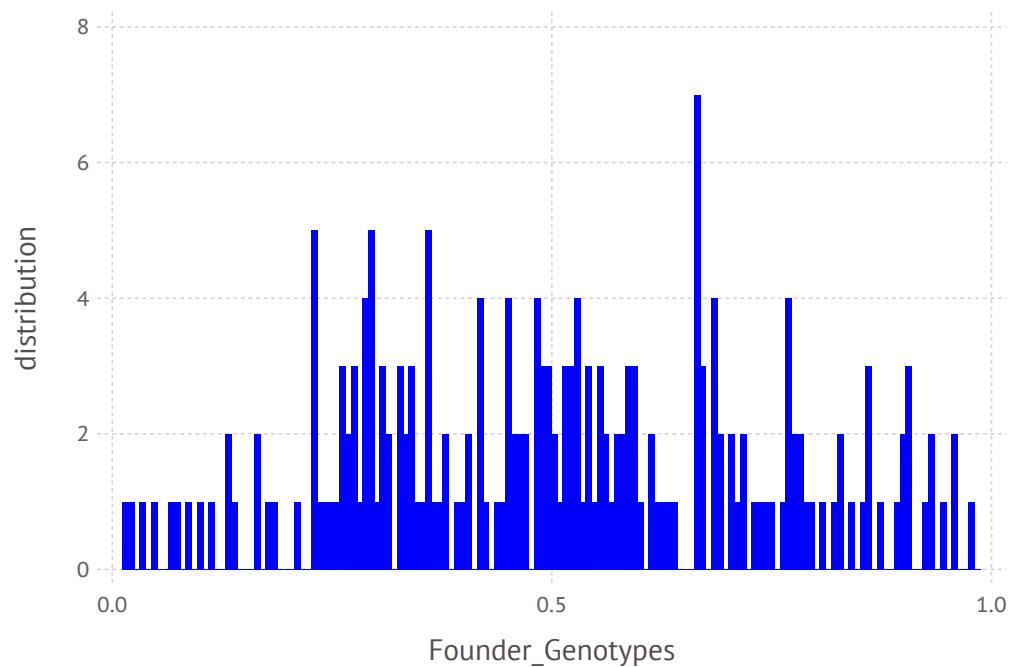
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

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

```
Out[20]: 1x200 Array{Float64,2}:
 0.065875  0.840625  0.27825  0.94725  ...  0.393875  0.90525  0.545125
```

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

```
Out[21]:
```



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



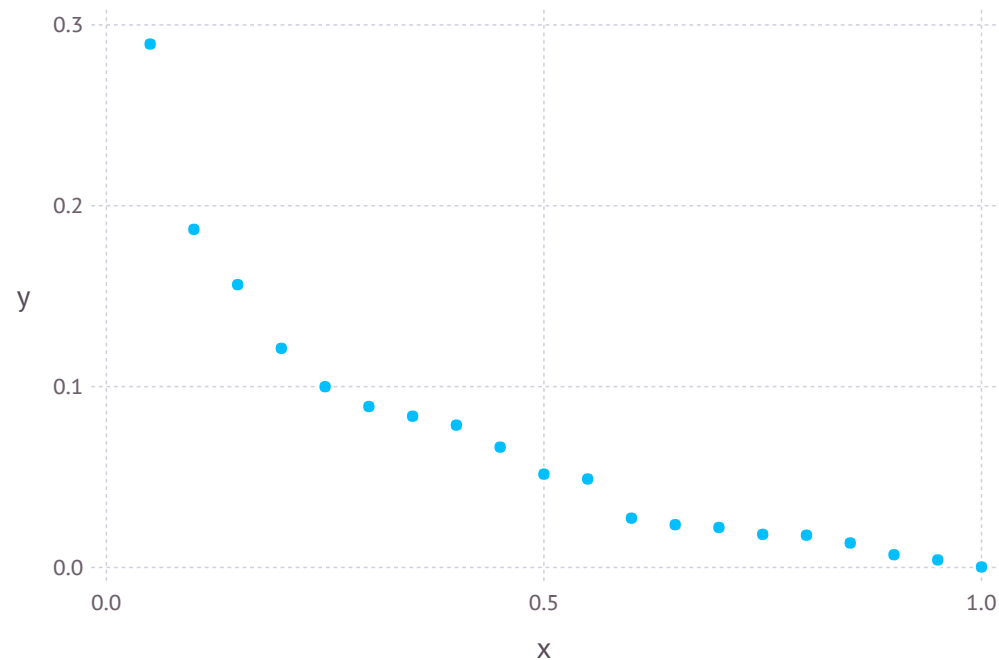
```
In [23]: for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

```
Out[24]: 1x20 Array{Float64,2}:
 0.000251486  0.004183  0.00706938 ...  0.156381  0.186965  0.289398
```

```
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]: 8.975929847646396
```

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

```
Out[31]: 0.5678471155020868
```

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

```
Out[32]: 5.110624039518781
```

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

```
Out[33]: 5.110624039518781
```

```
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]: 10.472922418732331
```

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

```
Out[36]: 10.467097685842877
```

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

```
Out[37]: 0.5727902920261992
```

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

```
Out[38]: 0.5700159660087326
```

## Pedigree: All animals

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

```
Out[39]: 48000x3 Array{Int64,2}:  
  40722  32874  37661  
  40723  36218  38593  
  40724  35055  39569  
  40725  33980  38993  
  40726  35086  38978  
  40727  34212  39392  
  40728  33245  39155  
  40729  34785  37366  
  40730  35098  40247  
  40731  32800  39300  
  40732  36417  38114  
  40733  33429  38664  
  40734  36691  37925  
      ⋮  
  88710  75043  78992  
  88711  75498  78747  
  88712  72752  80357  
  88713  75791  78060  
  88714  75174  77731  
  88715  74296  79252  
  88716  75634  77846  
  88717  75786  78979  
  88718  74608  79126  
  88719  72879  79388  
  88720  74703  79216  
  88721  73350  77747
```

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

```
41474
44324
43075
43292
40738
43936
44214
43249
42734
43066
44564
42263
42099
      :
75043
75498
72752
75791
75174
74296
75634
75786
74608
72879
74703
73350
```



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

```
Out[53]: 1000-element Array{Int64,1}:
```

```
41474
44324
43075
43292
40738
43936
44214
43249
42734
43066
44564
42263
42099
      :
73311
73124
74315
75717
76292
75700
75147
75236
76178
74622
74641
76136
```

```
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}:  
 41474  
 44324  
 43075  
 43292  
 40738  
 43936  
 44214  
 43249  
 42734  
 43066  
 44564  
 42263  
 42099  
      ⋮  
 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  10.105   9.966  
  40723   8.084   9.572  
  40724   8.898   8.779  
  40725   5.931   7.607  
  40726   7.998   8.996  
  40727   7.78    7.205  
  40728   6.653   8.376  
  40729   8.646   8.997  
  40730   7.989   9.56  
  40731   6.644   8.368  
  40732   8.178   7.596  
  40733   9.362   9.199  
  40734   8.153   8.972  
      ⋮  
  88710  10.547  10.969  
  88711  12.1    11.179  
  88712   9.136   9.773  
  88713   6.738   9.379  
  88714  11.171  10.984  
  88715   8.755   9.986  
  88716  13.074  10.562  
  88717   9.904  11.165  
  88718   9.508  11.356  
  88719   8.829  10.167  
  88720  14.253  11.162  
  88721  12.626   9.973
```

```
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 animals 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}:
```

```
 5  
 6  
 7  
13  
15  
25  
26  
27  
33  
35  
45  
46  
47  
:  
153  
155  
165  
166  
167  
173  
175  
185  
186  
187  
193  
195
```



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

```
Out[83]: 150-element Array{Int64,1}:
```

```
 1
 2
 3
 4
 8
 9
10
11
12
14
16
17
18
 ⋮
184
188
189
190
191
192
194
196
197
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  0  0  2  0  0  1  1  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  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  1  1  1  2  2  1  2  2  0  2  1
 0  2  0  2  2  2  2  2  0  2  1  0  2  ...  0  0  2  2  0  0  2  0  0  2  2  2
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  2  2  1  1  1  1  1  1  1  1  2  1
 0  1  1  2  2  0  0  0  2  0  0  2  2  ...  2  2  1  1  2  2  0  1  1  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  1  2  1  1  1  0  1  1  0  2  1
 0  2  1  2  1  0  1  1  1  1  1  0  1  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  1  1  2  2  0  2  2  1  1  1  1  2  ...  2  1  1  1  2  2  0  1  1  1  2  0
 0  2  1  2  1  0  0  0  2  0  0  1  1  ...  2  1  2  1  1  1  0  1  1  0  2  1
 0  2  1  2  1  0  0  0  2  0  0  1  0  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  2  0  2  2  2  2  2  0  2  1  0  1  ...  2  0  0  2  1  2  1  0  0  2  1  1
 0  1  1  1  1  1  2  2  0  2  2  0  2  ...  1  0  2  2  0  0  1  0  0  1  2  2
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 0  0  2  2  2  1  2  2  0  2  2  0  1  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  2  0  2  2  0  1  1  1  1  1  0  1  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  1  2  1  1  1  1  1  1  1  1  1  ...  0  0  2  2  0  0  2  0  0  2  2  2
 0  1  1  2  2  1  1  1  1  1  1  0  1  ...  2  1  1  1  2  2  0  1  1  1  2  0
 0  2  0  2  2  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  1  1  1  1  1  1  1  2  ...  2  1  1  2  1  1  1  0  0  1  2  2
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  0  0  2  0  2  1  0  0  1  1  2
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  2  2  1  2  2  0  1  1  1  1  1
 0  2  0  2  2  1  1  1  1  1  1  1  1  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  1  1  1  1  1  1  1  1  ...  2  2  1  1  1  1  1  1  1  1  2  1
 1  2  0  2  2  1  1  1  1  1  1  1  2  ...  2  2  2  0  2  2  0  2  2  0  2  0
```

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

```
Out[86]: 9000x50 Array{Int64,2}:
```

```
 2  0  0  1  1  0  0  0  0  2  0  2  0  ...  2  0  0  2  1  0  1  1  1  1  1  1
 2  0  0  1  1  1  0  0  1  1  1  2  0  ...  2  1  0  2  2  0  2  1  1  1  1  1
 2  0  0  0  2  2  0  2  1  1  0  0  1  ...  2  0  0  1  0  1  1  1  2  1  2  1
 2  2  2  2  2  1  0  0  1  1  0  1  0  ...  1  2  0  1  0  1  0  2  0  0  0  2
 1  0  0  1  1  2  0  1  1  1  1  1  0  ...  2  0  1  1  0  1  0  1  2  1  1  1
 2  0  0  2  0  2  1  1  1  1  0  0  2  ...  2  1  0  2  2  0  2  1  1  2  2  0
 2  0  0  0  1  2  0  0  1  1  0  1  1  ...  2  1  0  2  2  0  1  1  1  2  1  0
 1  0  1  1  2  0  0  1  2  0  1  1  1  ...  2  0  0  2  0  0  0  1  1  1  1  1
 2  0  2  2  0  0  1  1  1  1  0  1  1  ...  1  0  0  0  1  0  1  1  1  2  2  0
 1  0  0  1  1  1  1  2  1  1  1  1  1  ...  2  1  0  1  0  0  1  1  1  2  1  0
 1  0  0  0  2  1  1  2  1  1  1  1  1  ...  1  0  0  1  1  0  2  1  1  1  1  1
 2  2  2  1  1  2  1  2  0  2  0  1  0  ...  2  1  0  0  0  0  1  2  0  2  1  1
 1  1  2  2  1  2  1  1  0  2  1  1  1  ...  2  0  0  1  2  0  1  2  0  1  0  1
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 2  1  2  1  1  1  1  2  0  2  1  2  0  ...  1  2  1  1  0  1  0  0  2  2  2  0
 2  0  0  1  1  2  0  0  2  0  1  1  1  ...  2  2  0  2  2  0  2  1  1  1  1  1
 2  0  1  1  2  1  0  2  0  2  0  0  2  ...  2  1  0  2  1  0  1  0  2  2  2  0
 1  1  1  1  1  1  0  0  1  1  1  2  0  ...  1  1  1  1  1  0  1  2  0  0  0  2
 2  1  1  1  2  2  0  1  1  1  1  2  0  ...  2  1  0  2  1  0  0  1  1  2  2  0
 2  0  0  1  1  2  0  1  0  2  1  1  0  ...  1  1  1  1  0  1  0  2  0  1  0  1
 2  1  1  2  0  2  2  2  0  2  1  2  0  ...  2  1  0  1  0  0  2  0  2  2  1  1
 2  0  0  0  1  1  1  2  2  1  1  2  0  ...  0  2  0  2  2  1  2  2  0  1  0  1
 2  0  0  0  2  2  1  1  1  1  2  2  0  ...  2  1  0  2  2  0  2  0  2  2  2  0
 2  1  1  1  1  1  1  1  1  1  0  1  1  ...  1  2  1  0  1  0  1  0  2  2  2  0
 2  1  1  1  1  2  1  1  1  1  2  2  0  ...  1  2  0  2  0  0  0  1  2  1  1  1
 2  1  1  2  1  1  0  2  0  2  2  1  0  ...  1  0  0  1  2  0  2  1  1  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]: 0.15490653032207768
```

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

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

```
Out[100]: 0.3996361631884671
```

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

```
Out[101]: 48000x50 Array{Int64,2}:
```

```
 2  0  1  0  1  1  2  2  1  2  1  2  0  ...  2  0  0  2  2  0  1  1  1  2  1  0
 2  0  2  0  2  0  1  1  1  2  0  1  1      0  2  0  0  0  0  0  0  2  2  2  0
 1  0  0  0  2  1  0  1  1  1  0  1  0      2  0  0  1  0  0  0  1  1  2  2  0
 2  0  1  0  1  1  0  0  1  1  0  0  1      2  2  1  0  1  0  0  2  0  1  0  2
 1  0  0  1  1  1  0  0  1  1  0  1  1      2  0  1  1  2  0  1  1  1  2  2  0
 2  0  0  0  1  1  0  1  1  1  0  1  1  ...  2  0  1  0  0  0  1  2  0  0  0  0
 1  0  0  0  2  1  1  2  2  1  0  1  1      2  1  1  0  0  1  0  1  1  1  1  1
 2  0  1  1  0  2  0  1  1  1  0  1  1      2  0  0  2  1  0  1  2  0  1  1  1
 2  0  0  0  2  1  1  1  1  1  0  1  1      1  1  1  1  0  0  1  0  2  2  2  0
 2  0  0  1  0  1  0  1  1  1  1  0  1      1  2  1  1  0  0  0  0  2  2  1  1
 2  0  0  2  0  1  0  1  1  1  0  0  2  ...  1  1  0  1  1  0  1  1  1  2  2  0
 2  0  0  1  1  2  0  1  2  0  1  1  0      1  1  1  0  0  1  0  1  1  1  1  1
 2  1  1  1  1  0  0  0  0  2  1  1  1      1  1  1  1  1  0  1  2  0  1  0  1
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 2  1  2  1  1  1  1  2  0  2  1  2  0      1  2  1  1  0  1  0  0  2  2  2  0
 2  0  0  1  1  2  0  0  2  0  1  1  1      2  2  0  2  2  0  2  1  1  1  1  1
 2  0  1  1  2  1  0  2  0  2  0  0  2  ...  2  1  0  2  1  0  1  0  2  2  2  0
 1  1  1  1  1  1  0  0  1  1  1  2  0      1  1  1  1  1  0  1  2  0  0  0  2
 2  1  1  1  2  2  0  1  1  1  1  2  0      2  1  0  2  1  0  0  1  1  2  2  0
 2  0  0  1  1  2  0  1  0  2  1  1  0      1  1  1  1  0  1  0  2  0  1  0  1
 2  1  1  2  0  2  2  2  0  2  1  2  0      2  1  0  1  0  0  2  0  2  2  1  1
 2  0  0  0  1  1  1  2  2  1  1  2  0  ...  0  2  0  2  2  1  2  2  0  1  0  1
 2  0  0  0  2  2  1  1  1  1  2  2  0      2  1  0  2  2  0  2  0  2  2  2  0
 2  1  1  1  1  1  1  1  1  1  0  1  1      1  2  1  0  1  0  1  0  2  2  2  0
 2  1  1  1  1  2  1  1  1  1  2  2  0      1  2  0  2  0  0  0  1  2  1  1  1
 2  1  1  2  1  1  0  2  0  2  2  1  0      1  0  0  1  2  0  2  1  1  2  2  0
```

```
In [102]: QTL=qtEffects[QTLPos]
```

```
Out[102]: 50-element Array{Float64,1}:
```

```
0.201354  
0.196106  
0.196316  
0.202058  
0.200231  
0.197574  
0.198349  
0.199645  
0.199598  
0.202077  
0.201009  
0.202308  
0.198572  
⋮  
0.200268  
0.201481  
0.201055  
0.203665  
0.197999  
0.199979  
0.199093  
0.198786  
0.198607  
0.196989  
0.202283  
0.198506
```

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

```
Out[103]: 48000-element Array{Float64,1}:  
  9.98829  
  9.58343  
  8.78823  
  7.58417  
  8.98518  
  7.19487  
  8.39089  
  8.98371  
  9.59453  
  8.39871  
  7.59105  
  9.18887  
  8.99896  
  ⋮  
 10.9905  
 11.1997  
  9.79669  
  9.3798  
 10.9875  
  9.99173  
 10.5801  
 11.1934  
 11.398  
 10.1812  
 11.1975  
  9.98693
```

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

```
Out[104]: 8.986353827636341
```

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

```
Out[105]: 9.454270482905148
```

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

```
Out[106]: 9.70532706191817
```



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

```
Out[107]: 9.95248525005388
```

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

```
Out[108]: 10.223966711488368
```

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

```
Out[109]: 10.492205539328737
```

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

```
Out[110]: 9000-element Array{Float64,1}:
```

```
 9.60563
10.3997
 9.79035
10.1857
 9.59279
10.5874
 8.98573
 9.39045
 9.17919
 9.38558
 9.78558
10.1688
10.3783
  ⋮
10.9905
11.1997
 9.79669
 9.3798
10.9875
 9.99173
10.5801
11.1934
11.398
10.1812
11.1975
 9.98693
```

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

```
Out[111]: 10.46710791310378
```

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

```
Out[112]: 1.4807540854674386
```

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

```
Out[113]: 9.920281432665295
```

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

```
Out[114]: 0.9339276050289538
```

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

```
Out[115]: 9.97548158981956
```

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

```
Out[116]: 0.9891277621832195
```

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

```
Out[117]: 10.194924241667405
```

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

```
Out[118]: 1.2085704140310636
```

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

```
Out[119]: 10.475236205803142
```

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

```
Out[120]: 1.4888823781668012
```

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

```
Out[121]: 10.765711046565146
```

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

```
Out[122]: 1.779357218928805
```

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

```
Out[123]: 10.492205539328737
```

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

```
Out[124]: 1.5058517116923955
```