Figure 1: Figure 1: A typical unicorn.

genedropper: conduct a gene-drop analysis through a pedigree.

Susan Johnston, May 2019

A "gene-drop" simulation approach can model expected changes in allele frequencies due to genetic drift, given an identical pedigree structure.

To install this package, you can use the `install_github` command in `library(devtools)`:

```
library(devtools)
install_github("susjoh/genedroppeR")
```

NB This is a work in progress. More description will be given in time. The functions are also slower than they could be - we are working on it.

Things to add: * Weighted regression * Selection

Do you have comments, polite criticism, feedback, helpful suggestions? Please contact me at Susan.Johnston (at) ed.ac.uk.

Loading the library and examining the example dataset.

```
library(genedroppeR)
```

The library comes with an example dataset from a long-term study of unicorns on the island of Áiteigin off the South coast of Scotland (Figure 1). They have been genotyped for three loci: HornSNP, a major quantitative trait locus for horn length; MHC, encompassing haplotype variation at the Magic Histocompatibility Locus; and ColourSNP, a SNP responsible for a rare glitter colour polymorphism.

```
data(unicorn)
str(unicorn)
```

```
## 'data.frame':    3951 obs. of  7 variables:
##  $ id       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ mother   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ father   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ cohort   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ HornSNP  : Factor w/ 3 levels "AA","GA","GG": NA 3 NA 2 2 3 1 NA 2 1 ...
##  $ MHC      : Factor w/ 16 levels "AA","AB","AC",..: NA 7 11 8 6 4 NA 12 10 10 ...
##  $ ColourSNP: Factor w/ 3 levels "AA","AB","BB": NA 3 NA NA NA NA 3 3 3 NA ...
```

Unicorns have a promiscuous mating system, in common with other magical beasts. We can see an example of the pedigree below. The numbers at the side indicate the cohorts of when each unicorn was born. This can be any sequential series of numbers.

We can summarise this data in terms of the allele frequency changes using the function `summary_cohort()`. Let's start with the HornSNP locus.

```
unicorn.summ <- summary_cohort(id = unicorn$id,
                               mother = unicorn$mother,
                               father = unicorn$father,
                               cohort = unicorn$cohort,
                               genotype = unicorn$HornSNP)

unicorn.summ
```

```
##    cohort         A         G GenoCount FullCount PropGenotyped
## 1       1 0.4090909 0.5909091        33        47     0.7021277
## 2       2 0.3918919 0.6081081        74       105     0.7047619
## 3       3 0.3939394 0.6060606        33        99     0.3333333
## 4       4 0.4611111 0.5388889        90       103     0.8737864
## 5       5 0.3818898 0.6181102       127       144     0.8819444
## 6       6 0.3721591 0.6278409       176       188     0.9361702
## 7       7 0.3850575 0.6149425       174       181     0.9613260
## 8       8 0.3954082 0.6045918       196       206     0.9514563
## 9       9 0.4184783 0.5815217       184       194     0.9484536
## 10     10 0.4139785 0.5860215       186       190     0.9789474
## 11     11 0.3985149 0.6014851       202       236     0.8559322
## 12     12 0.4243119 0.5756881       218       231     0.9437229
## 13     13 0.4400826 0.5599174       242       255     0.9490196
## 14     14 0.4540541 0.5459459       185       187     0.9893048
## 15     15 0.4537037 0.5462963       216       219     0.9863014
```

```
## 16     16 0.4466912 0.5533088       272       343     0.7930029
## 17     17 0.4523810 0.5476190       147       204     0.7205882
## 18     18 0.4782609 0.5217391       253       272     0.9301471
## 19     19 0.4506803 0.5493197       294       322     0.9130435
## 20     20 0.4581281 0.5418719       203       225     0.9022222
## 21     NA       NaN       NaN         0         0           NaN
##    NonFounders Founders PropFounders
## 1            0       47   1.00000000
## 2           28       77   0.73333333
## 3           70       29   0.29292929
## 4           52       51   0.49514563
## 5           99       45   0.31250000
## 6          139       49   0.26063830
## 7          151       30   0.16574586
## 8          188       18   0.08737864
## 9          187        7   0.03608247
## 10         176       14   0.07368421
## 11         224       12   0.05084746
## 12         220       11   0.04761905
## 13         244       11   0.04313725
## 14         167       20   0.10695187
## 15         202       17   0.07762557
## 16         306       37   0.10787172
## 17         187       17   0.08333333
## 18         252       20   0.07352941
## 19         316        6   0.01863354
## 20         214       11   0.04888889
## 21          NA       NA           NA
```

This output provides the allele frequencies for the A and G alleles per cohort, as well as a count of the number of genotyped individuals and the total number of individuals per cohort, the proportion of genotyped individuals, the numbers of non founders and founders, and the proportion of founders per cohort. A founder is defined as an individual where neither the mother or father is known.

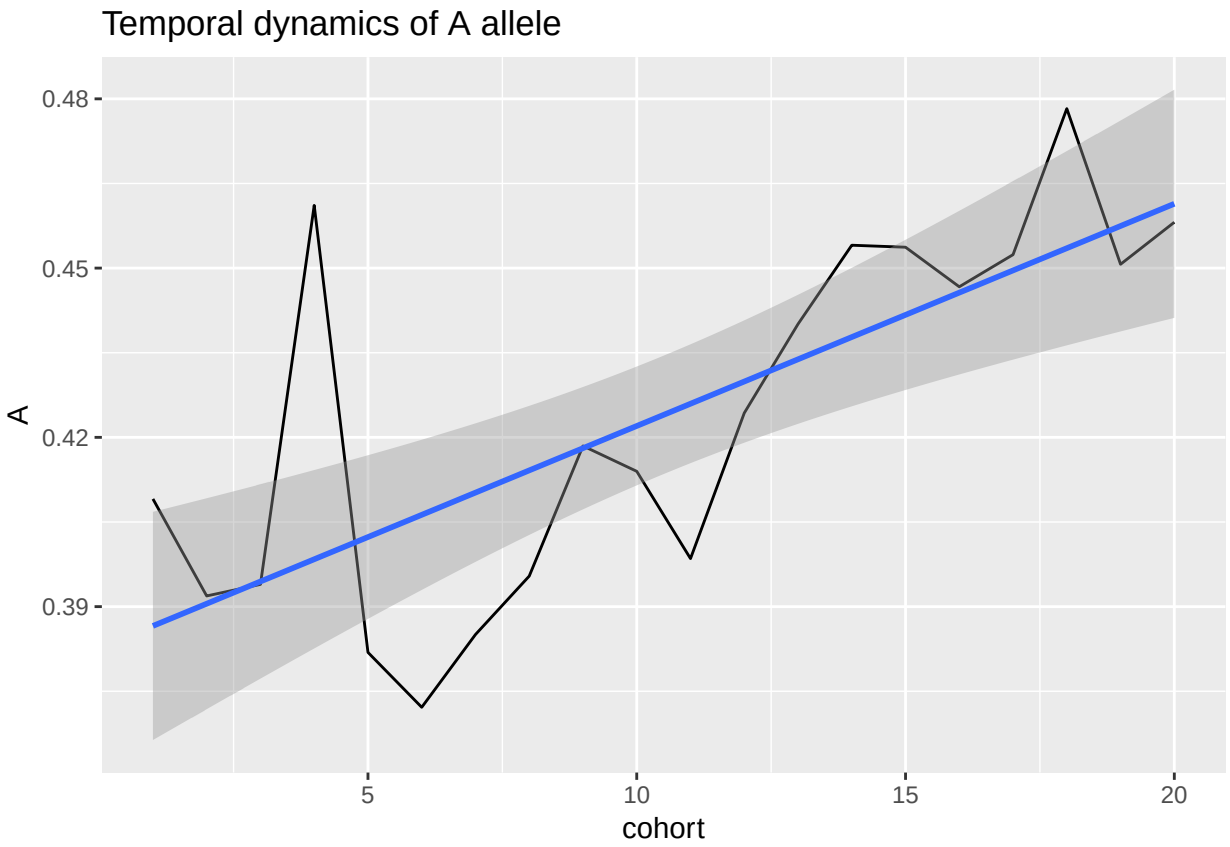
The frequency of the A allele, which confers a larger horn, seems to be increasing in the population:

```
library(ggplot2)

ggplot(unicorn.summ, aes(cohort, A)) +
  geom_line() +
  stat_smooth(method = "lm") +
  ggtitle("Temporal dynamics of A allele")
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```



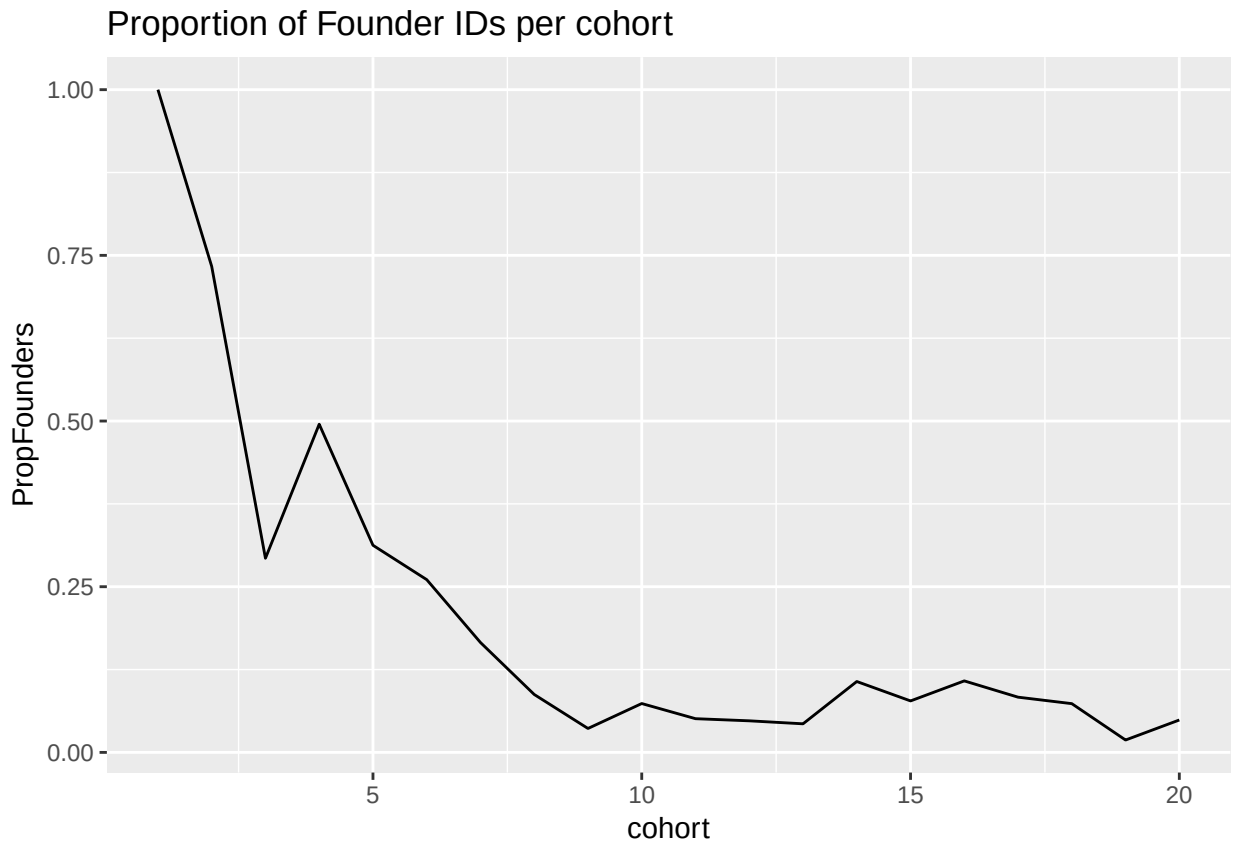
However, given the complex structure of the pedigree, it may be that this increase in allele frequency is more likely to be due to drift than selection. One approach to model this is to simulate allele frequency changes given a pedigree of the same structure, and see if the observed allele frequency change is significantly different from what we would expect by chance.

Example 1: The *Horns* locus.

In early cohorts, many individuals are likely to be founders. To run the gene-drop, we select some cohorts at the start to be the "founder" cohorts, where allele frequencies are sampled from the observed frequencies in each year. To determine which cohorts we define as founder cohorts, we need to explore the proportion of genotyped and founder individuals to determine the best way to set up our simulation.

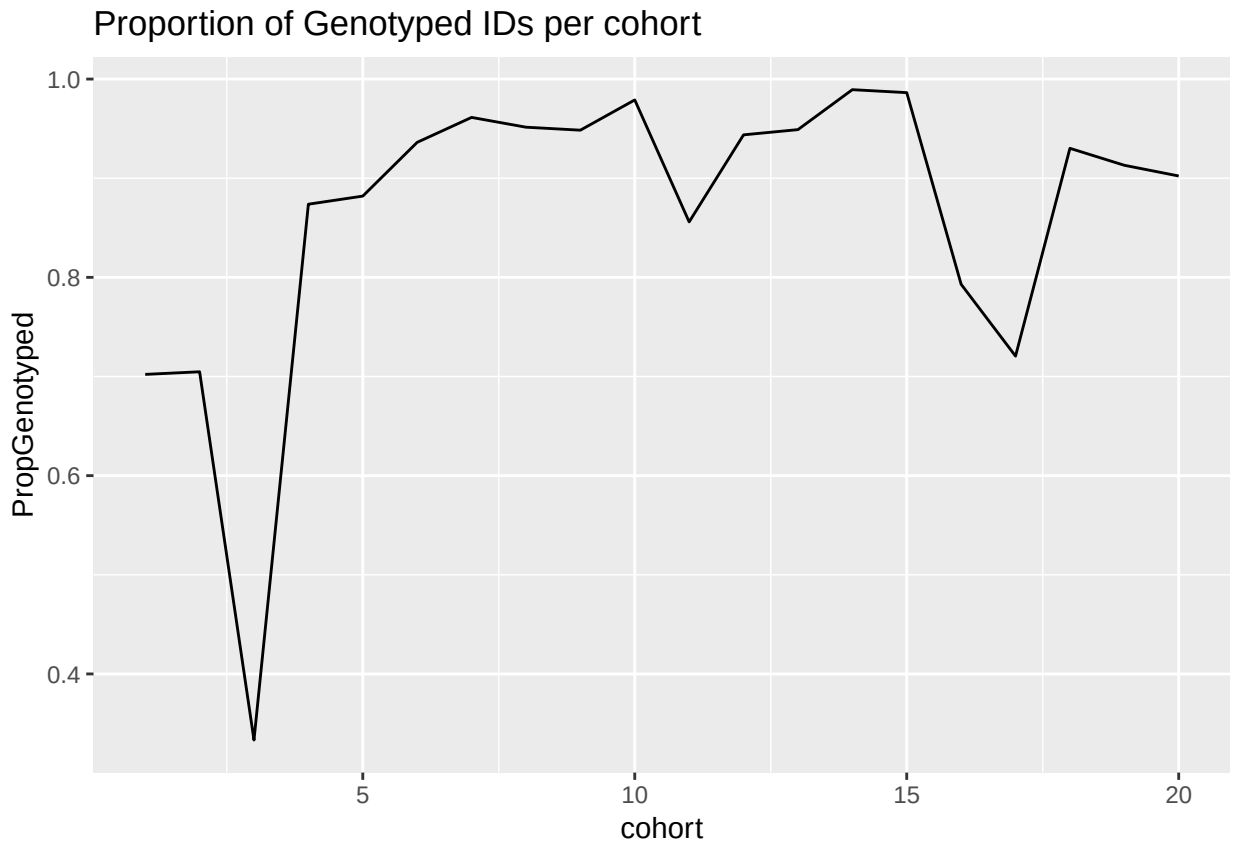
```
ggplot(unicorn.summ, aes(cohort, PropFounders)) +
  geom_line() +
  ggtitle("Proportion of Founder IDs per cohort")
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```




```
ggplot(unicorn.summ, aes(cohort, PropGenotyped)) +
  geom_line()  +
  ggtitle("Proportion of Genotyped IDs per cohort")
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```



The number of cohorts declines rapidly until around 5 cohorts, so lets define cohorts 1:5 as founder cohorts, and 6:20 as simulated cohorts i.e. we will investigate how allele frequencies change in the second 15 year period from year 6 to year 20.

As *Horns* is a biallelic locus, we can model the gene-drop simulation using the function `genedrop_snp()`. We define the `id`, `mother`, `father`, `cohort` and `genotype` using columns from the `unicorn` dataset. We then define the following:

- `nsim` - The number of simulations to run
- `n_founder_cohorts` - The number of founder cohorts - here we will choose 5.
- `fix_founders` - This is an option to keep the founder genotypes fixed in genotyped individuals, i.e. how allele frequencies would change from an almost identical starting point. In this analysis, we will keep this as F to allow some flexibility.
- `verbose` - This will output the function progress.
- `interval` - If verbose is TRUE, then this defines the interval at which to output the function progress.

Let's try it:

```
unicorn.UF <- genedrop_snp(id = unicorn$id,
                           mother = unicorn$mother,
                           father = unicorn$father,
                           cohort = unicorn$cohort,
                           genotype = unicorn$HornSNP,
                           nsim = 1000,
                           n_founder_cohorts = 5,
                           fix_founders = F,
                           verbose = T,
                           interval = 200)
```

```
## Running simulation 1 of 1000.
## Running simulation 201 of 1000.
## Running simulation 401 of 1000.
## Running simulation 601 of 1000.
## Running simulation 801 of 1000.
```

```
str(unicorn.UF)
```

```
## 'data.frame':    3951000 obs. of  7 variables:
##  $ ID            : chr  "1" "2" "3" "4" ...
##  $ True.Geno     : num  NA 2 NA 1 1 2 0 NA 1 0 ...
##  $ cohort        : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Mum.Allele    : num  1 1 1 0 0 1 0 0 0 0 ...
##  $ Dad.Allele    : num  0 1 0 1 1 1 0 1 1 0 ...
##  $ Simulation    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Simulated.Geno: num  1 2 1 1 1 2 0 1 1 0 ...
```

This outputs a data frame that includes the simulated genotypes for each individual. For computational reasons, the genotypes have been reported as allele dosages, e.g. 0 = AA, 1 = AG, 2 = GG.

This is a very large dataset, but we can summarise the allele frequencies per cohort with the function summary_genedrop:

```
unicorn.UF.summ <- summary_genedrop(unicorn.UF)
str(unicorn.UF.summ)
```

```
## List of 2
##  $ observed_frequencies :'data.frame':   20 obs. of  4 variables:
##   ..$ Cohort    : num [1:20] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ Simulation: num [1:20] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ Count     : int [1:20(1d)] 33 74 33 90 127 176 174 196 184 186 ...
##   .. ..- attr(*, "dimnames")=List of 1
##   .. .. ..$ : chr [1:20] "1" "2" "3" "4" ...
##   ..$ p         : num [1:20(1d)] 0.409 0.392 0.394 0.461 0.382 ...
##   .. ..- attr(*, "dimnames")=List of 1
##   .. .. ..$ : chr [1:20] "1" "2" "3" "4" ...
##  $ simulated_frequencies:'data.frame':   20000 obs. of  4 variables:
##   ..$ Cohort    : int [1:20000] 1 2 3 4 5 6 7 8 9 10 ...
##   ..$ Simulation: int [1:20000] 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ Count     : int [1:20000] 33 74 33 90 127 176 174 196 184 186 ...
##   ..$ p         : num [1:20000] 0.409 0.385 0.394 0.467 0.441 ...
```

This object contains two lists: the observed frequencies at *Horn* and the simulated frequencies.

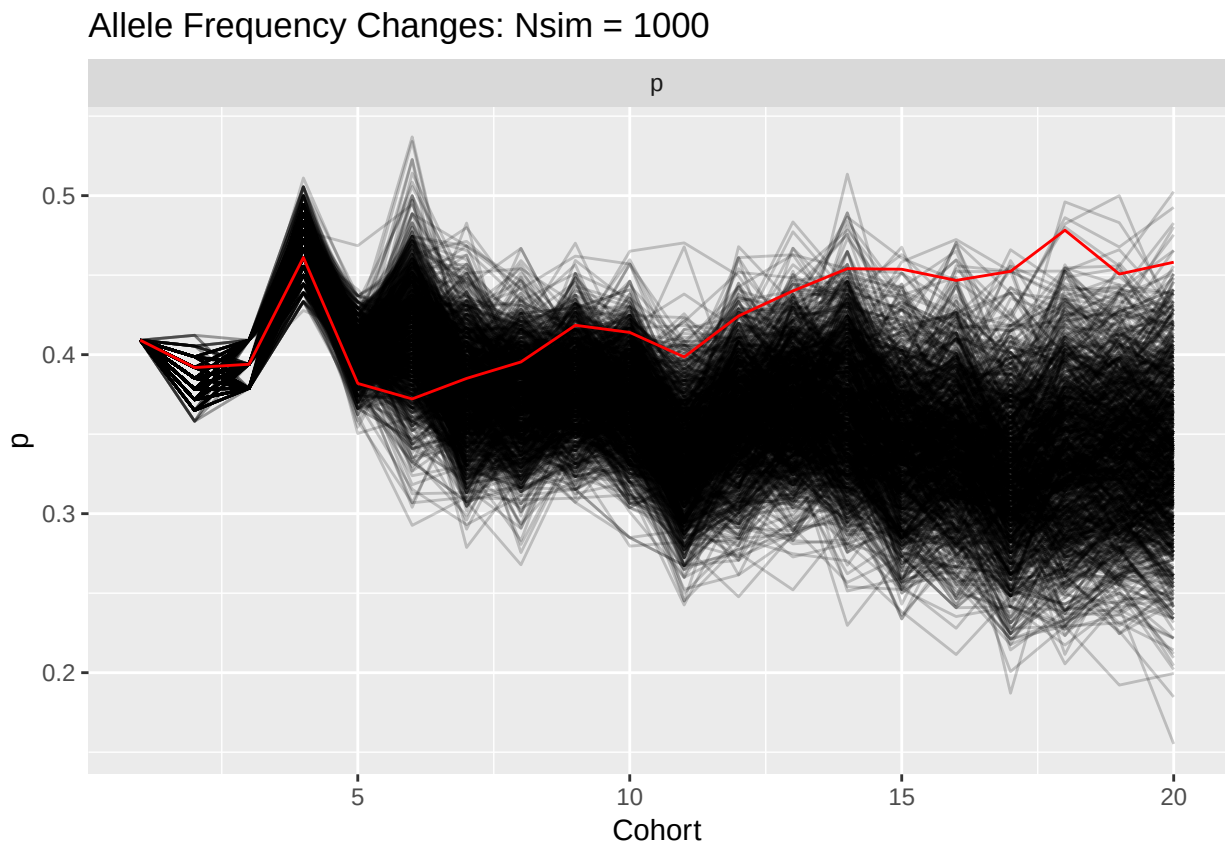
```
unicorn.UF.summ$observed_frequencies
```

```
##    Cohort Simulation Count         p
## 1       1          0    33 0.4090909
## 2       2          0    74 0.3918919
## 3       3          0    33 0.3939394
## 4       4          0    90 0.4611111
## 5       5          0   127 0.3818898
## 6       6          0   176 0.3721591
## 7       7          0   174 0.3850575
## 8       8          0   196 0.3954082
## 9       9          0   184 0.4184783
```

```
## 10      10           0   186 0.4139785
## 11      11           0   202 0.3985149
## 12      12           0   218 0.4243119
## 13      13           0   242 0.4400826
## 14      14           0   185 0.4540541
## 15      15           0   216 0.4537037
## 16      16           0   272 0.4466912
## 17      17           0   147 0.4523810
## 18      18           0   253 0.4782609
## 19      19           0   294 0.4506803
## 20      20           0   203 0.4581281
```

To visualise how the observed and simulated frequencies compare, we can use the `plot_genedrop_results()` function:

```
plot_genedrop_results(unicorn.UF.summ)
```

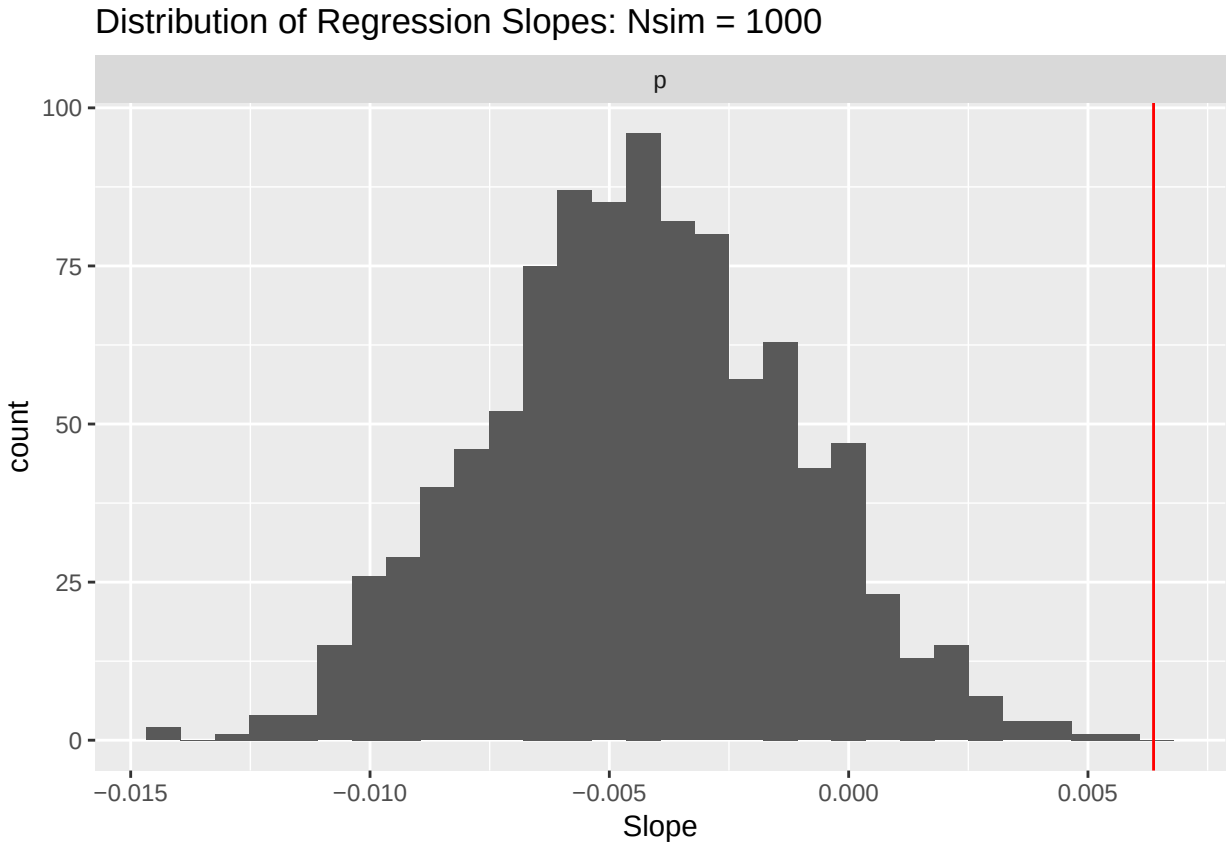


We can see that the founder cohorts (1:5) were sampled based on the observed frequencies, but this constraint is then removed after 5 generations. The red line shows the observed allele frequency change.

We can then use the function `plot_genedrop_lm_slopes()` to look at how the linear regression slopes of the allele frequencies compare to the observed slope (red vertical line). This function also outputs the number of simulated slopes that were higher or lower than the observed slope.

```
plot_genedrop_lm_slopes(unicorn.UF.summ,
                        n_founder_cohorts = 5,
                        remove_founders = T)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



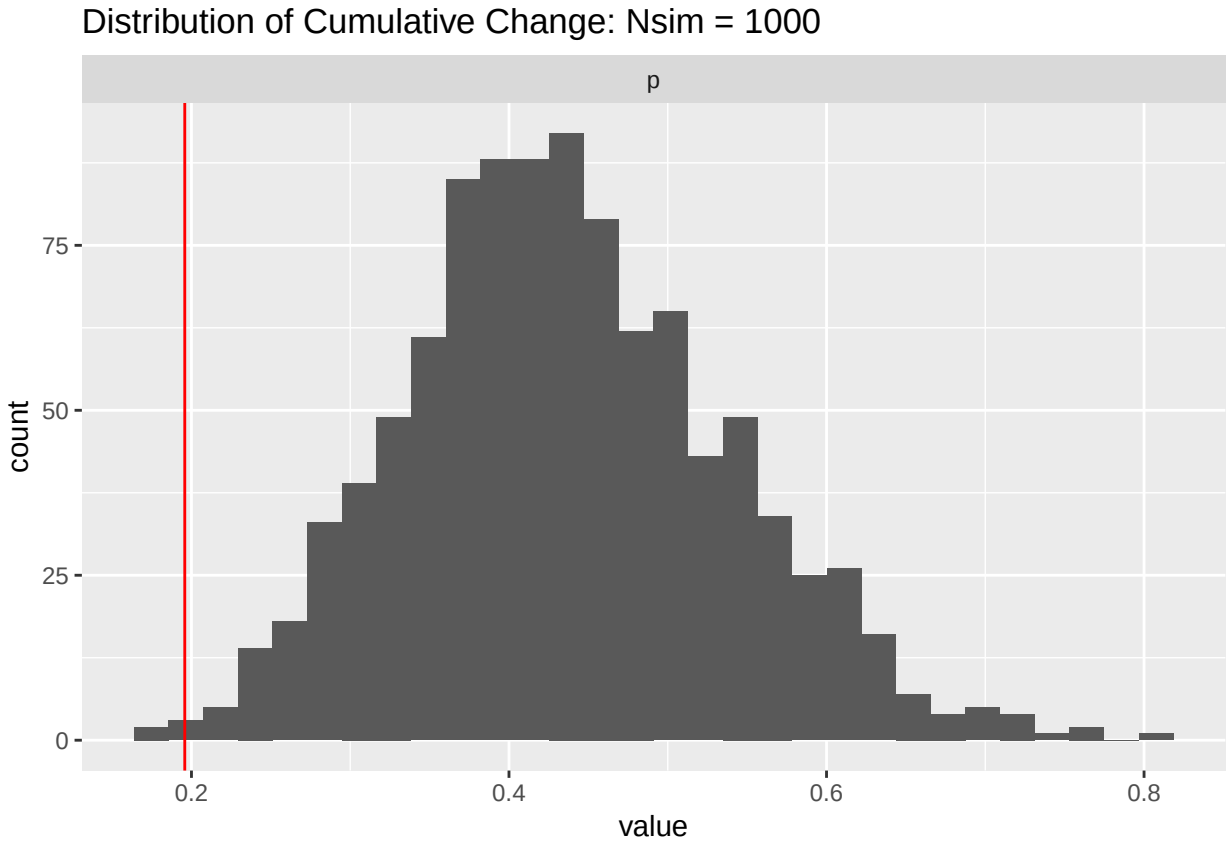

```
##   Iteration Allele       Slope Slopes.Lower Slopes.Higher
## 1         0      p 0.006370185         1000             0
```

There is some suggestion that the increase in frequency of the A allele could be due to selection in this population.

One apparent shortcoming of this approach is that we can only detect strong, directional selection. This is a valid criticism. However, one could potentially identify signature of balancing selection at a locus if there is less cumulative change in allele frequency over time than expected due to change (NB. At this stage this is just an idea - any feedback on this approach is welcome). The function `plot_genedrop_cumulative_change()` plots and reports the cumulated change over the non-founder cohorts:

```
plot_genedrop_cumulative_change(unicorn.UF.summ,
                                n_founder_cohorts = 5,
                                remove_founders = T)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##   Var1 Var2     value Cumulative.Change.Lower Cumulative.Change.Higher
## 1    0    p 0.1957828                       3                      997
```

This also suggests that the observed allele frequency change from year to year is less than expected by chance.

Example 2: The Magic Histocompatibility Locus (MHC)

This locus characterises haplotype variation at a moderately variable locus controlling magic compatibility within unicorns.

```
table(unicorn$MHC)
```

```
##
##  AA  AB  AC  AD  BA  BB  BC  BD  CA  CB  CC  CD  DA  DB  DC  DD
##  22  40 104 121  51  98 203 246 114 161 350 406 129 203 399 504
```

As you can see, there are 4 alleles at this locus. Because there are more than two alleles, we can conduct gene-dropping simulations with the function `genedrop_multi()`. *NB. If you are running this on genotypes with a delimiter e.g. "A/C" or "Lo_Hi", you will need to define it using `genotype_delim = "/"` or `genotype_delim = "_"` within the function.*

```
unicorn.summ <- summary_cohort(id = unicorn$id,
                               mother = unicorn$mother,
                               father = unicorn$father,
                               cohort = unicorn$cohort,
                               genotype = unicorn$MHC)

unicorn.summ
```

```
##    cohort          A         B         C         D GenoCount FullCount
## 1       1 0.04054054 0.2837838 0.3243243 0.3513514        37        47
## 2       2 0.12837838 0.2094595 0.2972973 0.3648649        74       105
## 3       3 0.10526316 0.2236842 0.3026316 0.3684211        76        99
## 4       4 0.08536585 0.1768293 0.3658537 0.3719512        82       103
## 5       5 0.09734513 0.2035398 0.2964602 0.4026549       113       144
## 6       6 0.09935897 0.2019231 0.3044872 0.3942308       156       188
## 7       7 0.09507042 0.2007042 0.3133803 0.3908451       142       181
## 8       8 0.09567901 0.2067901 0.2993827 0.3981481       162       206
## 9       9 0.08450704 0.1760563 0.3274648 0.4119718       142       194
## 10     10 0.09235669 0.1783439 0.3375796 0.3917197       157       190
## 11     11 0.07853403 0.1570681 0.3691099 0.3952880       191       236
## 12     12 0.08196721 0.1912568 0.2814208 0.4453552       183       231
## 13     13 0.08080808 0.1717172 0.3409091 0.4065657       198       255
## 14     14 0.08389262 0.1677852 0.3557047 0.3926174       149       187
## 15     15 0.11578947 0.1710526 0.3289474 0.3842105       190       219
## 16     16 0.09821429 0.1946429 0.2982143 0.4089286       280       343
## 17     17 0.09177215 0.1518987 0.3354430 0.4208861       158       204
## 18     18 0.11659193 0.1345291 0.3834081 0.3654709       223       272
## 19     19 0.11264822 0.1343874 0.3636364 0.3893281       253       322
## 20     20 0.08918919 0.1297297 0.3459459 0.4351351       185       225
## 21     NA        NaN       NaN       NaN       NaN         0         0
##    PropGenotyped NonFounders Founders PropFounders
## 1      0.7872340           0       47   1.00000000
## 2      0.7047619          28       77   0.73333333
## 3      0.7676768          70       29   0.29292929
## 4      0.7961165          52       51   0.49514563
## 5      0.7847222          99       45   0.31250000
## 6      0.8297872         139       49   0.26063830
## 7      0.7845304         151       30   0.16574586
## 8      0.7864078         188       18   0.08737864
## 9      0.7319588         187        7   0.03608247
## 10     0.8263158         176       14   0.07368421
## 11     0.8093220         224       12   0.05084746
## 12     0.7922078         220       11   0.04761905
## 13     0.7764706         244       11   0.04313725
## 14     0.7967914         167       20   0.10695187
## 15     0.8675799         202       17   0.07762557
## 16     0.8163265         306       37   0.10787172
## 17     0.7745098         187       17   0.08333333
## 18     0.8198529         252       20   0.07352941
## 19     0.7857143         316        6   0.01863354
## 20     0.8222222         214       11   0.04888889
## 21           NaN          NA       NA           NA
```

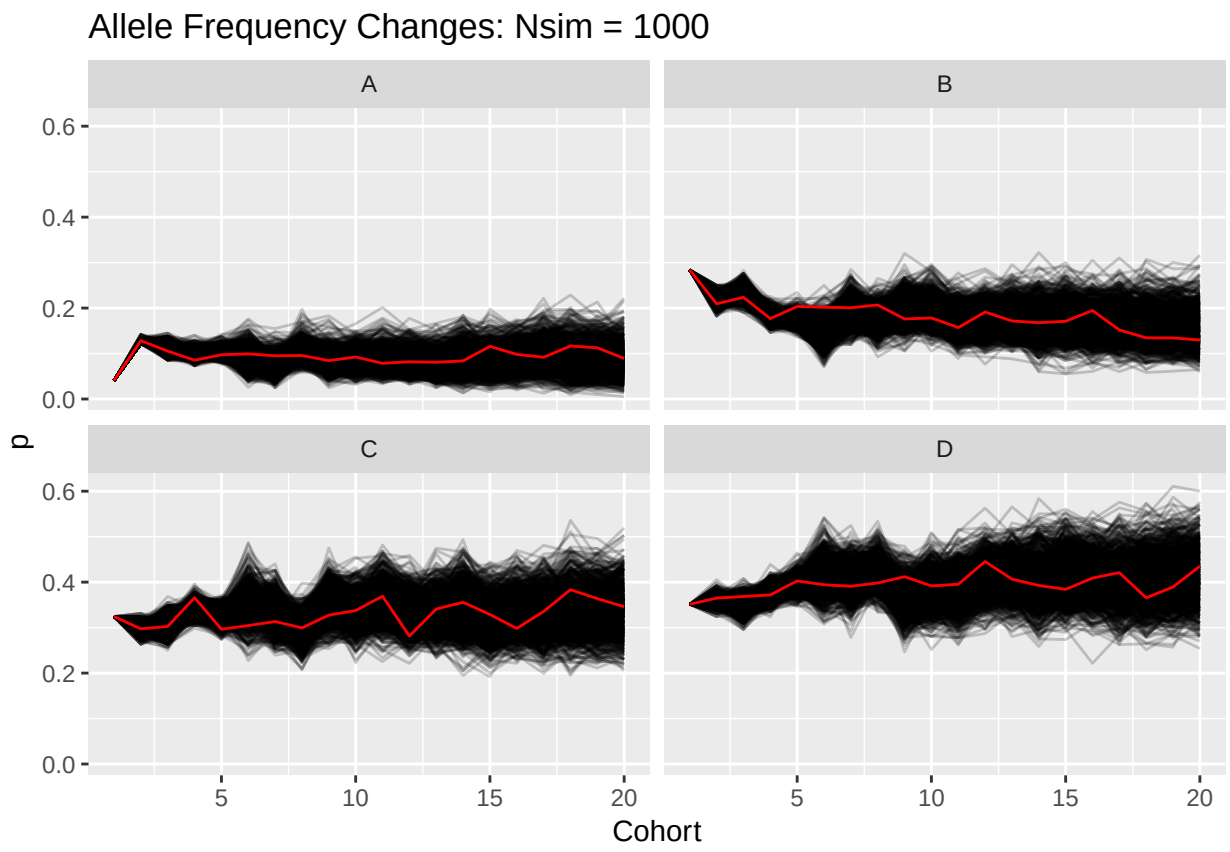
```r
unicorn.MHC.UF <- genedrop_multi(id = unicorn$id,
                                 mother = unicorn$mother,
                                 father = unicorn$father,
                                 cohort = unicorn$cohort,
                                 genotype = unicorn$MHC,
                                 nsim = 1000,
                                 n_founder_cohorts = 5,
                                 fix_founders = F,
                                 verbose = T,
```

```
                                    interval = 200)
```

```
## Running simulation 1 of 1000.
## Running simulation 201 of 1000.
## Running simulation 401 of 1000.
## Running simulation 601 of 1000.
## Running simulation 801 of 1000.
```

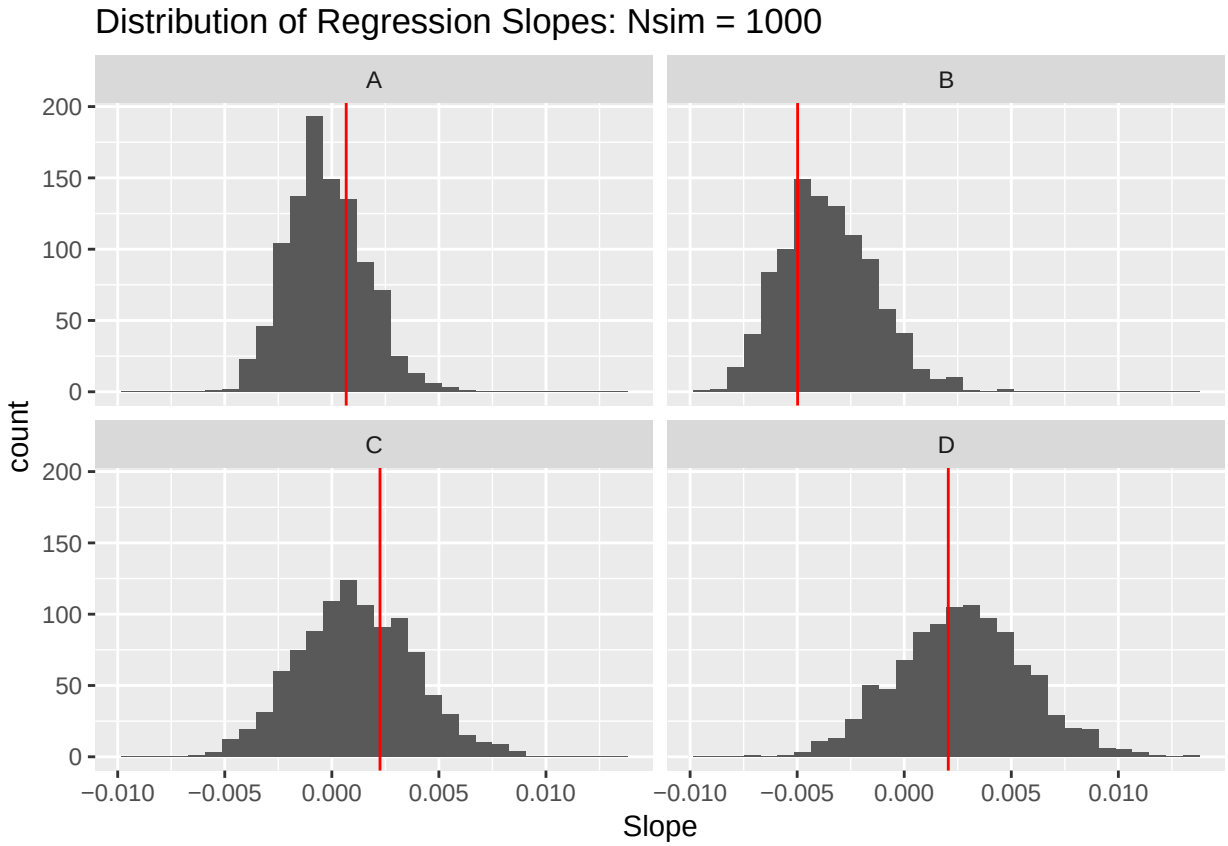
```
unicorn.MHC.UF.summ <- summary_genedrop(unicorn.MHC.UF)

plot_genedrop_results(unicorn.MHC.UF.summ)
```




```
plot_genedrop_lm_slopes(unicorn.MHC.UF.summ)
```

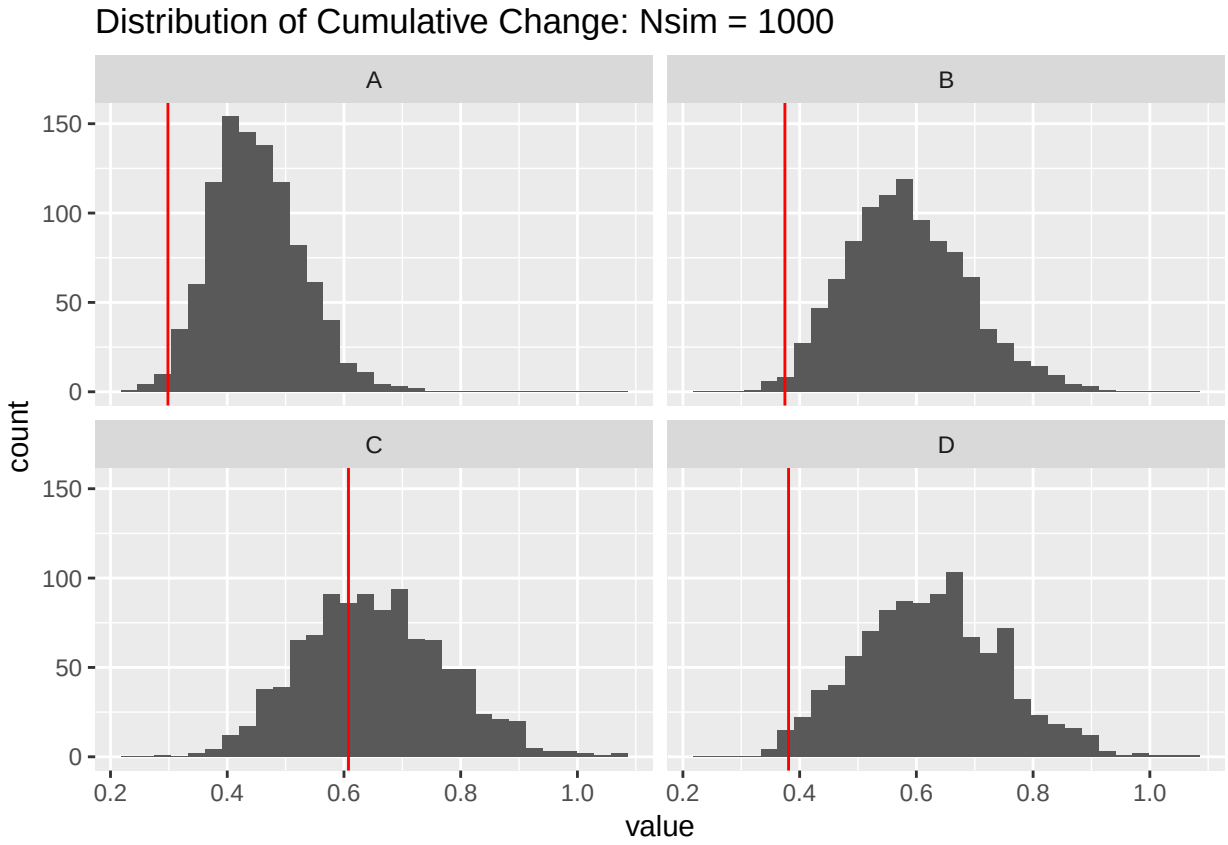
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##   Iteration Allele         Slope Slopes.Lower Slopes.Higher
## 1         0      A  0.0006718187          708           292
## 2         0      B -0.0049788334          267           733
## 3         0      C  0.0022493400          660           340
## 4         0      D  0.0020576748          409           591
```

```
plot_genedrop_cumulative_change(unicorn.MHC.UF.summ)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##   Var1 Var2     value Cumulative.Change.Lower Cumulative.Change.Higher
## 1    0    A 0.2983984                      11                       989
## 2    0    B 0.3747645                      10                       990
## 3    0    C 0.6077412                     383                       617
## 4    0    D 0.3810273                      11                       989
```

Example 3: The Colour Locus

```
unicorn.summ <- summary_cohort(id = unicorn$id,
                               mother = unicorn$mother,
                               father = unicorn$father,
                               cohort = unicorn$cohort,
                               genotype = unicorn$ColourSNP)

unicorn.summ
```

```
##    cohort          A         B GenoCount FullCount PropGenotyped
## 1       1 0.01388889 0.9861111        36        47     0.7659574
## 2       2 0.03623188 0.9637681        69       105     0.6571429
## 3       3 0.05421687 0.9457831        83        99     0.8383838
## 4       4 0.01315789 0.9868421        76       103     0.7378641
## 5       5 0.02020202 0.9797980        99       144     0.6875000
## 6       6 0.04609929 0.9539007       141       188     0.7500000
## 7       7 0.04850746 0.9514925       134       181     0.7403315
## 8       8 0.03548387 0.9645161       155       206     0.7524272
```

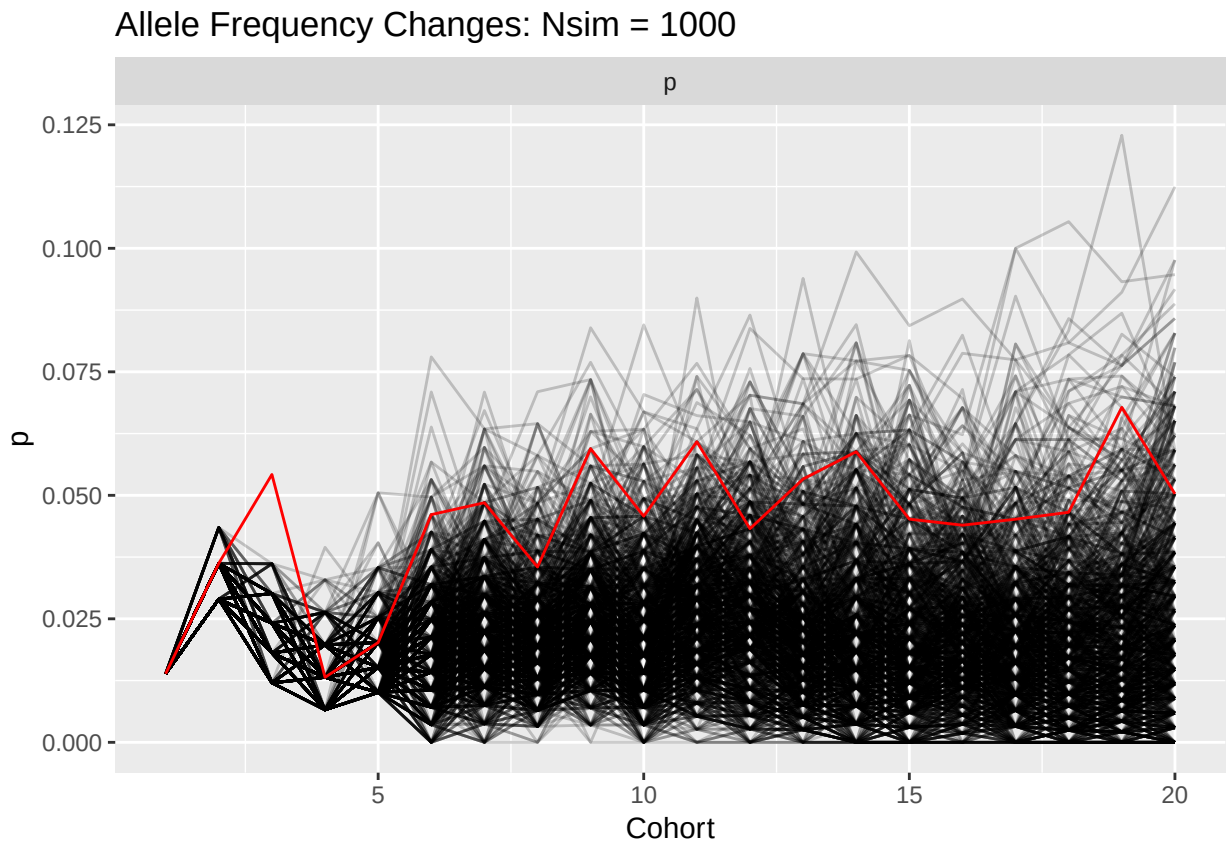
```
## 9       9 0.05944056 0.9405594       143       194     0.7371134
## 10     10 0.04577465 0.9542254       142       190     0.7473684
## 11     11 0.06084656 0.9391534       189       236     0.8008475
## 12     12 0.04324324 0.9567568       185       231     0.8008658
## 13     13 0.05329949 0.9467005       197       255     0.7725490
## 14     14 0.05882353 0.9411765       136       187     0.7272727
## 15     15 0.04518072 0.9548193       166       219     0.7579909
## 16     16 0.04395604 0.9560440       273       343     0.7959184
## 17     17 0.04516129 0.9548387       155       204     0.7598039
## 18     18 0.04656863 0.9534314       204       272     0.7500000
## 19     19 0.06779661 0.9322034       236       322     0.7329193
## 20     20 0.05029586 0.9497041       169       225     0.7511111
## 21     NA        NaN       NaN         0         0           NaN
##    NonFounders Founders PropFounders
## 1            0       47   1.00000000
## 2           28       77   0.73333333
## 3           70       29   0.29292929
## 4           52       51   0.49514563
## 5           99       45   0.31250000
## 6          139       49   0.26063830
## 7          151       30   0.16574586
## 8          188       18   0.08737864
## 9          187        7   0.03608247
## 10         176       14   0.07368421
## 11         224       12   0.05084746
## 12         220       11   0.04761905
## 13         244       11   0.04313725
## 14         167       20   0.10695187
## 15         202       17   0.07762557
## 16         306       37   0.10787172
## 17         187       17   0.08333333
## 18         252       20   0.07352941
## 19         316        6   0.01863354
## 20         214       11   0.04888889
## 21          NA       NA           NA
```

```r
unicorn.colour.UF <- genedrop_snp(id = unicorn$id,
                                  mother = unicorn$mother,
                                  father = unicorn$father,
                                  cohort = unicorn$cohort,
                                  genotype = unicorn$ColourSNP,
                                  nsim = 1000,
                                  n_founder_cohorts = 5,
                                  fix_founders = F,
                                  verbose = T,
                                  interval = 200)
```

```
## Running simulation 1 of 1000.
## Running simulation 201 of 1000.
## Running simulation 401 of 1000.
## Running simulation 601 of 1000.
## Running simulation 801 of 1000.
```

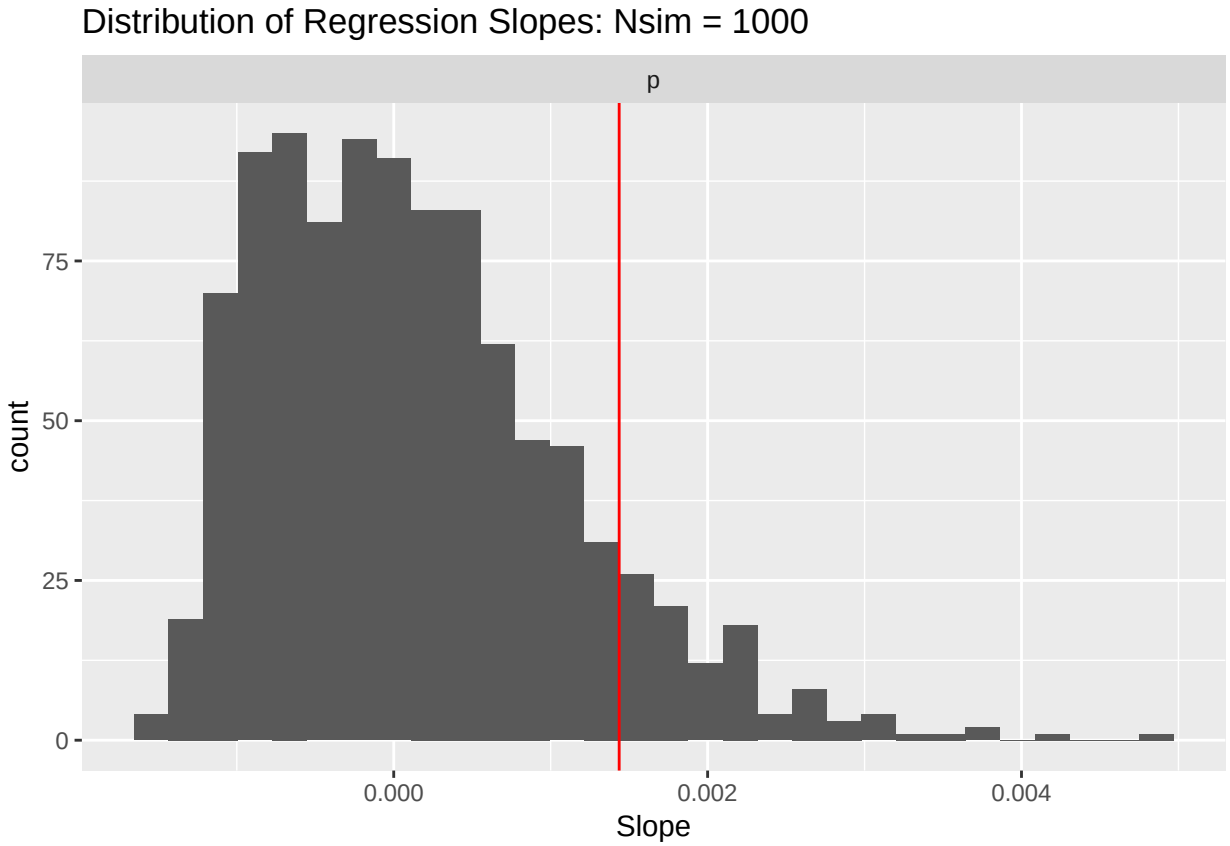
```
unicorn.colour.UF.summ <- summary_genedrop(unicorn.colour.UF)

plot_genedrop_results(unicorn.colour.UF.summ)
```




```
plot_genedrop_lm_slopes(unicorn.colour.UF.summ)
```

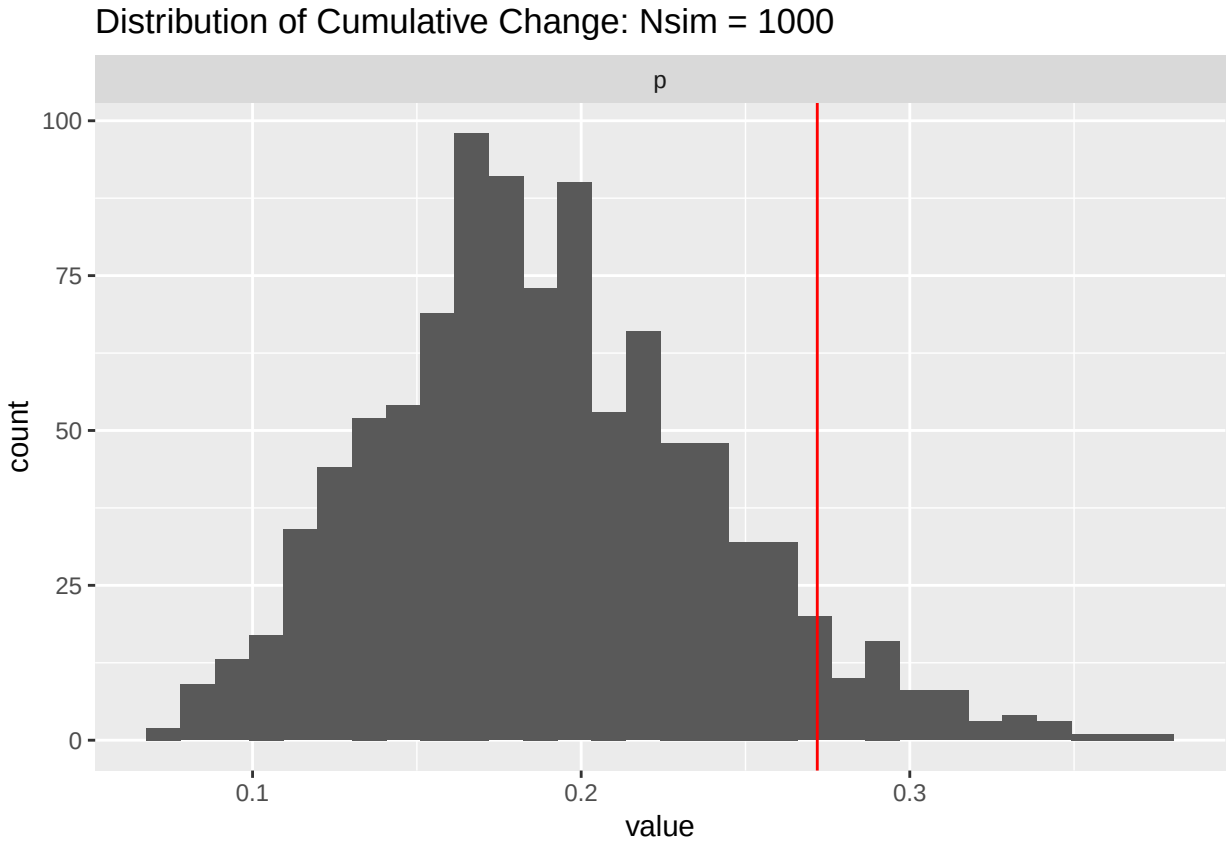
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##   Iteration Allele       Slope Slopes.Lower Slopes.Higher
## 1         0      p 0.001436429          898           102
```

```
plot_genedrop_cumulative_change(unicorn.colour.UF.summ)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
##   Var1 Var2    value Cumulative.Change.Lower Cumulative.Change.Higher
## 1    0    p 0.271847                     934                        66
```