

```
In [1]: # Founders: real haplotype data (ch1to10.200SNP)
# "Sample animals for sire and dam candidates" & "Dams and Sires": dam size = N of dams
# One Dam -> 1 male & 1 female (1 dam -> 2 offsprings)
# selection: increase
# 5 generation selection: increase
# heritability = 0.5
# Phenotypes : all animals in G0 to G4
# Genotypes : all progeny in G5 and all sires in each generation
# Change muAlpha = 0.2
# 10 chromosomes; 200 loci per chromosome = > 2000 Loci (500 QTL & 1500 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  = 0.1
        numLoci    = 200
        nQTL       = 50
        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)
```

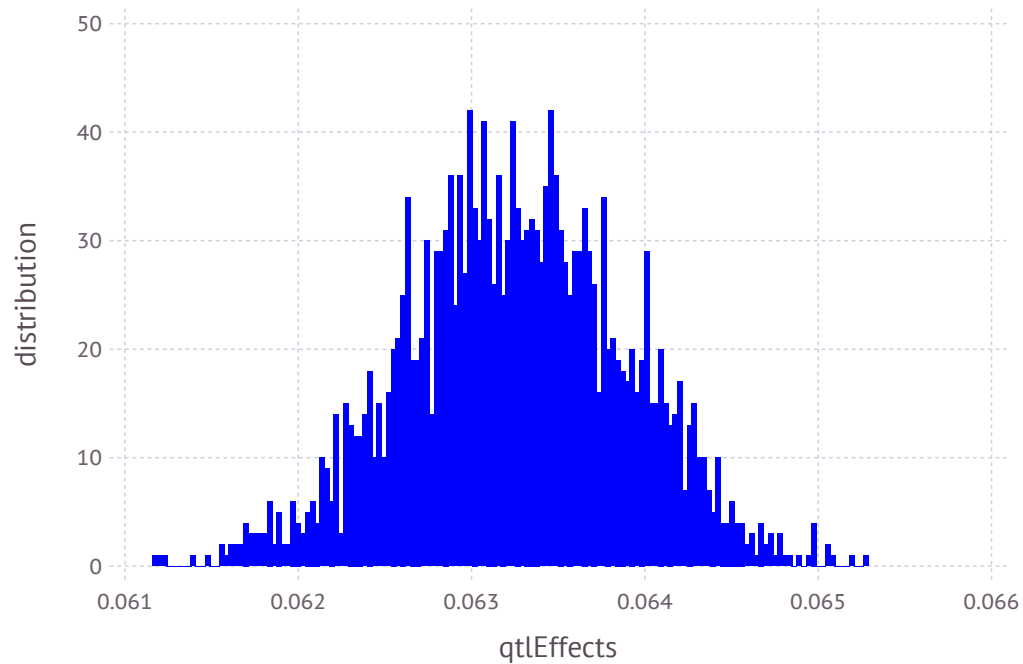
```
In [7]: qtlEffects
```

```
Out[7]: 2000-element Array{Float64,1}:  
  0.063266  
  0.0633232  
  0.0633805  
  0.0634677  
  0.0635943  
  0.0633221  
  0.0630592  
  0.0639862  
  0.0632293  
  0.0634891  
  0.0627156  
  0.0621245  
  0.0633108  
  ⋮  
  0.063372  
  0.063202  
  0.0617029  
  0.06328  
  0.0627552  
  0.0631919  
  0.0641241  
  0.0617969  
  0.0634812  
  0.0617003  
  0.0634097  
  0.0636767
```

```
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]: 0.06325471382134437
```

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

```
Out[11]: 4.1039295649160796e-7
```

```
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/ch1to10.2000SNP.group1")
         dams  = XSim.sampleFounders("/home/nicole/Jupyter/HERdata/data/Clean/newdata/ch1to10.2000SNP.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
```

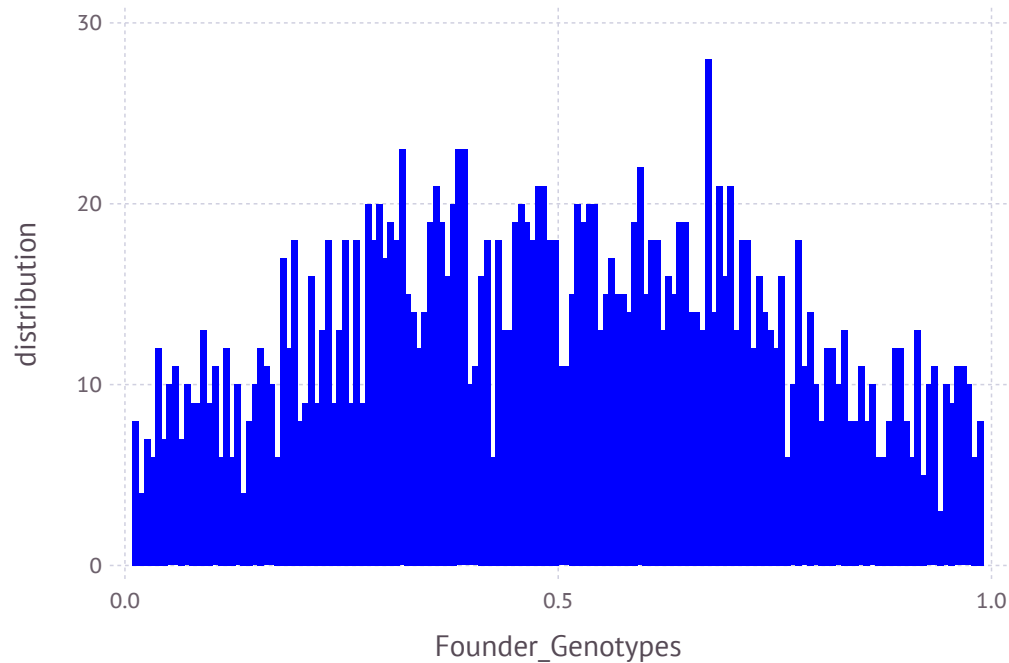
```
In [19]: gSPSire = XSim.getOurGenotypes(popSP[1])
gSPDam = XSim.getOurGenotypes(popSP[2])
gSP = [gSPSire;gSPSire];
```

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

```
Out[20]: 1x2000 Array{Float64,2}:
 0.0635  0.84625  0.28025  0.953125 ... 0.283625  0.344  0.451375  0.276875
```

```
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);
```



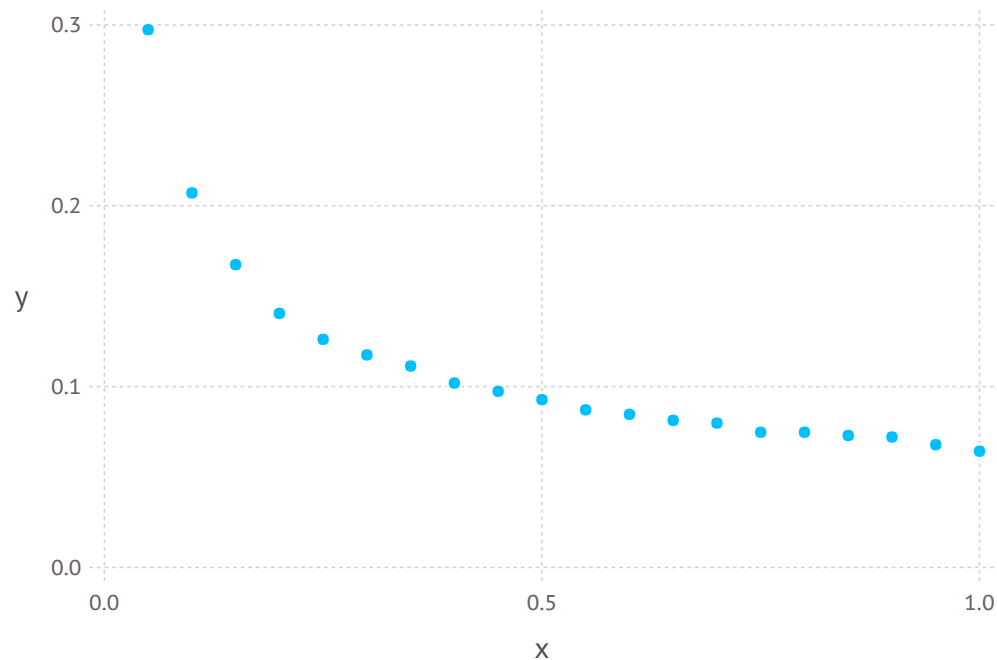
```
In [23]: for i=1:(nRows-20)
LDMat[i,:] = corMat[i,(i+1):(i+20)].^2
end
```

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

```
Out[24]: 1x20 Array{Float64,2}:
 0.0642942  0.0678797  0.0721447  0.0729801  ...  0.167444  0.207111  0.297347
```

```
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]: 30.32417112748493
```

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

```
Out[31]: 0.6824714340882967
```

```
In [32]: XSim.common.varRes = varGen    #heritability = 0.5
```

```
Out[32]: 0.6824714340882967
```

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

```
Out[33]: 0.6824714340882967
```

```
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]: 33.1519477079159
```

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

```
Out[36]: 33.1325478256955
```

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

```
Out[37]: 0.5481984896708653
```

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

```
Out[38]: 0.5832898316267019
```

Pedigree: All animals

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

```
Out[39]: 48000x3 Array{Int64,2}:  
  40722  36136  39630  
  40723  34970  39354  
  40724  35903  40599  
  40725  35004  37386  
  40726  34195  39027  
  40727  36609  37472  
  40728  32750  38903  
  40729  35888  37121  
  40730  36105  39681  
  40731  34009  38287  
  40732  33226  39173  
  40733  33976  40532  
  40734  35198  37475  
      ⋮  
  88710  76408  78413  
  88711  76197  79756  
  88712  76695  78547  
  88713  73693  80479  
  88714  76609  80675  
  88715  76599  80325  
  88716  75318  79830  
  88717  75897  79605  
  88718  75181  79896  
  88719  76482  79810  
  88720  75303  77980  
  88721  73369  79097
```

```
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]: 2000
```

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

```
Out[45]: 48000x2001 Array{Int64,2}:
```

```
40722 0 1 1 2 2 1 1 1 1 ... 1 1 2 1 0 1 2 2 1 0 1 0
40723 0 2 1 2 1 0 0 0 2 0 ... 0 2 0 0 0 1 1 1 1 1 1 1
40724 0 2 0 2 2 0 0 0 2 0 ... 2 1 2 2 0 1 1 1 1 1 0 0
40725 0 1 1 2 2 1 1 1 1 1 ... 1 1 1 1 0 0 1 1 0 1 1 0
40726 0 2 0 2 2 1 1 1 1 1 ... 1 1 1 1 0 0 1 1 0 1 1 0
40727 0 1 2 2 1 1 1 1 1 1 ... 2 1 2 2 0 1 2 2 1 0 1 1
40728 0 1 1 1 1 0 1 1 1 1 ... 1 1 0 0 1 2 2 2 1 0 2 2
40729 0 1 1 2 2 1 2 2 1 1 ... 1 2 1 1 0 1 2 2 1 0 1 0
40730 1 2 1 2 1 0 0 0 2 0 ... 1 2 1 1 0 1 2 2 1 0 1 0
40731 0 2 2 2 0 0 0 0 2 0 ... 2 0 2 1 0 0 1 1 0 1 1 0
40732 0 2 0 2 2 0 0 0 2 0 ... 1 2 1 1 0 1 2 2 1 0 1 0
40733 0 2 0 2 2 0 0 0 2 0 ... 2 1 1 1 2 2 2 2 0 0 2 2
40734 0 2 1 2 1 0 0 0 2 0 ... 1 2 1 1 0 1 2 2 1 0 1 1
      ⋮      ⋮      ⋮      ⋮      ⋮      ⋮      ⋮      ⋮      ⋮
88710 0 2 0 2 2 0 1 1 1 1 ... 0 2 0 0 0 0 2 2 1 0 1 0
88711 0 2 0 2 2 0 0 0 2 0 ... 2 1 2 2 0 1 1 1 1 1 0 0
88712 0 1 2 2 2 1 2 2 1 1 ... 1 2 1 1 0 1 2 2 1 0 1 0
88713 0 2 0 2 2 1 1 1 1 1 ... 1 2 1 1 0 1 1 1 1 1 0 0
88714 0 2 0 2 2 1 2 2 1 1 ... 2 1 2 1 0 1 2 2 1 0 0 0
88715 0 2 0 2 2 1 1 1 1 1 ... 1 2 1 1 1 2 2 2 1 0 1 1
88716 0 1 1 2 2 1 2 2 0 2 ... 0 2 0 0 0 0 2 2 0 0 2 1
88717 0 2 0 2 2 0 0 0 2 0 ... 1 2 1 1 0 1 2 2 1 0 1 1
88718 0 2 0 2 2 0 0 0 2 0 ... 2 1 2 1 1 2 2 2 1 0 1 1
88719 0 1 1 2 2 1 2 2 0 2 ... 2 2 2 2 0 2 2 2 2 0 0 0
88720 0 2 0 2 2 2 2 2 0 2 ... 2 0 2 2 0 1 2 2 1 0 1 0
88721 0 2 1 2 1 0 0 0 2 0 ... 1 1 1 1 1 1 1 1 0 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]: 48000x2000 Array{Int64,2}:
```

```
 0  1  1  2  2  1  1  1  1  1  1  0  1  ...  1  1  2  1  0  1  2  2  1  0  1  0
 0  2  1  2  1  0  0  0  2  0  0  1  1  ...  0  2  0  0  0  1  1  1  1  1  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  2  2  0  1  1  1  1  1  0  0
 0  1  1  2  2  1  1  1  1  1  1  1  2  ...  1  1  1  1  0  0  1  1  0  1  1  0
 0  2  0  2  2  1  1  1  1  1  1  1  2  ...  1  1  1  1  0  0  1  1  0  1  1  0
 0  1  2  2  1  1  1  1  1  1  1  1  1  ...  2  1  2  2  0  1  2  2  1  0  1  1
 0  1  1  1  1  0  1  1  1  1  1  1  2  ...  1  1  0  0  1  2  2  2  1  0  2  2
 0  1  1  2  2  1  2  2  1  1  1  1  1  ...  1  2  1  1  0  1  2  2  1  0  1  0
 1  2  1  2  1  0  0  0  2  0  0  2  0  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  2  2  2  0  0  0  0  2  0  0  2  1  ...  2  0  2  1  0  0  1  1  0  1  1  0
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  1  1  2  2  2  2  0  0  2  2
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  1  2  1  1  0  1  2  2  1  0  1  1
 ⋮           ⋮           ⋮           ⋮           ⋮           ⋮
 0  2  0  2  2  0  1  1  1  1  1  1  2  ...  0  2  0  0  0  0  2  2  1  0  1  0
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  2  2  0  1  1  1  1  1  0  0
 0  1  2  2  2  1  2  2  1  1  1  0  0  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  1  2  1  1  0  1  1  1  1  1  0  0
 0  2  0  2  2  1  2  2  1  1  1  1  2  ...  2  1  2  1  0  1  2  2  1  0  0  0
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  1  2  1  1  1  2  2  2  1  0  1  1
 0  1  1  2  2  1  2  2  0  2  1  0  0  ...  0  2  0  0  0  0  2  2  0  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  1  2  1  1  0  1  2  2  1  0  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  2  1  1  2  2  2  1  0  1  1
 0  1  1  2  2  1  2  2  0  2  2  0  1  ...  2  2  2  2  0  2  2  2  2  0  0  0
 0  2  0  2  2  2  2  2  0  2  2  0  2  ...  2  0  2  2  0  1  2  2  1  0  1  0
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  1  1  1  1  1  1  1  1  0  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}:
```

```
44270
43584
41841
43933
43000
43150
42145
43675
43496
44068
43994
43676
42877
      :
76408
76197
76695
73693
76609
76599
75318
75897
75181
76482
75303
73369
```



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

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

```
44270
```

```
43584
```

```
41841
```

```
43933
```

```
43000
```

```
43150
```

```
42145
```

```
43675
```

```
43496
```

```
44068
```

```
43994
```

```
43676
```

```
42877
```

```
:
```

```
75318
```

```
74289
```

```
75784
```

```
75369
```

```
76487
```

```
76104
```

```
73629
```

```
74021
```

```
76008
```

```
75220
```

```
75572
```

```
76219
```

```
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}:  
 44270  
 43584  
 41841  
 43933  
 43000  
 43150  
 42145  
 43675  
 43496  
 44068  
 43994  
 43676  
 42877  
      ⋮  
 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]: (9000,2001)
```

```
In [62]: GSOFF5Row = size(GSOFF5,1)
```

```
Out[62]: 9000
```

```
In [63]: GSOFF5Col = size(GSOFF5,2)
```

```
Out[63]: 2001
```

```
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  31.027  31.438
          40723  32.518  30.726
          40724  30.166  30.302
          40725  28.332  29.905
          40726  30.18   30.611
          40727  30.293  31.801
          40728  28.191  29.339
          40729  29.057  31.357
          40730  31.07   29.149
          40731  30.746  30.73
          40732  27.685  29.413
          40733  28.904  30.288
          40734  31.699  31.693
              ⋮
          88710  32.442  33.081
          88711  31.504  32.314
          88712  34.862  35.049
          88713  34.015  34.031
          88714  33.557  33.652
          88715  33.819  34.291
          88716  34.543  33.456
          88717  35.001  33.898
          88718  33.352  34.222
          88719  36.231  34.857
          88720  32.767  33.963
          88721  33.166  34.408
```

```
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]: 500-element Array{Int64,1}:
```

```
 1
 8
 9
16
17
18
25
29
37
38
39
40
44
 ⋮
1956
1957
1960
1964
1977
1978
1979
1985
1990
1992
1995
1996
```



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

```
Out[83]: 1500-element Array{Int64,1}:
```

```
 2
 3
 4
 5
 6
 7
10
11
12
13
14
15
19
 :
1984
1986
1987
1988
1989
1991
1993
1994
1997
1998
1999
2000
```

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]: 9000x2000 Array{Int64,2}:
 0  2  1  2  1  0  0  0  2  0  0  1  1  ...  0  2  0  0  0  1  1  1  1  1  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  2  0  2  2  0  1  1  1  1  1  1  2  ...  1  2  1  1  1  1  2  2  0  0  2  1
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  1  1  2  2  0  1  1  1  1  1  0  0  ...  1  2  1  1  0  1  2  2  1  0  1  1
 0  1  1  2  2  1  1  1  1  1  1  1  2  ...  1  1  1  1  1  1  1  1  0  1  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  1  2  1  1  1  2  2  2  1  0  1  1
 0  2  0  2  2  0  0  0  2  0  0  0  0  ...  2  0  2  2  0  0  0  0  0  2  0  0
 0  2  1  2  1  0  0  0  2  0  0  2  2  ...  0  2  0  0  0  0  1  1  1  1  0  0
 0  2  0  2  2  0  1  1  1  1  1  1  2  ...  2  0  1  0  1  2  2  2  2  0  2  2
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  2  2  2  2  1  2  2  2  1  0  1  1
 0  1  1  2  1  1  1  1  1  1  1  0  0  ...  1  1  1  1  0  1  1  1  1  1  1  1
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  2  2  2  2  1  2  2  2  1  0  1  0
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 0  2  0  2  2  0  1  1  1  1  1  1  2  ...  0  2  0  0  0  0  2  2  1  0  1  0
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  2  2  0  1  1  1  1  1  0  0
 0  1  2  2  2  1  2  2  1  1  1  0  0  ...  1  2  1  1  0  1  2  2  1  0  1  0
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  1  2  1  1  0  1  1  1  1  1  0  0
 0  2  0  2  2  1  2  2  1  1  1  1  2  ...  2  1  2  1  0  1  2  2  1  0  0  0
 0  2  0  2  2  1  1  1  1  1  1  0  1  ...  1  2  1  1  1  2  2  2  1  0  1  1
 0  1  1  2  2  1  2  2  0  2  1  0  0  ...  0  2  0  0  0  0  2  2  0  0  2  1
 0  2  0  2  2  0  0  0  2  0  0  2  2  ...  1  2  1  1  0  1  2  2  1  0  1  1
 0  2  0  2  2  0  0  0  2  0  0  1  1  ...  2  1  2  1  1  2  2  2  1  0  1  1
 0  1  1  2  2  1  2  2  0  2  2  0  1  ...  2  2  2  2  0  2  2  2  2  0  0  0
 0  2  0  2  2  2  2  2  0  2  2  0  2  ...  2  0  2  2  0  1  2  2  1  0  1  0
 0  2  1  2  1  0  0  0  2  0  0  2  1  ...  1  1  1  1  1  1  1  1  0  1  1  1
```

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

```
Out[86]: 9000x500 Array{Int64,2}:
```

```
 0  0  2  1  1  1  0  1  1  0  0  0  1  ...  0  0  2  1  1  1  1  1  2  0  1  1
 0  0  2  1  1  1  1  2  2  0  0  0  2  ...  0  1  2  1  0  2  0  0  2  1  2  2
 0  1  1  1  0  1  1  1  0  0  0  1  1  ...  0  0  1  1  2  2  1  1  2  1  2  2
 0  0  2  1  1  1  0  1  0  0  1  1  1  ...  1  0  0  1  1  2  1  0  2  1  2  2
 0  1  1  0  2  1  0  2  1  1  1  1  2  ...  1  1  1  2  1  2  1  0  2  1  2  2
 0  1  1  1  1  0  0  1  1  0  1  1  2  ...  0  0  1  2  0  1  1  1  1  1  1  1
 0  0  2  1  0  1  1  1  1  0  0  0  1  ...  0  2  2  2  0  1  1  0  2  1  2  2
 0  0  2  0  2  0  0  2  2  1  1  0  1  ...  0  1  1  1  0  2  0  2  0  2  0  0
 0  0  2  1  0  1  1  1  0  0  0  0  1  ...  0  0  2  2  0  0  2  0  2  0  1  1
 0  1  1  2  1  1  0  1  1  0  0  0  1  ...  1  0  2  1  1  2  1  2  0  0  2  2
 0  0  2  1  0  1  0  0  1  1  1  0  0  ...  0  1  1  2  0  2  0  1  2  2  2  2
 0  1  1  1  1  0  0  1  2  0  0  0  2  ...  0  0  1  1  0  2  0  1  1  1  1  1
 0  1  1  1  0  0  0  1  1  1  1  0  0  ...  0  2  2  2  1  2  0  1  2  2  2  2
 ⋮          ⋮          ⋮          ⋮          ⋮          ⋮
 0  1  1  2  1  1  0  1  0  1  1  0  1  ...  1  0  1  1  1  1  2  0  2  0  2  2
 0  0  2  1  1  0  0  2  2  0  0  0  2  ...  0  1  1  2  0  2  0  1  1  2  1  1
 0  2  1  1  2  0  0  2  1  1  1  0  2  ...  1  2  2  1  1  2  1  1  2  1  2  2
 0  1  1  1  1  0  0  2  2  0  0  0  2  ...  0  0  1  1  0  2  0  0  2  1  1  1
 0  2  1  1  1  1  0  2  1  1  1  0  2  ...  0  2  2  1  1  2  0  1  1  1  2  2
 0  1  1  0  2  0  1  2  2  0  0  0  2  ...  0  1  2  1  2  2  1  0  2  1  2  2
 0  2  0  1  2  1  1  2  1  0  0  0  2  ...  0  1  1  1  2  2  1  0  2  0  2  2
 0  0  2  2  0  2  0  1  0  1  1  0  1  ...  0  2  2  2  1  2  0  0  2  1  2  2
 0  0  2  1  1  1  1  1  0  0  0  0  1  ...  0  1  2  2  1  2  1  1  1  1  2  2
 0  2  0  2  1  0  0  2  1  2  2  0  1  ...  0  0  2  2  0  1  1  0  2  2  2  2
 0  2  0  2  0  0  0  2  2  2  2  0  0  ...  0  2  2  1  0  2  0  2  0  2  2  2
 0  0  2  1  0  1  0  0  1  2  2  0  2  ...  0  0  1  2  0  1  1  1  1  1  1  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]: 0.6780247936141927
```

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

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

```
Out[100]: 0.824094370565869
```

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

```
Out[101]: 48000x500 Array{Int64,2}:
```

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

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

```
Out[102]: 500-element Array{Float64,1}:
```

```
0.063266  
0.0639862  
0.0632293  
0.063651  
0.063939  
0.0635984  
0.0638471  
0.0631333  
0.0638577  
0.0626122  
0.0633143  
0.0641923  
0.06392  
:  
0.0636605  
0.0617921  
0.0636174  
0.0644281  
0.0619862  
0.0637579  
0.0626133  
0.0624834  
0.063202  
0.06328  
0.0641241  
0.0617969
```

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

```
Out[103]: 48000-element Array{Float64,1}:  
 31.351  
 30.661  
 30.2391  
 29.8456  
 30.5421  
 31.723  
 29.2776  
 31.2978  
 29.0891  
 30.6444  
 29.3385  
 30.221  
 31.6201  
  ⋮  
 33.0087  
 32.2504  
 34.9709  
 33.9496  
 33.5679  
 34.2113  
 33.4085  
 33.8212  
 34.1518  
 34.7845  
 33.8856  
 34.3237
```

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

```
Out[104]: 30.252975249922255
```

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

```
Out[105]: 30.794711928831962
```

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

```
Out[106]: 31.39736566010726
```



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

```
Out[107]: 31.981151963722862
```

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

```
Out[108]: 32.510080518346136
```

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

```
Out[109]: 33.067297558403354
```

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

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

```
30.661
32.2517
30.5979
31.6066
31.3524
31.3727
32.1269
30.86
30.5992
31.1054
31.7354
31.2343
31.3091
⋮
33.0087
32.2504
34.9709
33.9496
33.5679
34.2113
33.4085
33.8212
34.1518
34.7845
33.8856
34.3237
```

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

```
Out[111]: 33.00445751898185
```

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

```
Out[112]: 2.751482269059597
```

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

```
Out[113]: 31.336808098878702
```

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

```
Out[114]: 1.0838328489564475
```

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

```
Out[115]: 31.969310041722228
```

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

```
Out[116]: 1.716334791799973
```

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

```
Out[117]: 32.55047030064906
```

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

```
Out[118]: 2.2974950507268055
```

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

```
Out[119]: 33.03471790830986
```

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

```
Out[120]: 2.7817426583876035
```

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

```
Out[121]: 33.61737966848915
```

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

```
Out[122]: 3.3644044185668953
```

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

```
Out[123]: 33.067297558403354
```

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

```
Out[124]: 2.814322308481099
```