

```
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 = 0.3
# 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                                     #  $\alpha \sim N(100,1)$ 
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  $\alpha \mu = 0.2$ 
XSim.init(numChr,numLoci,chrLength,geneFreq,mapPos,qtlMarker,qtlEffects,mutRate)
```

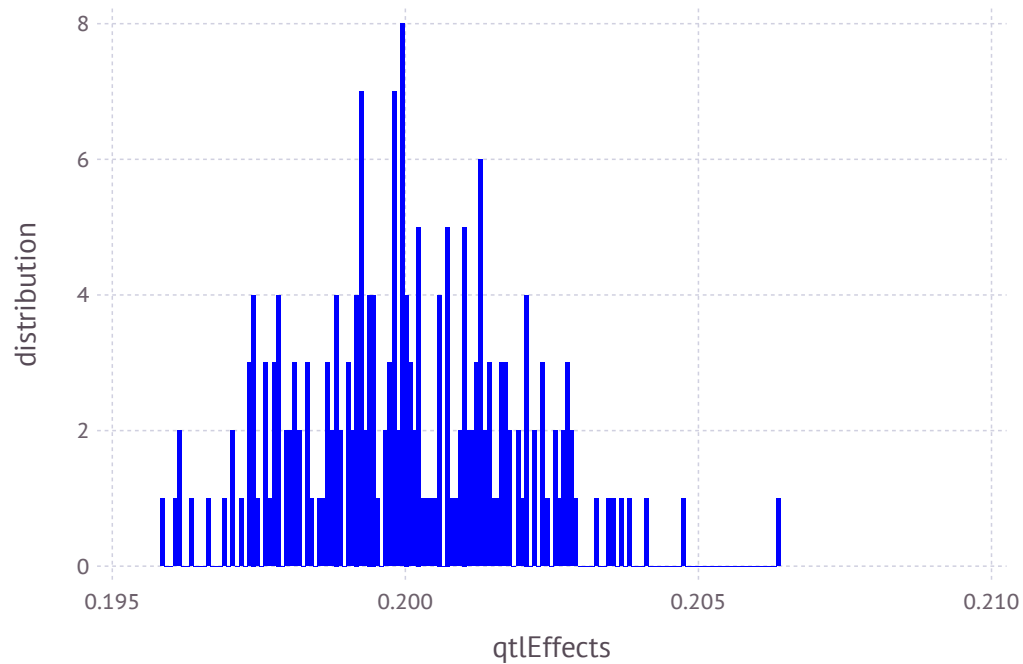
```
In [7]: qtlEffects
```

```
Out[7]: 200-element Array{Float64,1}:  
 0.199921  
 0.197063  
 0.1974  
 0.197968  
 0.203546  
 0.197768  
 0.200566  
 0.197337  
 0.202378  
 0.201039  
 0.201669  
 0.200362  
 0.198143  
 ⋮  
 0.2017  
 0.200469  
 0.199273  
 0.200746  
 0.198124  
 0.197959  
 0.200255  
 0.198916  
 0.199444  
 0.196897  
 0.200098  
 0.201622
```

```
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]: 0.2000227823350741

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

Out[11]: 3.359662540554264e-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
```

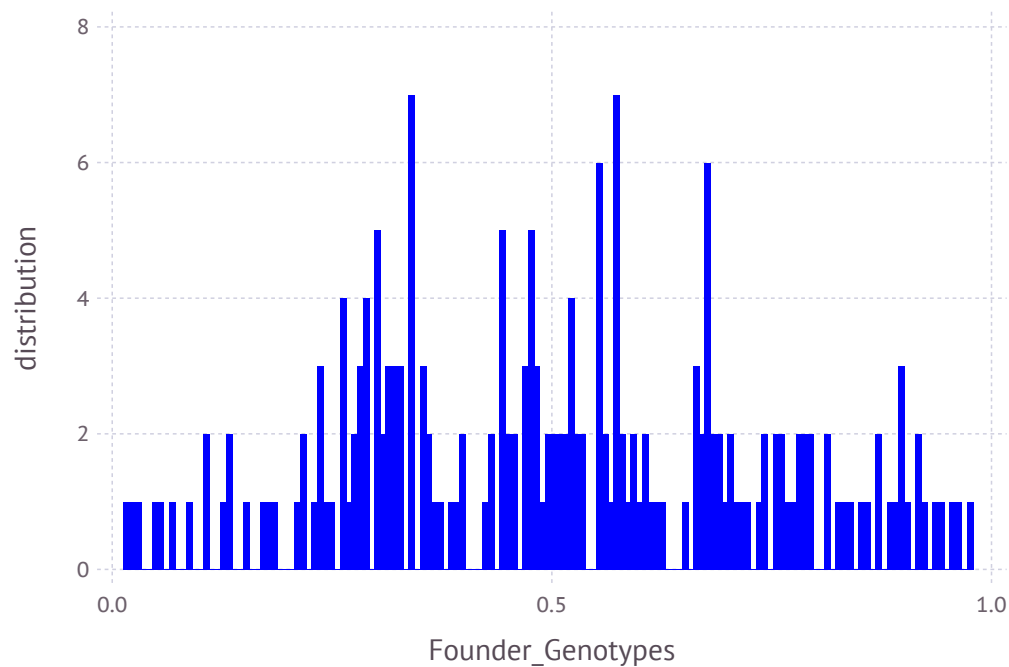
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

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

```
Out[20]: 1x200 Array{Float64,2}:
 0.05775  0.8315  0.299  0.943625  0.8125  ...  0.385125  0.893625  0.563875
```

```
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);
```



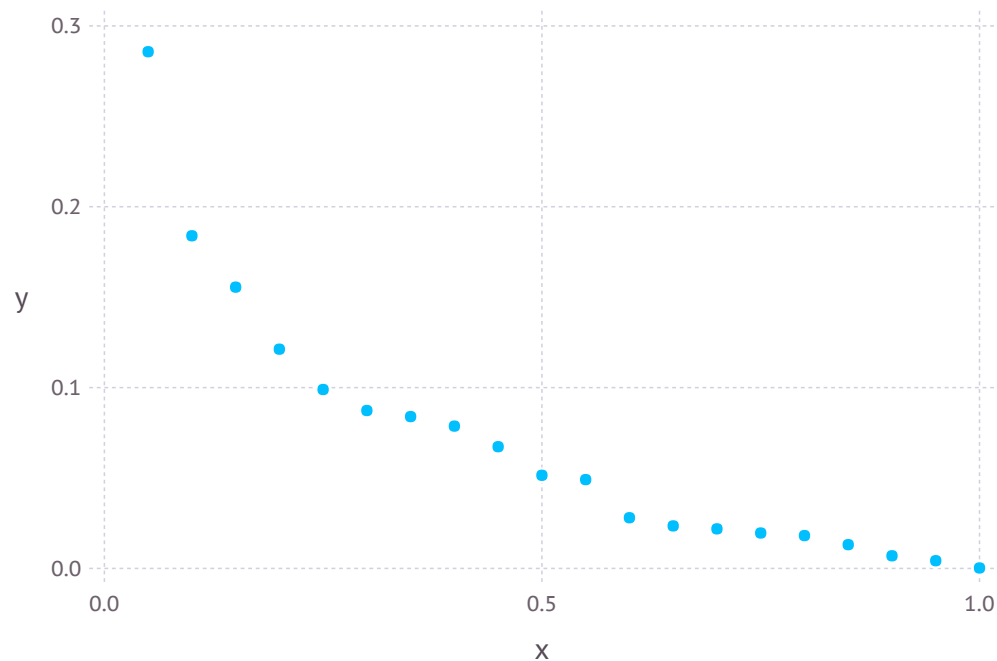
```
In [23]: for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

```
Out[24]: 1x20 Array{Float64,2}:
 0.000285404  0.00429435  0.00699437 ...  0.15557  0.183938  0.285753
```

```
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]: 10.162343283558108
```

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

```
Out[31]: 0.60244710580098
```

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

```
Out[32]: 1.4057099135356201
```

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

```
Out[33]: 1.4057099135356201
```

```
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]: 12.167398669964184
```

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

```
Out[36]: 12.18282794879621
```

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

```
Out[37]: 0.5006857683806543
```

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

```
Out[38]: 0.4911582300733375
```

Pedigree: All animals

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

```
Out[39]: 48000x3 Array{Int64,2}:  
  40722  36568  38466  
  40723  34049  40664  
  40724  34985  37386  
  40725  35989  38311  
  40726  34080  36894  
  40727  36538  37346  
  40728  33279  39328  
  40729  35213  37072  
  40730  34061  38920  
  40731  34607  36900  
  40732  34978  39190  
  40733  35661  39315  
  40734  36070  37685  
      ⋮  
  88710  75040  79970  
  88711  74874  79588  
  88712  76396  78093  
  88713  76198  80615  
  88714  75495  79959  
  88715  76142  78998  
  88716  74756  79621  
  88717  74883  77119  
  88718  75343  80693  
  88719  74673  80550  
  88720  76501  80558  
  88721  76242  77577
```

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

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

```
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}:
```

```
41407
44582
42623
43741
41318
42436
44102
44240
43357
42504
44257
44658
42340
⋮
75040
74874
76396
76198
75495
76142
74756
74883
75343
74673
76501
76242
```



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

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

```
41407
44582
42623
43741
41318
42436
44102
44240
43357
42504
44257
44658
42340
⋮
74883
75749
75500
75589
74653
73304
76389
75081
73328
74763
74747
75935
```

```
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}:
```

```
41407
44582
42623
43741
41318
42436
44102
44240
43357
42504
44257
44658
42340
⋮
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  11.116  10.11  
  40723   9.154   9.514  
  40724   9.699  10.106  
  40725   9.53   9.508  
  40726  11.383  10.71  
  40727   9.353   9.921  
  40728  12.539  10.902  
  40729  10.941  11.307  
  40730  11.633  11.499  
  40731  10.824   9.896  
  40732   9.019  10.704  
  40733   8.709  10.293  
  40734   9.725   9.127  
      ⋮  
  88710   9.785  12.492  
  88711  12.549  11.894  
  88712  13.292  12.503  
  88713  13.383  13.095  
  88714  13.245  13.273  
  88715  14.362  12.499  
  88716  10.944  11.896  
  88717  14.329  13.682  
  88718  13.439  13.475  
  88719  11.572  11.896  
  88720  12.588  12.096  
  88721  13.798  12.292
```

```
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}:
```

```
 2
 4
14
16
19
22
24
34
36
39
42
44
54
 ⋮
156
159
162
164
174
176
179
182
184
194
196
199
```



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

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

```
 1
 3
 5
 6
 7
 8
 9
10
11
12
13
15
17
 ⋮
186
187
188
189
190
191
192
193
195
197
198
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  0  ...  1  1  2  1  1  1  1  1  1  1  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  0  1  1  2  0  0  0  0  ...  2  2  1  1  1  2  0  1  1  0  2  1
 0  2  1  2  1  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  ...  2  1  0  2  1  2  1  0  0  1  1  2
 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
 1  2  0  2  2  0  1  1  1  1  1  1  1  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  0  2  2  2  0  2  2  0  2  2  0  1  ...  2  1  1  1  2  2  0  1  1  1  2  0
 0  1  2  1  0  0  1  1  1  1  1  1  1  ...  2  1  2  2  1  2  0  0  0  2  1  2
 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  2  1  2  1  0  0  0  2  0  0  1  0  ...  1  0  1  2  0  1  1  0  0  1  2  2
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  1  0  1  2  1  2  0  1  1  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  1  1  2  1  2  0  0  0  1  2  1
  ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 0  2  0  2  2  1  2  2  1  1  1  1  2  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  0  1  1  1  1  1  0  1  ...  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  0  2  2  0  2  2  0  2  2  0  2  ...  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  1  2  1  0  1  1  1  1  1  1  1  ...  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  ...  1  1  2  2  0  1  0  1  1  0  2  2
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  2  2  0  0  0  0  2  0  0  2  0  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  1  2  2  1  1  1  1  1  1  1  2  ...  2  2  1  1  2  2  0  1  1  0  2  1
 0  2  0  2  2  0  0  0  1  1  1  0  0  ...  2  2  1  1  1  1  0  1  1  0  1  1
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  2  1  1  1  2  2  0  1  1  1  2  0
 1  2  0  2  2  0  2  2  0  2  2  0  0  ...  2  0  1  1  2  2  0  1  1  1  2  1
```

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

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

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

```
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]: 0.4290456925114299
```

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

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

```
Out[100]: 0.6586153015683713
```

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

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

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

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

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

```
0.197063
0.197968
0.197603
0.196101
0.201891
0.20057
0.201437
0.199789
0.199039
0.199824
0.201254
0.197184
0.199318
⋮
0.202936
0.19776
0.202547
0.201319
0.201092
0.202049
0.197734
0.197811
0.19996
0.197959
0.198916
0.200098
```

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

```
Out[103]: 48000-element Array{Float64,1}:  
 10.1731  
  9.58644  
 10.1912  
  9.58036  
 10.7854  
  9.98148  
 10.974  
 11.3764  
 11.5857  
  9.98544  
 10.7804  
 10.3688  
  9.18554  
  ⋮  
 12.5771  
 11.9859  
 12.5859  
 13.1866  
 13.3764  
 12.5896  
 11.9809  
 13.7798  
 13.5858  
 12.0012  
 12.1907  
 12.3708
```

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

```
Out[104]: 10.233033658437947
```

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

```
Out[105]: 10.710578863981564
```

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

```
Out[106]: 11.09144865518318
```



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

```
Out[107]: 11.509129625243723
```

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

```
Out[108]: 11.89764397848844
```

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

```
Out[109]: 12.263034941939255
```

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

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

```
10.974
12.1863
10.987
11.5833
10.7846
10.5903
12.3865
11.5822
11.3896
11.5704
10.3758
12.1709
11.7863
⋮
12.5771
11.9859
12.5859
13.1866
13.3764
12.5896
11.9809
13.7798
13.5858
12.0012
12.1907
12.3708
```

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

```
Out[111]: 12.221983112073598
```

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

```
Out[112]: 1.9889494536356516
```

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

```
Out[113]: 11.187607595970455
```

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

```
Out[114]: 0.9545739375325084
```

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

```
Out[115]: 11.468343320879894
```

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

```
Out[116]: 1.2353096624419475
```

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

```
Out[117]: 11.879524109095126
```

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

```
Out[118]: 1.6464904506571791
```

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

```
Out[119]: 12.29374946757466
```

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

```
Out[120]: 2.0607158091367133
```

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

```
Out[121]: 12.638617872221655
```

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

```
Out[122]: 2.4055842137837082
```

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

```
Out[123]: 12.263034941939255
```

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

```
Out[124]: 2.030001283501308
```