Simulating Mendel's Experiments

Philipp Heilmann

2026-02-10

1 Introduction to Genetic Simulation

1.1 Form of seeds: Mendel's experiments, one trait

The SelectionTools contain simulation software. We start by simulating Mendel's experiments. The first step of a simulation is to define the genome and locus positions that should be simulated. We have to define a data.frame that contains one locus per row. Each locus (i.e. each row) requires the chromosome the locus is on, it's position on the chromosome, a name and a class.
The locus name can be chosen arbitrary, and the class name describes whether the locus is a marker or a trait. The class name is important later for selection. An arbitrary number of loci can be defined, each in a new row.

```
map <- data.frame(chrom = 1,
                  pos = 0.0,
                  name = "form")
define.genome(map)
```

```
Warning: Chromosome length cannot be 0; increased to 0.01
```

```
M: 1 chromosomes defined
```

The number of chromosomes and their respective lengths are inferred from the data we provided.

Now we define the parental lines of the cross. They have the names P1 and P2 and are homozygous for the two alleles 1 and 2, respectively. Alleles must be coded with numbers, not with characters. The 1 stands for round seeds and the 2 for wrinkled. We know from previous experiments that round is dominant over wrinkled.

```
init.population("P1", homozygote(1))
init.population("P2", homozygote(2))
```

Then we cross the parental lines, the first argument of the cross command is the name of the newly generated population followed by the name of the two parental populations. The number at the end is the number of progenies.

```
cross("F1","P1", "P2", 563)
```

We now look at our populations. The first argument of the evaluate.genotype command is the name of the population, the second is the locus name that should be evaluated. P1 consists of one individual that carries at both homologous chromosomes at the locus form the allele 1.

```
evaluate.genotype2("P1","form")
```

```
   form-1  form-2  count frequency
1       1       1     1         1
```

evaluate.genotype(“P1”,“form”)

```
evaluate.genotype2("P2","form")
```

```
   form-1  form-2  count frequency
1       2       2     1         1
```

and F1 consists of 563 plants that carry one copy of allele 1 and one copy of allele 2. Because round is dominant, they are all round.

```
evaluate.genotype("F1","form")
```

```
   form-1  form-2  count frequency
1       1       2   563         1
```

Mendel’s 1 law: *The progenies from crosses of homozygous lines are uniform and heterozygous.*

We take a sample of 252 out of the 563 F1 plants

```
sample.population ("SampleFromF1","F1",252)
```

and self these. From the selfing we get 7223 seeds.

```
cross("F2","SampleFromF1","SampleFromF1",7323,self=1)
```

We count the wrinkled seeds and find 1853 of them.

```
evaluate.genotype("F2","form")
```

```
evaluate.genotype("F2","form")
```

```
   form-1  form-2  count frequency
1       1       1   1856 0.2534480
2       1       2   3638 0.4967909
3       2       2   1829 0.2497610
```

Mendel's 2 law: *After selfing the F1s from crosses of homozygous lines, the progenies in the F2 are segregating.*

On the genotypic level (which we simulate) the segregation ratio is 1:2:1. Depending on dominant/recessive or co-dominant inheritance this is a segregation of either 3:1 or 1:2:1 for the phenotypes.

1.2 Color and form of seeds: Mendel's experiments, two raits

We now consider two traits. We first assume that the two loci underlying the traits form and color are located on two different chromosomes and generate an F2 population:

```
reset.all()

map <- data.frame(chrom = 1:2,
                  pos = c(0.0, 0.0),
                  name = c("form","color"))
define.genome(map)
```

```
Warning: Chromosome length cannot be 0; increased to 0.01

M: 2 chromosomes defined
```

```
init.population("P1",homozygote(1))
init.population("P2",homozygote(2))
cross("F1","P1","P2",10)
cross("F2","F1","F1",1000)

evaluate.genotype2("F2", map$name)
```

```
   form-1  form-2  color-1  color-2  count frequency
1       1       1        1        1     53     0.053
2       1       1        1        2    128     0.128
3       1       1        2        2     68     0.068
4       1       2        1        1    116     0.116
5       1       2        1        2    261     0.261
6       1       2        2        2    127     0.127
7       2       2        1        1     62     0.062
8       2       2        1        2    121     0.121
9       2       2        2        2     64     0.064
```

```
evaluate.genotype2("F2", map$name, mode=0)
```

```
    form-1  form-2  color-1  color-2  count frequency
1        1       1        1        1     53     0.053
2        1       1        1        2     66     0.066
3        1       1        2        1     62     0.062
4        1       1        2        2     68     0.068
5        1       2        1        1     56     0.056
6        1       2        1        2     62     0.062
7        1       2        2        1     70     0.070
8        1       2        2        2     70     0.070
9        2       1        1        1     60     0.060
10       2       1        1        2     74     0.074
11       2       1        2        1     55     0.055
12       2       1        2        2     57     0.057
13       2       2        1        1     62     0.062
14       2       2        1        2     68     0.068
15       2       2        2        1     53     0.053
16       2       2        2        2     64     0.064
```

***Mendel's 3 law**: Genes are inherited independently.*

1.3 Backcrossing and recombination

Bacrossing to the recessive parent is much easier to interpret:

```
reset.all()

map <- data.frame(chrom = 1:2,
                  pos = c(0.0, 0.0),
                  name = c("form","color"))
```

```
define.genome(map)

init.population("P1",homozygote(1))
init.population("P2",homozygote(2))
cross ("F1","P1","P2",563)
cross ("BC1","F1","P2",1000)
evaluate.genotype2("BC1", map$name)
```

In our simulated BC1 population, we observe the four phenotypes with approximately the same frequencies:

```
   form-1  form-2  color-1  color-2  count frequency
1       1       2        1        2   249     0.249
2       1       2        2        2   260     0.260
3       2       2        1        2   238     0.238
4       2       2        2        2   253     0.253
```

Now the two loci are on the same chromosome, the first at the telomere, and the second in a map distance of 0.1 M:

```
   map <- data.frame(chrom = c(1,1),
                pos = c(0.0, 0.1),
                name = c("form","color"),
                class = c("trait1","trait2"))
  define.genome(map)
  ...
```

```
reset.all()

map <- data.frame(chrom = c(1,1),
                  pos = c(0.0, 0.0),
                  name = c("form","color"))
define.genome(map)

init.population("P1",homozygote(1))
init.population("P2",homozygote(2))
cross ("F1","P1","P2",563)
cross ("BC1","F1","P2",1000)
evaluate.genotype2("BC1", map$name)
```

```
   form-1  form-2  color-1  color-2  count frequency
```

```
1        1       2        1        2   490      0.49
2        2       2        2        2   510      0.51
```

Recombination frequencies *can be estimated in backcross populations by summing up the frequency of recombinant gametes.*

2 Random Mating Populations

2.1 F2 Population

We start with an F2 population of peas and look at the genotype frequencies and allele frequencies of the seed form.

```
map <- data.frame(1,0.01,"form")
define.genome(map)
```

```
M: 1 chromosomes defined
```

```
init.population("P1",homozygote(1))
init.population("P2",homozygote(2))
cross("F1","P1","P2",1)
cross("F2","F1","F1",1000)
evaluate.genotype2 ("F2","form")
```

```
   form-1  form-2  count frequency
1        1       1   241     0.241
2        1       2   523     0.523
3        2       2   236     0.236
```

```
evaluate.allele.freq2("F2","form")
```

```
   form count   freq
1     1  1005 0.5025
2     2   995 0.4975
```

We carry out random mating and investigate genotype and allele frequencies:

```
cross("SYN1","F2","F2",1000)
cross("SYN2","SYN1","SYN1",1000)
cross("SYN3","SYN2","SYN2",1000)
evaluate.genotype2("SYN3","form")
```

```
   form-1  form-2  count frequency
1       1       1   240     0.240
2       1       2   521     0.521
3       2       2   239     0.239
```

```
evaluate.allele.freq2("SYN3","form")
```

```
   form count   freq
1     1  1001 0.5005
2     2   999 0.4995
```

2.2 Population of homozygous lines

We start with a population of homozygous lines and look at the genotype and allele frequencies after one generation of random mating:

```
remove.all.populations()
init.population("P11",homozygote(1,500))
init.population("P22",homozygote(2,500))
append.population("ADM","P11")
append.population("ADM","P22")
evaluate.genotype2 ("ADM","form")
```

```
   form-1  form-2  count frequency
1       1       1   500       0.5
2       2       2   500       0.5
```

```
evaluate.allele.freq2("ADM","form")
```

```
   form count freq
1     1  1000  0.5
2     2  1000  0.5
```

```
cross("SYN1","ADM","ADM",1000)
evaluate.genotype("SYN1","form")
```

```
   form-1  form-2  count frequency
1       1       1   228     0.228
2       1       2   487     0.487
3       2       2   285     0.285
```

```
evaluate.allele.freq("SYN1","form")
```

```
   form count   freq
1     1   943 0.4715
2     2  1057 0.5285
```

***Hardy-Weinberg proportions**: $p^2 : 2pq : q^2$ are reached in one generation of random mating, irrespective of the gentype frequencies in the base population.*

2.3 Arbitrary allele frequencies

```
remove.all.populations()
init.population("P11",homozygote(1,100))
init.population("P22",homozygote(2,900))
append.population("ADM","P11")
append.population("ADM","P22")
evaluate.genotype2 ("ADM","form")
```

```
   form-1  form-2  count frequency
1       1       1   100       0.1
2       2       2   900       0.9
```

```
evaluate.allele.freq2("ADM","form")
```

```
   form count freq
1     1   200  0.1
2     2  1800  0.9
```

```
cross("SYN1","ADM","ADM",1000)
evaluate.genotype("SYN1","form")
```

```
   form-1  form-2  count frequency
1       1       1      5     0.005
2       1       2    176     0.176
3       2       2    819     0.819
```

```
evaluate.allele.freq("SYN1","form")
```

```
   form count  freq
1     1   186 0.093
2     2  1814 0.907
```

```
cross("SYN1","ADM","ADM",1000)
cross("SYN2","SYN1","SYN1",1000)
cross("SYN3","SYN2","SYN2",1000)
evaluate.genotype("SYN3","form")
```

```
   form-1  form-2  count frequency
1       1       1     14     0.014
2       1       2    181     0.181
3       2       2    805     0.805
```

```
evaluate.allele.freq("SYN3","form")
```

```
   form count   freq
1     1   209 0.1045
2     2  1791 0.8955
```

Hardy-Weinberg law *is valid for arbitrary allele frequencies* p *and* q*.*

3 Populations under selection

3.1 Selection against a recessive allele

We start with a population of peas in HWE in which the recessive allele 'wrinkeld', which is coded as allele '2' has the allele frequency $q \approx 0.1$

```
reset.all()
map <- data.frame(1,0.0,"form")
define.genome(map)
```

```
Warning: Chromosome length cannot be 0; increased to 0.01
```

```
M: 1 chromosomes defined
```

```
init.population("P11",homozygote(1,900))
init.population("P22",homozygote(2,100))
append.population("ADM","P11")
append.population("ADM","P22")
cross("SYN1","ADM","ADM",1000)
evaluate.genotype2("SYN1","form")
```

```
   form-1  form-2  count frequency
1       1       1   803     0.803
2       1       2   185     0.185
3       2       2    12     0.012
```

```
evaluate.allele.freq2("SYN1","form")
```

```
   form count   freq
1     1  1791 0.8955
2     2   209 0.1045
```

We now want to select all round seeds. This means, we select against the recessive allele. For simulating selection, effects are required. An effect assigns values to alleles, for this assignment the class names of the loci are used.

~~If an individual has at a locus that belongs to the class trait1 the allele 1 then it gets a uniform value of 1 added to its genotypic value. This effect definition is stored under name effect1. Often effects are defined for quantitative characters, there a population mean is needed. In our example we don't need a population mean, this is the first 0 in the effect definition.~~

With this effect definition, a genotypic value for each individual of the population can be calculated. Those individuals that carry two times the allele 1 at the locus form, get a 2, those having one copy of the allele get a 1, and those that don't carry the allele 1 get a zero. We generate a new population called Allrounds that consists of individuals selected from SYN1 on basis of the effect called effect1. ~~The selection consists of all individuals~~

~~that belong to the best two classes, this is the 2 at the end of the command. The best two classes are the homozygous individuals (carrying two copies of the allele 1), and the heterozygous (carrying one copy).~~

We are interested in all genotypes that have the round phenotype. This is the case if the genotype is carrying allele 1 at least once. We pass the trait and the allele combinations that we want to select as a data.frame to the function. Here, 1|1 and 1|2 can result in round seeds. Effects are estimated automatically and then selection is carried out. The selected genotypes end up in a new population called "allrounds".

```
sel.crit.form <- data.frame(
  locus    =  c("form", "form") ,
  allele1  = c(1,1),
  allele2  = c(1,2)
)

sel.crit.form
```

```
  locus allele1 allele2
1  form       1       1
2  form       1       2
```

```
select.genotypes("allrounds","SYN1",
                 sel.crit.form)
```

We look at the genotype of the selected plants:

```
evaluate.genotype2("allrounds","form")
```

```
   form-1  form-2  count frequency
1       1       1   803  0.812753
2       1       2   185  0.187247
```

We now grow the seeds and random mate the plants.

```
cross ("SYNA","allrounds","allrounds",1000)
evaluate.genotype2 ("SYNA","form")
```

```
   form-1  form-2  count frequency
1       1       1   840     0.840
2       1       2   149     0.149
3       2       2    11     0.011
```

```
evaluate.allele.freq2("SYNA","form")
```

```
   form count   freq
1     1  1829 0.9145
2     2   171 0.0855
```

The selection against wrinkeld seed was not working well! However, it is difficult to draw conclusions just from one simulation. We repeat our simulation 100 times:

```
reset.all()
map <- data.frame(1,0.0,"form")
define.genome(map)
```

```
M: 1 chromosomes defined
```

```
for (i in 1:100){
  init.population("P11",homozygote(1,900))
  init.population("P22",homozygote(2,100))
  append.population("ADM","P11")
  append.population("ADM","P22")
  cross("SYN1","ADM","ADM",1000)
  select.genotypes("allrounds","SYN1",
                sel.crit.form)
  cross ("SYNA","allrounds","allrounds",1000)
  append.population("STORE","SYNA")
}
```

Now the numbers are easier to interpret:

```
evaluate.genotype("STORE","form")
```

```
   form-1  form-2  count frequency
1       1       1 82821   0.82821
2       1       2 16354   0.16354
3       2       2   825   0.00825
```

```
evaluate.allele.freq("STORE","form")
```

```
   form  count    freq
1     1 181996 0.90998
2     2  18004 0.09002
```

Selection against recessive alleles: *Even if strong selection is carried out against those individuals of a population, that show the phenotype of a recessive allele, this phenotype will reoccur in subsequent generations.*

3.2 Selection for fitness

A human population live in a Malaria region. Assume the allele frequency of the sickle cell anemia gene is $f(S) \approx 0.1$.

```
reset.all()
map <- data.frame(1,0.0,"SZA","trait1")
define.genome(map)
```

```
Warning: Chromosome length cannot be 0; increased to 0.01
```

```
M: 1 chromosomes defined
```

```
init.population("P11",homozygote(1,90))
init.population("P22",homozygote(2,10))
append.population("ADM","P11")
append.population("ADM","P22")
cross("SYN1","ADM","ADM",1000)
evaluate.genotype("SYN1","SZA")
```

```
   SZA-1  SZA-2  count frequency
1      1      1   802     0.802
2      1      2   182     0.182
3      2      2    16     0.016
```

```
evaluate.allele.freq("SYN1","SZA")
```

```
   SZA count  freq
1    1  1786 0.893
2    2   214 0.107
```

We first split the population in three subpopulations, corresponding to the genotype. As selected genotypes are removed from the populations, we first remove both homozygous genotypes. What remains in the original dataset are the heterozygous genotypes.

```
select.genotypes("R2","SYN1",data.frame("SZA",1,1))
evaluate.genotype("R2","SZA")
```

```
   SZA-1  SZA-2  count frequency
1       1      1   802         1
```

```
select.genotypes("R0","SYN1",data.frame("SZA",2,2))
```

```
Warning in define.effects.df(n): Effects already exist for SZA, using
preexisting effects.
```

```
evaluate.genotype("R0","SZA")
```

```
   SZA-1  SZA-2  count frequency
1       2      2    16         1
```

```
rename.population ("SYN1","R1")
evaluate.genotype("R0","SZA")
```

```
   SZA-1  SZA-2  count frequency
1       2      2    16         1
```

Under strong malaria pressure the probability that an individual contributes to the next generation are

```
w2 <- 0.5 # Homozygous normal, allele 1
w1 <- 1.0 # Heterozygous
w0 <- 0.2 # Homozygous sickle cell anemia, allele 2
```

Using random numbers for the binomial distribution generate the fraction of the population that contributes to the next generation

Hier fehlt irgendwas. Gleich wird n.inds benutzt, die werden aber nirgendwo generiert. Daher wird dieser Code gerade nicht ausgeführt.

```
( nR2 <- n.inds$count[n.inds$PopName=="R2"])
( nR1 <- n.inds$count[n.inds$PopName=="R1"])
( nR0 <- n.inds$count[n.inds$PopName=="R0"])
( sR2 <- rbinom(n=1,size=nR2,prob=w2) )
( sR1 <- rbinom(n=1,size=nR1,prob=w1) )
( sR0 <- rbinom(n=1,size=nR0,prob=w0) )
sample.population("S2","R2",sR2)
sample.population("S1","R1",sR1)
sample.population("S0","R0",sR0)
remove.population("Sx")
append.population("Sx","S2")
append.population("Sx","S1")
append.population("Sx","S0")
evaluate.genotype2 ("Sx","SZA")
evaluate.allele.freq2("Sx","SZA")
```

The next generation:

```
cross ("SYNA","Sx","Sx",1000)
evaluate.genotype2 ("SYNA","SZA")
evaluate.allele.freq2("SYNA","SZA")
```

4 New functionality: Selecting multiple traits

We have a large population of peas that show all different kinds of phenotypes. Their peas can either be yellow or green and they can be round or wrinkled. Additionally, their flowers can either be white, red or pink (when heterozygous).

```
reset.all()
sel.crit.wrinkeled <- data.frame(
  locus    = "form" ,
  allele1  = 2       ,
  allele2  = 2
)

sel.crit.yellow <- data.frame(
  locus    = c ("color" , "color"),
  allele1  = c (1       ,  1     ),
  allele2  = c( 1       ,  2     )
)
```

```
sel.crit.pink <- data.frame(
  locus    = "flower" ,
  allele1  = 1        ,
  allele2  = 2
)
criteria <- rbind (sel.crit.yellow, sel.crit.wrinkeled, sel.crit.pink)
```

We want to select: wrinkled, green peas with pink flowers. We create a population with three loci on three chromosomes:

```
map <- data.frame(chrom = c(1,2,3),
                  pos = c(0.01,0.05,0.1),
                  name = c("form","color","flower"))
define.genome(map)
```

```
M: 3 chromosomes defined
```

Some cycles of random mating:

```
init.population("P1",homozygote(1))
init.population("P2",homozygote(2))
cross("F1","P1","P2",1)
cross("F2","F1","F1",1000)
cross("F3","F2","F2",1000)
```

And now our selection:

```
select.genotypes("Sel","F3", criteria)
```

```
evaluate.genotype2("F3", map$name)
```

```
   form-1  form-2  color-1  color-2  flower-1  flower-2  count  frequency
1       1       1        2        2         1         1     12 0.04858300
2       1       1        2        2         1         2     20 0.08097166
3       1       1        2        2         2         2     22 0.08906883
4       1       2        2        2         1         1     26 0.10526316
5       1       2        2        2         1         2     67 0.27125506
6       1       2        2        2         2         2     36 0.14574899
7       2       2        2        2         1         1     20 0.08097166
8       2       2        2        2         1         2     31 0.12550607
9       2       2        2        2         2         2     13 0.05263158
```

```
evaluate.genotype2("Sel", map$name)
```

```
  form-1 form-2 color-1 color-2 flower-1 flower-2 count frequency
1      2      2       1       1        1        2    26 0.6842105
2      2      2       1       2        1        2    12 0.3157895
```

```
criteria
```

```
  locus allele1 allele2
1 color       1       1
2 color       1       2
3  form       2       2
4 flower      1       2
```