

```
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)
```

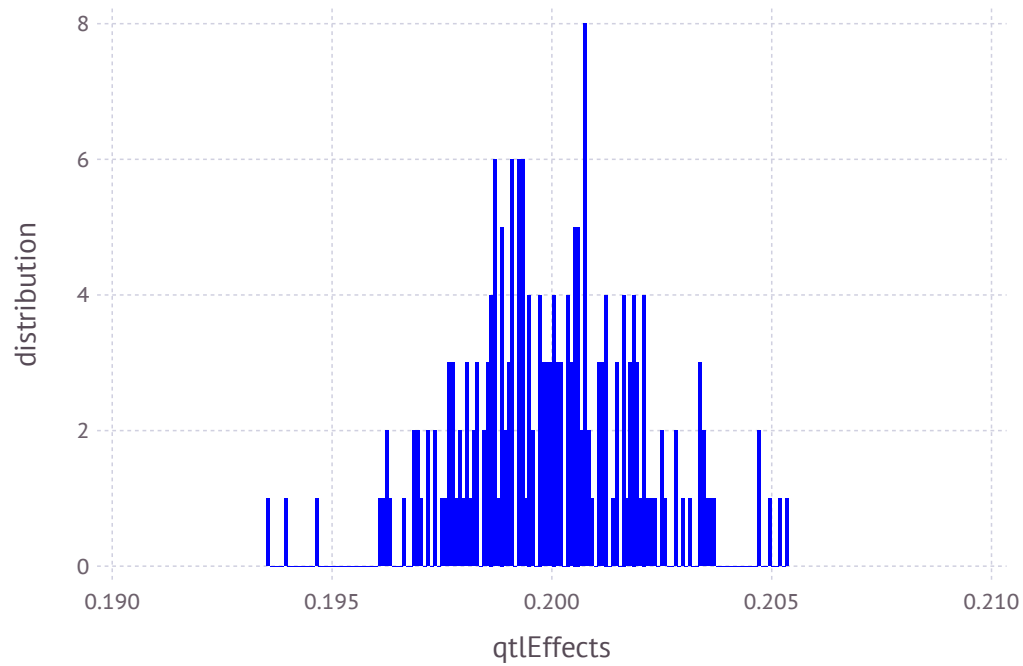
```
In [7]: qtlEffects
```

```
Out[7]: 200-element Array{Float64,1}:  
 0.19823  
 0.202595  
 0.200571  
 0.201068  
 0.193977  
 0.196861  
 0.199914  
 0.197634  
 0.197206  
 0.198048  
 0.199773  
 0.200503  
 0.200388  
  ⋮  
 0.199553  
 0.203531  
 0.201607  
 0.200024  
 0.196226  
 0.201796  
 0.200166  
 0.201233  
 0.198663  
 0.202831  
 0.19736  
 0.199956
```

```
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]: 0.19995619261625058
```

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

```
Out[11]: 4.066607000557279e-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
```

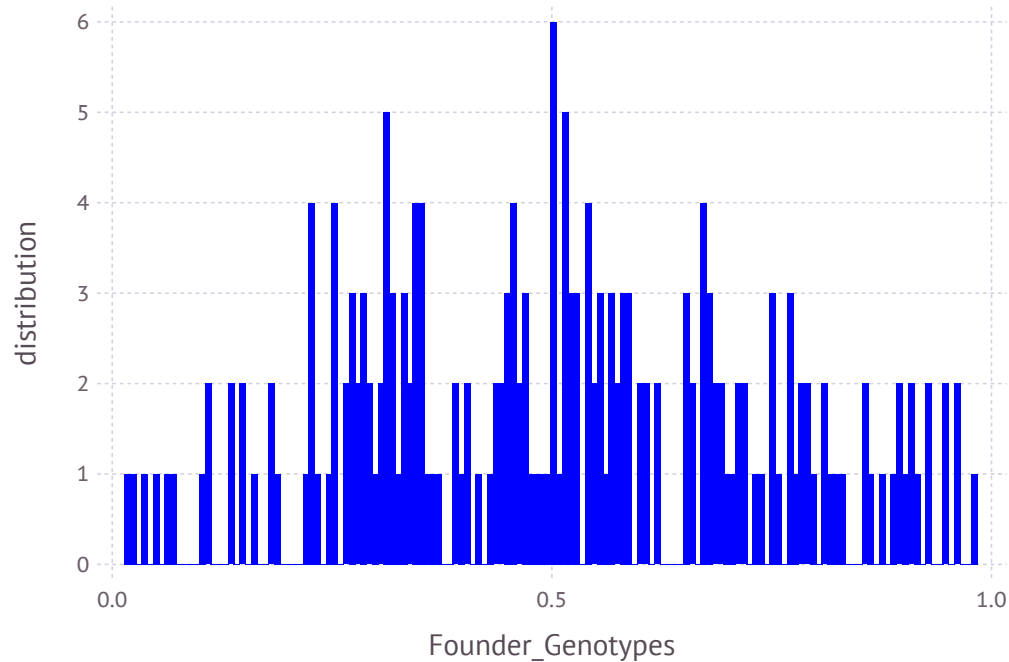
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

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

```
Out[20]: 1x200 Array{Float64,2}:
 0.063125  0.829  0.30775  0.945  0.8085 ...  0.35275  0.3885  0.893  0.57025
```

```
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);
```



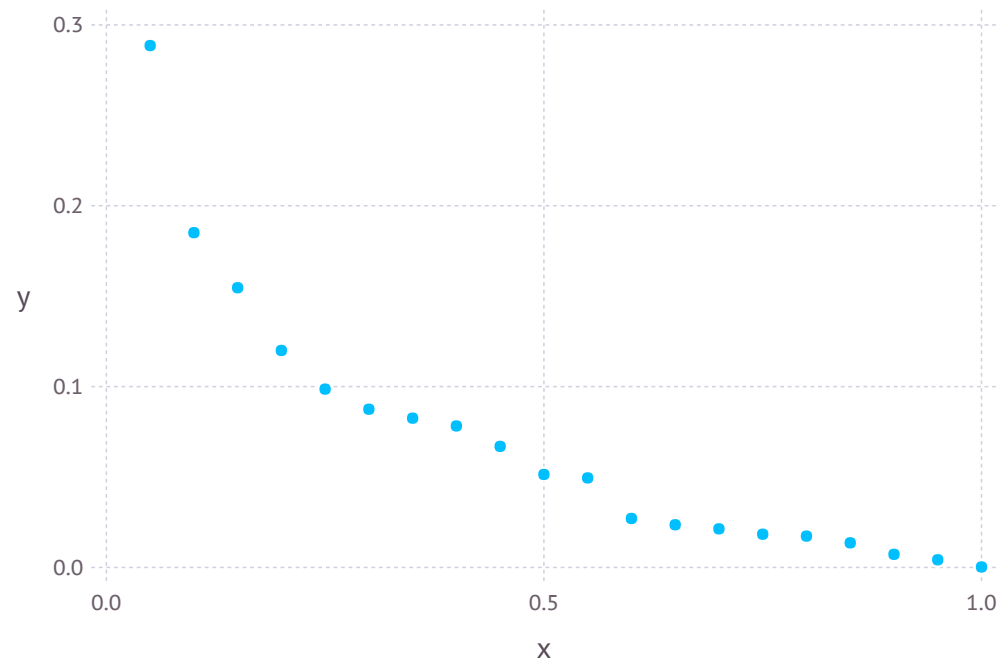
```
In [23]: for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

```
Out[24]: 1x20 Array{Float64,2}:
 0.000283706  0.00424247  0.00725716 ...  0.154674  0.185136  0.288596
```

```
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]: 11.138524962927796
```

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

```
Out[31]: 1.0279949931049404
```

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

```
Out[32]: 9.251954937944463
```

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

```
Out[33]: 9.251954937944463
```

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

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

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

```
Out[35]: 13.021934951839459
```

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

```
Out[36]: 13.0227225468594
```

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

```
Out[37]: 0.8150741556743577
```

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

```
Out[38]: 0.8300963854322594
```

Pedigree: All animals

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

```
Out[39]: 48000x3 Array{Int64,2}:  
 40722  33871  39825  
 40723  33371  39354  
 40724  36242  39145  
 40725  36460  37081  
 40726  36449  37258  
 40727  34676  37189  
 40728  35902  38527  
 40729  33156  40403  
 40730  32993  37325  
 40731  34196  37947  
 40732  34856  38621  
 40733  33529  38312  
 40734  34125  37549  
      ⋮  
 88710  75255  77377  
 88711  73737  80105  
 88712  73006  77500  
 88713  76535  79343  
 88714  76501  79879  
 88715  73008  77386  
 88716  74588  76761  
 88717  74590  78315  
 88718  76035  79235  
 88719  73215  77810  
 88720  73635  79710  
 88721  75637  79630
```

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

```
44207
42941
43862
44256
44200
44205
42667
41578
43092
43781
43925
44522
40823
⋮
75255
73737
73006
76535
76501
73008
74588
74590
76035
73215
73635
75637
```



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

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

```
44207
42941
43862
44256
44200
44205
42667
41578
43092
43781
43925
44522
40823
⋮
74525
74296
76035
73008
74528
74014
74559
74345
74642
72755
76707
73098
```

```
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}:
 44207
 42941
 43862
 44256
 44200
 44205
 42667
 41578
 43092
 43781
 43925
 44522
 40823
      ⋮
 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    9.665   10.21
 40723    9.574   10.804
 40724    9.742   10.999
 40725   10.796   11.004
 40726    9.884   11.205
 40727   11.758   10.199
 40728    8.586    9.209
 40729    9.954   11.398
 40730   13.023   12.414
 40731    8.658   11.806
 40732   11.256   11.409
 40733   12.575   11.805
 40734   12.633   12.41
      ⋮
 88710   15.381   14.007
 88711   13.011   13.201
 88712   17.757   13.603
 88713   13.019   14.805
 88714   14.55    14.399
 88715   15.152   12.002
 88716   15.892   13.599
 88717   11.432   13.007
 88718   14.76    13.799
 88719   13.344   12.803
 88720   15.478   11.808
 88721   18.406   12.604
```

```
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}:
```

```
 3  
 4  
 8  
12  
13  
23  
24  
28  
32  
33  
43  
44  
48  
:  
152  
153  
163  
164  
168  
172  
173  
183  
184  
188  
192  
193
```



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

```
Out[83]: 150-element Array{Int64,1}:
 1
 2
 5
 6
 7
 9
10
11
14
15
16
17
18
 ⋮
186
187
189
190
191
194
195
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  1  1  1  1  0  1  1  1  1  1  1  2  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  1  1  2  2  0  0  0  2  0  0  1  1  ...  1  1  2  1  1  1  1  1  1  1  2  1
 0  1  1  1  1  0  1  1  1  1  1  1  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  1  1  2  2  0  0  0  2  1  2
 1  2  0  2  2  1  2  2  0  2  2  0  1  ...  2  1  2  1  2  2  0  1  1  1  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  2  0  0  2  2  2  0  0  0  2  2  0
 0  2  0  2  2  1  1  1  1  1  1  1  2  ...  2  0  0  2  0  2  0  0  0  0  2  2
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  0  0  2  2  1  1  2  0  0  1  1  2
 0  1  1  1  1  0  2  2  1  1  1  1  2  ...  2  1  1  1  1  2  0  1  1  0  2  1
 0  2  1  2  1  0  0  0  2  0  0  1  0  ...  2  0  0  2  2  2  0  0  0  2  2  0
 0  2  0  2  2  1  1  1  1  1  1  1  2  ...  2  0  2  2  1  0  1  0  0  1  2  1
 0  1  1  2  2  1  2  2  0  2  2  0  1  ...  2  1  1  1  2  2  0  1  1  1  2  0
⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 0  1  1  2  2  1  0  0  2  0  0  2  2  ...  2  0  2  2  2  1  1  0  0  2  1  2
 1  1  1  1  1  1  2  2  0  2  2  0  2  ...  2  1  2  1  2  1  0  1  1  0  2  1
 0  1  1  2  2  1  2  2  1  1  1  1  2  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  1  1  2  2  1  1  1  1  1  1  1  2  ...  2  1  0  2  2  2  0  0  0  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  2  1  2  1  2  2  0  1  1  1  1  1
 0  2  1  2  1  1  1  1  2  0  0  2  1  ...  2  0  1  2  2  1  0  0  0  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  2  1  2  2  2  2  0  0  0  2  1  1
 0  2  0  2  2  0  1  1  2  0  0  1  1  ...  2  2  1  1  1  1  0  1  1  0  1  1
 0  0  2  2  2  1  2  2  0  2  2  0  1  ...  2  1  1  1  2  2  0  1  1  1  2  0
 0  2  0  2  2  1  1  1  1  1  1  1  2  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  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  1  1  1  1  1  1  1  2  ...  2  1  1  1  2  2  0  1  1  1  2  0
```

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

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

```
 1  1  1  1  2  1  1  1  0  2  1  1  1  ...  1  2  0  2  2  1  0  1  2  1  1  1
 1  2  0  1  1  2  1  1  2  0  1  1  1  ...  0  2  0  2  2  1  0  1  2  1  1  1
 1  1  1  1  2  1  1  0  1  1  0  0  1  ...  1  2  0  2  2  1  2  2  2  2  1  1
 0  2  2  1  2  1  1  0  2  0  1  0  1  ...  2  2  1  2  2  0  1  2  2  1  1  2
 0  2  2  0  1  2  1  1  1  1  1  1  2  ...  2  2  0  2  2  2  0  2  2  2  1  2
 0  2  0  1  1  1  0  0  1  1  0  0  1  ...  2  1  0  2  2  2  0  2  2  2  0  2
 0  2  0  2  2  0  0  0  0  2  1  1  2  ...  1  2  1  2  2  0  0  2  0  0  2  2
 0  2  1  1  2  1  0  1  1  1  1  1  1  ...  1  2  0  2  2  1  0  2  2  1  2  0
 0  2  0  2  2  2  2  0  2  0  0  0  0  ...  2  2  0  2  2  2  0  1  2  1  2  1
 1  1  2  1  2  1  0  1  1  1  1  0  1  ...  2  2  0  2  2  2  1  2  2  1  1  1
 1  2  0  1  0  0  0  0  0  2  2  0  1  ...  1  1  0  2  2  2  0  2  0  0  2  2
 0  2  1  1  2  2  2  2  1  1  1  1  2  ...  1  2  0  2  2  2  0  2  2  1  2  1
 1  2  2  0  1  2  2  0  1  1  2  2  2  ...  2  2  0  2  2  1  0  2  1  1  1  2
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 1  2  0  2  2  2  1  1  1  1  1  1  1  ...  1  2  0  2  2  2  0  2  2  2  2  2
 1  1  2  0  2  1  2  1  1  1  2  1  1  ...  1  1  0  2  2  2  1  2  2  1  1  2
 1  2  2  1  2  1  1  0  1  1  2  1  1  ...  1  2  0  2  2  1  1  2  1  2  0  2
 1  2  1  1  2  1  0  1  1  1  2  2  2  ...  2  2  1  2  2  2  0  2  1  0  2  2
 0  2  0  2  2  2  2  2  1  1  2  1  1  ...  2  2  1  2  2  0  1  2  2  2  1  2
 1  2  1  2  1  1  1  0  2  0  0  0  0  ...  1  2  0  2  2  2  0  2  1  0  2  2
 0  2  0  2  2  1  0  1  2  0  1  1  1  ...  2  2  0  2  2  2  0  2  1  2  2  2
 0  2  1  1  1  1  1  1  1  1  2  0  0  ...  1  2  0  2  2  2  2  2  2  1  1  1
 2  2  2  0  1  1  1  1  1  1  1  2  ...  2  2  0  2  1  2  0  2  1  1  1  2
 0  2  1  1  2  2  1  1  2  0  1  1  2  ...  0  2  1  2  2  1  0  2  2  2  0  2
 0  2  0  2  2  2  1  0  1  1  1  1  2  ...  0  2  0  2  2  2  0  2  2  1  1  2
 0  2  1  1  2  1  0  0  1  2  1  1  1  ...  0  2  1  2  2  2  0  2  1  1  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]: 0.13898252974397315
```

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

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

```
Out[100]: 0.3642493315134955
```

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

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

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

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

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

```
0.200571  
0.201068  
0.197634  
0.200503  
0.200388  
0.197334  
0.20043  
0.199459  
0.20111  
0.197497  
0.202112  
0.200603  
0.202107  
⋮  
0.202525  
0.199124  
0.202092  
0.20213  
0.198471  
0.19927  
0.198619  
0.199476  
0.19904  
0.198877  
0.200024  
0.196226
```

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

```
Out[103]: 48000-element Array{Float64,1}:  
 10.1827  
 10.7852  
 10.9943  
 10.985  
 11.1809  
 10.1832  
  9.18567  
 11.3826  
 12.3987  
 11.7937  
 11.4003  
 11.7834  
 12.3894  
  ⋮  
 13.9818  
 13.1833  
 13.5814  
 14.7875  
 14.3868  
 11.9846  
 13.5932  
 12.9797  
 13.7799  
 12.7759  
 11.7829  
 12.5849
```

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

```
Out[104]: 11.118173273076431
```

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

```
Out[105]: 11.892114850999038
```

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

```
Out[106]: 12.1668299086559
```



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

```
Out[107]: 12.472281774609666
```

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

```
Out[108]: 12.722214279786632
```

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

```
Out[109]: 13.004836065219001
```

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

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

```
11.3791
11.986
12.5793
12.5845
13.5784
11.7825
11.7934
12.9839
13.3904
12.9737
10.9804
12.1868
13.5853
⋮
13.9818
13.1833
13.5814
14.7875
14.3868
11.9846
13.5932
12.9797
13.7799
12.7759
11.7829
12.5849
```

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

```
Out[111]: 12.983741315607702
```

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

```
Out[112]: 1.8655680425312706
```

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

```
Out[113]: 12.639902283099453
```

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

```
Out[114]: 1.5217290100230212
```

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

```
Out[115]: 12.424522067488493
```

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

```
Out[116]: 1.3063487944120613
```

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

```
Out[117]: 12.746843385094273
```

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

```
Out[118]: 1.6286701120178417
```

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

```
Out[119]: 12.950443222792547
```

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

```
Out[120]: 1.8322699497161157
```

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

```
Out[121]: 13.313205635111778
```

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

```
Out[122]: 2.195032362035347
```

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

```
Out[123]: 13.004836065219001
```

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

```
Out[124]: 1.8866627921425696
```