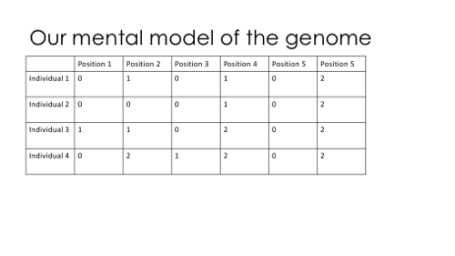
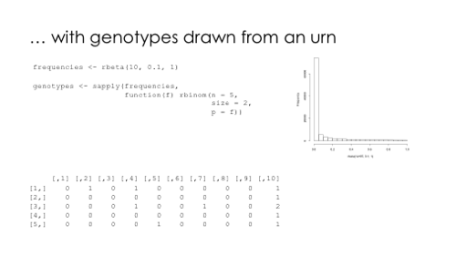
A few weeks ago, I gave [a talk at the Edinburgh R users group EdinbR](http://edinbr.org/edinbr/2019/03/12/march-meeting.html) on the [RAGE paper](https://onunicornsandgenes.blog/2019/04/28/paper-removal-of-alleles-by-genome-editing-rage-against-deleterious-load/). Since this is an R meetup, the talk concentrated on the mechanics of genetic data simulation and with the paper as a case study. I showed off some of what Chris Gaynor’s AlphaSimR can do, and how we built on that to make the specifics of this simulation study. The slides are on the [EdinbR Github](https://github.com/EdinbR/edinbr-talks/tree/master/2019-03-20).

Genetic simulation is useful for all kinds of things. Sure, they’re only as good as the theory that underpins them, but the willingness to try things out in simulations is one of the things I always liked about breeding research.

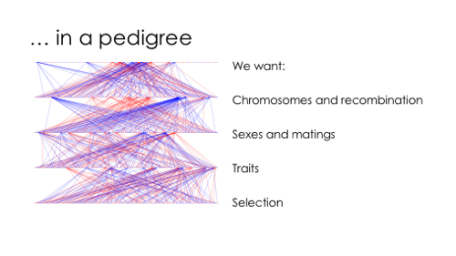
This is my description of the logic of genetic simulation: we think of the genome as a large table of genotypes, drawn from some distribution of allele frequencies.

[](https://martinsbioblogg.files.wordpress.com/2019/06/slide04.png)

To make an utterly minimal simulation, we could draw allele frequencies from some distribution (like a Beta distribution), and then draw the genotypes from a binomial distribution. Done!

[](https://martinsbioblogg.files.wordpress.com/2019/06/slide05.png)

However, there is a ton of nuance we would like to have: chromosomes, linkage between variants, sexes, mating, selection …

[](https://martinsbioblogg.files.wordpress.com/2019/06/slide06.png)

AlphaSimR addresses all of this, and allows you to throw individuals and populations around to build pretty complicated designs. Here is the small example simulation I used in the talk.

library(AlphaSimR)

library(ggplot2)

## Generate founder chromsomes

FOUNDERPOP <- runMacs(nInd = 1000,

nChr = 10,

segSites = 5000,

inbred = FALSE,

species = "GENERIC")

## Simulation parameters

SIMPARAM <- SimParam$new(FOUNDERPOP)

SIMPARAM$addTraitA(nQtlPerChr = 100,

mean = 100,

var = 10)

SIMPARAM$addSnpChip(nSnpPerChr = 1000)

SIMPARAM$setGender("yes\_sys")

## Founding population

pop <- newPop(FOUNDERPOP,

simParam = SIMPARAM)

pop <- setPheno(pop,

varE = 20,

simParam = SIMPARAM)

## Breeding

print("Breeding")

breeding <- vector(length = 11, mode = "list")

breeding[[1]] <- pop

for (i in 2:11) {

print(i)

sires <- selectInd(pop = breeding[[i - 1]],

gender = "M",

nInd = 25,

trait = 1,

use = "pheno",

simParam = SIMPARAM)

dams <- selectInd(pop = breeding[[i - 1]],

nInd = 500,

gender = "F",

trait = 1,

use = "pheno",

simParam = SIMPARAM)

breeding[[i]] <- randCross2(males = sires,

females = dams,

nCrosses = 500,

nProgeny = 10,

simParam = SIMPARAM)

breeding[[i]] <- setPheno(breeding[[i]],

varE = 20,

simParam = SIMPARAM)

}

## Look at genetic gain and shift in causative variant allele frequency

mean\_g <- unlist(lapply(breeding, meanG))

sd\_g <- sqrt(unlist(lapply(breeding, varG)))

plot\_gain <- qplot(x = 1:11,

y = mean\_g,

ymin = mean\_g - sd\_g,

ymax = mean\_g + sd\_g,

geom = "pointrange",

main = "Genetic mean and standard deviation",

xlab = "Generation", ylab = "Genetic mean")

start\_geno <- pullQtlGeno(breeding[[1]], simParam = SIMPARAM)

start\_freq <- colSums(start\_geno)/(2 \* nrow(start\_geno))

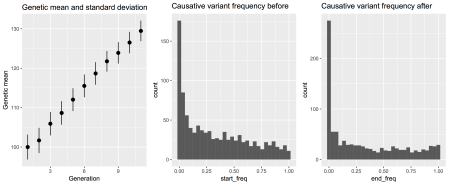
end\_geno <- pullQtlGeno(breeding[[11]], simParam = SIMPARAM)

end\_freq <- colSums(end\_geno)/(2 \* nrow(end\_geno))

plot\_freq\_before <- qplot(start\_freq, main = "Causative variant frequency before")

plot\_freq\_after <- qplot(end\_freq, main = "Causative variant frequency after")

This code builds a small livestock population, breeds it for ten generations, and looks at the resulting selection response in the form of a shift of the genetic mean, and the changes in the underlying distribution of causative variants. Here are the resulting plots:

[](https://martinsbioblogg.files.wordpress.com/2019/06/blogpost.png)