

```
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)
```

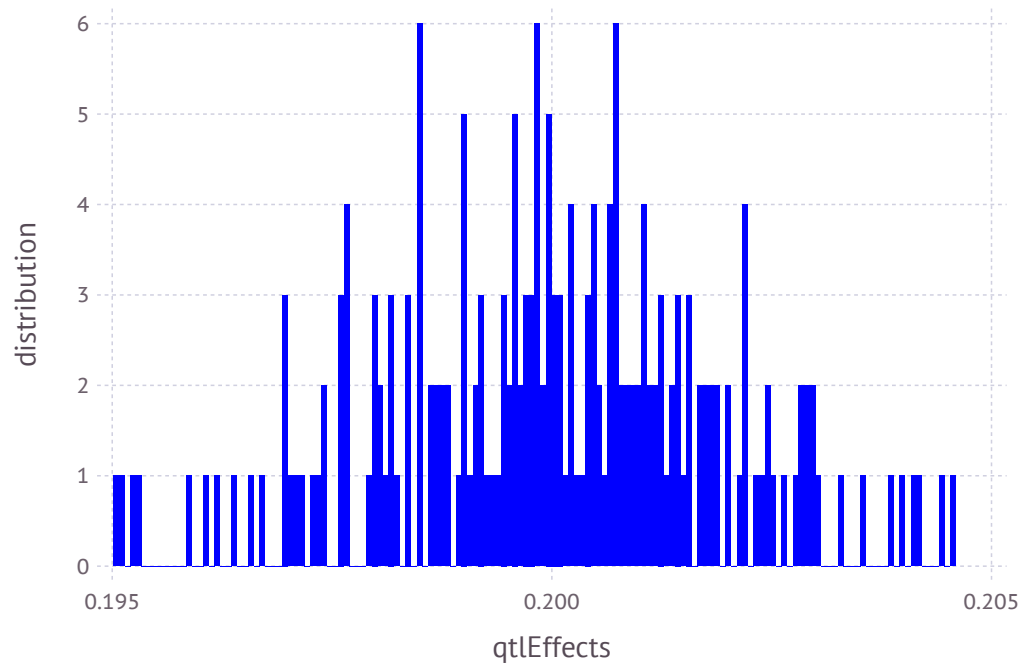
```
In [7]: qtlEffects
```

```
Out[7]: 200-element Array{Float64,1}:  
 0.199321  
 0.201272  
 0.197673  
 0.200656  
 0.200844  
 0.20077  
 0.204452  
 0.201264  
 0.20049  
 0.198129  
 0.200415  
 0.201159  
 0.201203  
  ⋮  
 0.196402  
 0.202627  
 0.202447  
 0.195014  
 0.202218  
 0.201267  
 0.200753  
 0.201426  
 0.199189  
 0.199974  
 0.197889  
 0.199805
```

```
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]: 0.19996905037115947
```

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

```
Out[11]: 3.6470212491308034e-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
```

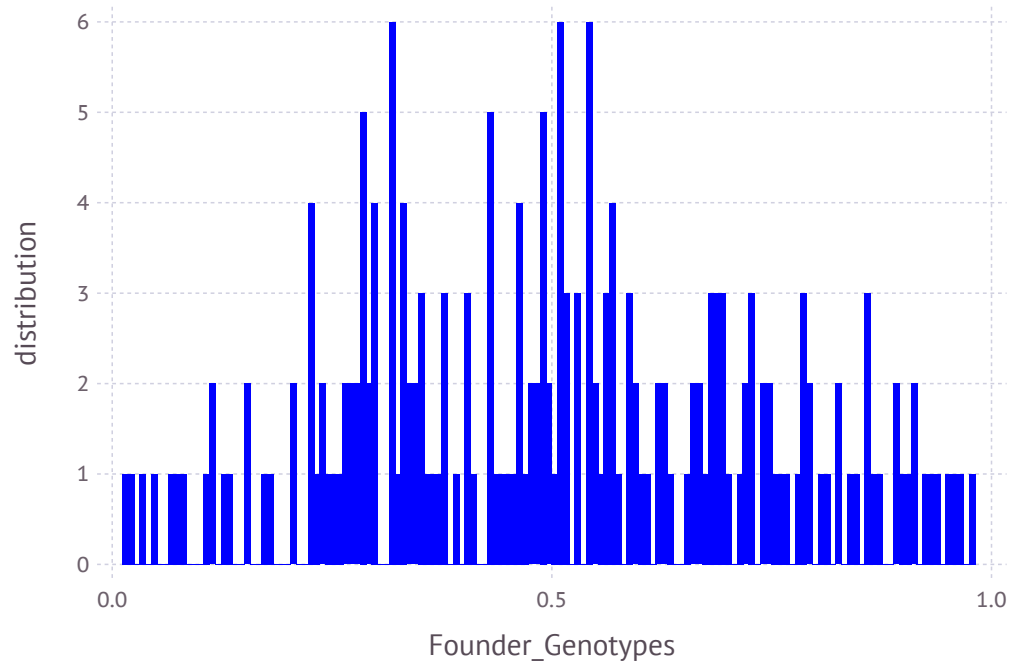
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

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

```
Out[20]: 1x200 Array{Float64,2}:
 0.068125  0.844125  0.27  0.95425  0.839375  ...  0.371875  0.890625  0.54625
```

```
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);
```



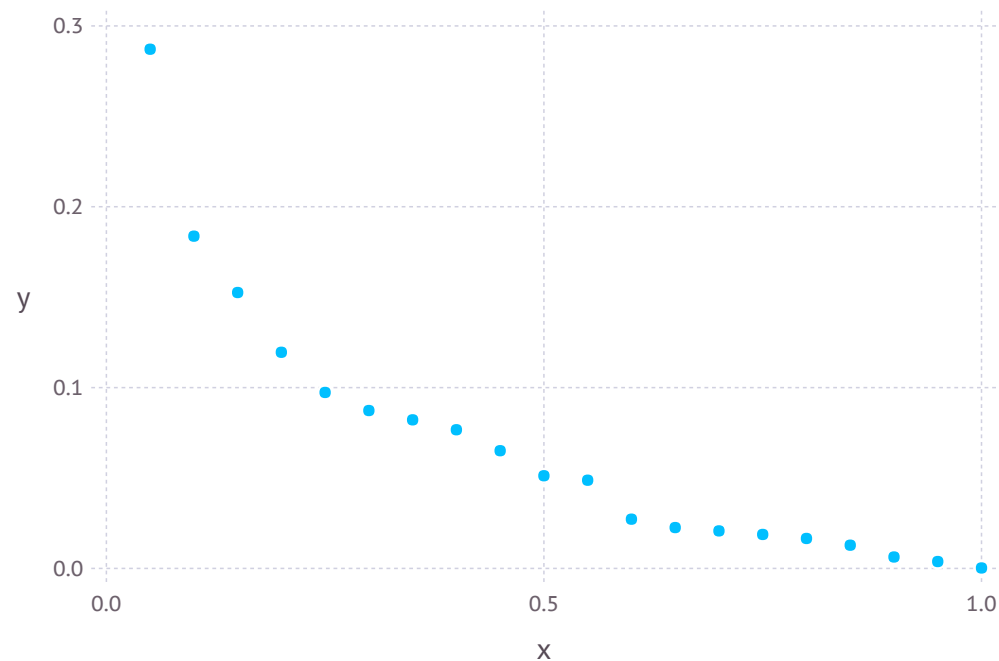
```
In [23]: for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

```
Out[24]: 1x20 Array{Float64,2}:
 0.000271605  0.00382646  0.00634456  ...  0.152572  0.183752  0.28712
```

```
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]: 11.05771502398269
```

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

```
Out[31]: 0.8720257198155774
```

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

```
Out[32]: 7.848231478340197
```

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

```
Out[33]: 7.848231478340197
```

```
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]: 12.729399880562992
```

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

```
Out[36]: 12.720693751988541
```

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

```
Out[37]: 0.7348910234478364
```

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

```
Out[38]: 0.7408072163763152
```

Pedigree: All animals

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

```
Out[39]: 48000x3 Array{Int64,2}:  
  40722  33255  38376  
  40723  34117  38696  
  40724  35213  37680  
  40725  34072  38432  
  40726  35252  38931  
  40727  34930  36761  
  40728  35837  39045  
  40729  32749  37119  
  40730  32900  36780  
  40731  34253  39725  
  40732  33766  40641  
  40733  34083  39037  
  40734  34590  39063  
      ⋮  
  88710  75916  80717  
  88711  75995  78070  
  88712  75938  80193  
  88713  76533  77136  
  88714  76105  80579  
  88715  73897  78828  
  88716  75531  77851  
  88717  72801  78788  
  88718  76435  77409  
  88719  73224  77623  
  88720  74034  80705  
  88721  75210  78110
```

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

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

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

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

Create matrix of "genotype" covariates for all animals

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

```
Out[43]: 48000
```

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

```
Out[44]: 200
```

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

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

```
40722 0 2 0 2 2 0 1 1 1 1 ... 2 2 2 0 2 2 0 2 2 0 2 0
40723 0 1 1 2 2 0 1 1 1 1 ... 2 0 1 2 2 1 0 0 0 1 1 2
40724 1 2 0 2 2 0 1 1 1 1 ... 2 2 0 2 1 1 1 0 0 1 1 2
40725 0 2 1 2 1 0 0 0 2 0 ... 0 0 2 2 0 0 2 0 0 2 2 2
40726 0 2 1 2 1 1 1 1 1 1 ... 1 1 2 2 1 1 1 0 0 1 2 1
40727 0 2 0 2 2 0 0 0 2 0 ... 2 1 1 1 2 2 0 1 1 1 2 0
40728 0 2 0 2 2 1 1 1 1 1 ... 0 0 2 2 0 0 2 0 0 2 2 2
40729 0 1 1 2 2 0 1 1 1 1 ... 2 1 1 0 1 2 0 1 1 0 2 1
40730 0 0 2 2 2 0 2 2 0 2 ... 2 2 1 1 1 1 0 1 1 0 1 1
40731 0 1 2 1 0 0 1 1 1 1 ... 2 1 0 2 1 1 0 0 0 1 1 1
40732 0 1 1 2 2 1 1 1 1 1 ... 2 1 2 0 2 2 0 2 2 0 2 0
40733 1 2 0 2 2 1 1 1 1 1 ... 2 1 1 1 2 1 1 1 1 0 2 1
40734 0 2 1 2 1 1 1 1 1 1 ... 2 1 2 1 1 1 0 1 1 1 1 1
      ⋮                ⋮                ⋮                ⋮                ⋮
88710 1 1 1 2 2 0 2 2 0 2 ... 2 2 1 1 1 1 1 1 1 1 2 1
88711 0 1 1 1 1 0 1 1 1 1 ... 2 2 2 0 2 2 0 2 2 0 2 0
88712 0 2 0 2 2 0 1 1 1 1 ... 2 1 1 1 1 1 0 0 0 0 1 2
88713 0 2 0 2 2 0 0 0 2 0 ... 2 2 2 0 2 2 0 2 2 0 2 0
88714 0 2 0 2 2 0 1 1 1 1 ... 2 2 2 0 2 2 0 2 2 1 2 1
88715 1 2 0 2 2 0 1 1 1 1 ... 2 1 2 1 2 1 1 1 1 0 1 1
88716 0 2 0 2 2 0 0 0 2 0 ... 1 0 1 2 1 0 2 0 0 1 2 2
88717 0 2 0 2 2 0 1 1 1 1 ... 1 1 2 2 1 1 1 0 0 2 2 2
88718 0 0 2 2 2 0 2 2 0 2 ... 2 1 1 2 1 2 0 0 0 1 2 1
88719 0 1 1 2 2 2 2 2 0 2 ... 1 0 2 2 1 0 2 0 0 2 2 2
88720 0 2 0 2 2 0 0 0 2 0 ... 2 2 1 2 1 1 0 0 0 1 1 1
88721 0 2 0 2 2 1 1 1 1 1 ... 2 0 2 1 2 2 0 1 1 1 1 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  0  1  1  1  1  1  0  0  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  1  1  2  2  0  1  1  1  1  1  0  0  ...  2  0  1  2  2  1  0  0  0  1  1  2
 1  2  0  2  2  0  1  1  1  1  1  0  0  ...  2  2  0  2  1  1  1  0  0  1  1  2
 0  2  1  2  1  0  0  0  2  0  0  1  1  ...  0  0  2  2  0  0  2  0  0  2  2  2
 0  2  1  2  1  1  1  1  1  1  1  1  1  ...  1  1  2  2  1  1  1  0  0  1  2  1
 0  2  0  2  2  0  0  0  2  0  0  1  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  ...  0  0  2  2  0  0  2  0  0  2  2  2
 0  1  1  2  2  0  1  1  1  1  1  0  0  ...  2  1  1  0  1  2  0  1  1  0  2  1
 0  0  2  2  2  0  2  2  0  2  2  0  0  ...  2  2  1  1  1  1  0  1  1  0  1  1
 0  1  2  1  0  0  1  1  1  1  1  1  1  ...  2  1  0  2  1  1  0  0  0  1  1  1
 0  1  1  2  2  1  1  1  1  1  1  1  2  ...  2  1  2  0  2  2  0  2  2  0  2  0
 1  2  0  2  2  1  1  1  1  1  1  0  1  ...  2  1  1  1  2  1  1  1  1  0  2  1
 0  2  1  2  1  1  1  1  1  1  1  1  1  ...  2  1  2  1  1  1  0  1  1  1  1  1
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 1  1  1  2  2  0  2  2  0  2  2  0  0  ...  2  2  1  1  1  1  1  1  1  1  2  1
 0  1  1  1  1  0  1  1  1  1  1  0  1  ...  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  ...  2  1  1  1  1  1  0  0  0  0  1  2
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  2  2  0  2  2  0  2  2  0  2  0
 0  2  0  2  2  0  1  1  1  1  1  1  1  ...  2  2  2  0  2  2  0  2  2  1  2  1
 1  2  0  2  2  0  1  1  1  1  1  1  1  ...  2  1  2  1  2  1  1  1  1  0  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  1  0  1  2  1  0  2  0  0  1  2  2
 0  2  0  2  2  0  1  1  1  1  1  0  1  ...  1  1  2  2  1  1  1  0  0  2  2  2
 0  0  2  2  2  0  2  2  0  2  2  0  1  ...  2  1  1  2  1  2  0  0  0  1  2  1
 0  1  1  2  2  2  2  2  0  2  2  0  2  ...  1  0  2  2  1  0  2  0  0  2  2  2
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  2  1  2  1  1  0  0  0  1  1  1
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  2  0  2  1  2  2  0  1  1  1  1  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}:
```

```
41207
42076
44397
42631
43099
42854
41235
44049
42385
42204
41207
42022
41929
      :
75916
75995
75938
76533
76105
73897
75531
72801
76435
73224
74034
75210
```



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

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

```
41207
42076
44397
42631
43099
42854
41235
44049
42385
42204
42022
41929
42738
⋮
75647
76208
76698
74726
76275
75670
72817
74405
75781
75659
72892
76533
```

```
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}:
 41207
 42076
 44397
 42631
 43099
 42854
 41235
 44049
 42385
 42204
 42022
 41929
 42738
      ⋮
 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    9.895   10.864
          40723   10.7    10.643
          40724   13.592   12.263
          40725    9.99    10.234
          40726   11.292   11.057
          40727   13.534   11.271
          40728    8.933    9.444
          40729    9.565   10.453
          40730   10.512   11.263
          40731   10.976   11.054
          40732   10.46    10.251
          40733   12.74    11.262
          40734    9.747   11.461
              ⋮
          88710   18.83    14.289
          88711   14.312   14.276
          88712   11.063   11.871
          88713   11.658   12.869
          88714   14.888   14.077
          88715   14.798   13.684
          88716   14.975   12.678
          88717   11.305   13.056
          88718   16.726   13.481
          88719    7.1     11.865
          88720   11.594   12.661
          88721    9.795   12.258
```

```
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}:
```

```
 1  
 4  
 7  
11  
12  
21  
24  
27  
31  
32  
41  
44  
47  
:  
151  
152  
161  
164  
167  
171  
172  
181  
184  
187  
191  
192
```



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

```
Out[83]: 150-element Array{Int64,1}:
 2
 3
 5
 6
 8
 9
10
13
14
15
16
17
18
 ⋮
186
188
189
190
193
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}:
```

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

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

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

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

```
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]: 0.14250660480414992
```

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

WARNING: imported binding for cor overwritten in module Main

```
Out[100]: 0.3769909662673336
```

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

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

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

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

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

```
0.199321
0.200656
0.204452
0.200415
0.201159
0.202898
0.200851
0.19965
0.200055
0.20072
0.200701
0.198827
0.199747
⋮
0.199508
0.199181
0.202887
0.20108
0.198187
0.199739
0.199018
0.198682
0.196943
0.199552
0.202447
0.195014
```

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

```
Out[103]: 48000-element Array{Float64,1}:  
  10.7864  
  10.5824  
  12.1764  
  10.1784  
  10.9795  
  11.1803  
   9.38858  
  10.393  
  11.196  
  10.9935  
  10.203  
  11.1758  
  11.3853  
   ⋮  
  14.1778  
  14.1795  
  11.7835  
  12.7831  
  13.976  
  13.5815  
  12.5712  
  12.9739  
  13.3728  
  11.7794  
  12.5631  
  12.1922
```

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

```
Out[104]: 10.986940765122164
```

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

```
Out[105]: 11.617256540400032
```

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

```
Out[106]: 11.865577784251036
```



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

```
Out[107]: 12.1366752374092
```

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

```
Out[108]: 12.406582850600662
```

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

```
Out[109]: 12.63960035074135
```

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

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

```
12.1764
11.1803
11.985
11.573
12.7923
11.9867
11.5883
12.7815
12.1776
12.3741
12.5715
12.5619
10.9858
⋮
14.1778
14.1795
11.7835
12.7831
13.976
13.5815
12.5712
12.9739
13.3728
11.7794
12.5631
12.1922
```

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

```
Out[111]: 12.619619879469928
```

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

```
Out[112]: 1.6326791143477646
```

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

```
Out[113]: 12.233585250190034
```

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

```
Out[114]: 1.2466444850678702
```

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

```
Out[115]: 12.11036359060031
```

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

```
Out[116]: 1.123422825478146
```

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

```
Out[117]: 12.38585079323222
```

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

```
Out[118]: 1.3989100281100573
```

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

```
Out[119]: 12.695657657539915
```

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

```
Out[120]: 1.7087168924177512
```

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

```
Out[121]: 12.873423254930337
```

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

```
Out[122]: 1.8864824898081736
```

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

```
Out[123]: 12.63960035074135
```

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

```
Out[124]: 1.6526595856191868
```