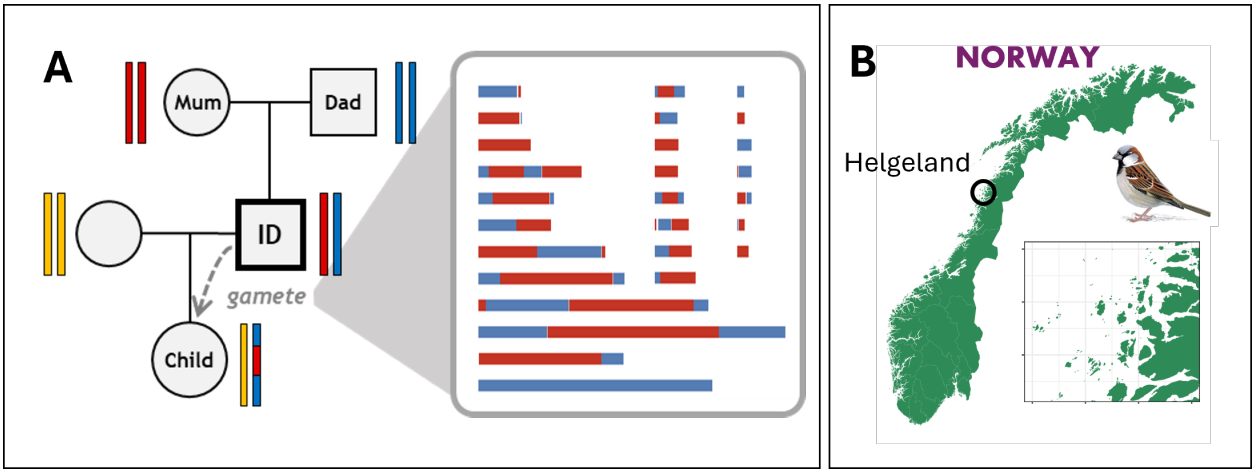
Genomic Prediction of Sparrow Recombination

Susan Johnston

2024-09-10

1. Introduction - recombination in house sparrows.



The positions of meiotic crossover events are measured by tracking the inheritance patterns of alleles between parents and offspring and determining the positions of changes in phase. Our dataset `recsumm` has estimates of crossover count (`co_count`) and intra-chromosomal allelic shuffling (`intra-shuff`) for 13042 `offspring` resulting from gametes transmitted from 2652 unique `id`.It also includes the `total_coverage` which is the length of the genome informative for identifying crossover events.

```
head(recsumm)
```

```
##              meiosis      id  offspring sex total_coverage co_count intra_shuff
## 1 8118424_8L64240_A3 8118424 8L64240_A3   M      901765568       17  0.03330310
## 2    8118424_8L64241 8118424    8L64241   M      903819159       11  0.02020535
## 3 8118424_8L64242_C3 8118424 8L64242_C3   M      902947627       16  0.02541074
## 4    8118424_8L89644 8118424    8L89644   M      903580120       17  0.02776473
## 5    8118424_8L89675 8118424    8L89675   M      903886149       16  0.02607208
## 6    8118424_8L89683 8118424    8L89683   M      903886149       16  0.02943656
```

There is also the dataset `recmeans` that summarises this information per individual:

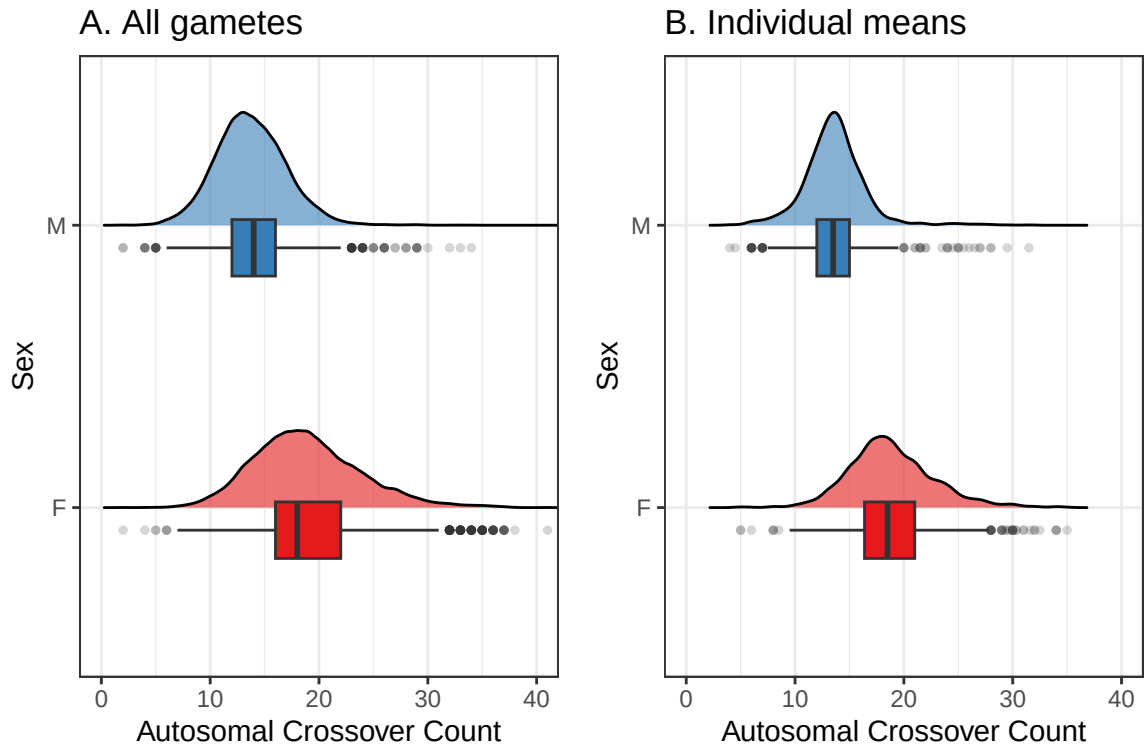
```
head(recmeans)
```

```
## # A tibble: 6 x 9
## # Groups:   id [6]
##   id      sex   mean_co_count mean_intra_shuff mean_total_coverage var_co_count
##   <fct>   <chr>         <dbl>            <dbl>               <dbl>        <dbl>
## 1 8118424 M              14.0           0.0217          902725968.         9.57
## 2 8118425 F              17.8           0.0224          888106074.        21.6
## 3 8118432 F              18.3           0.0300          896633120.        42.3
## 4 8118433 M              12.4           0.0142          890301349.         9.28
```

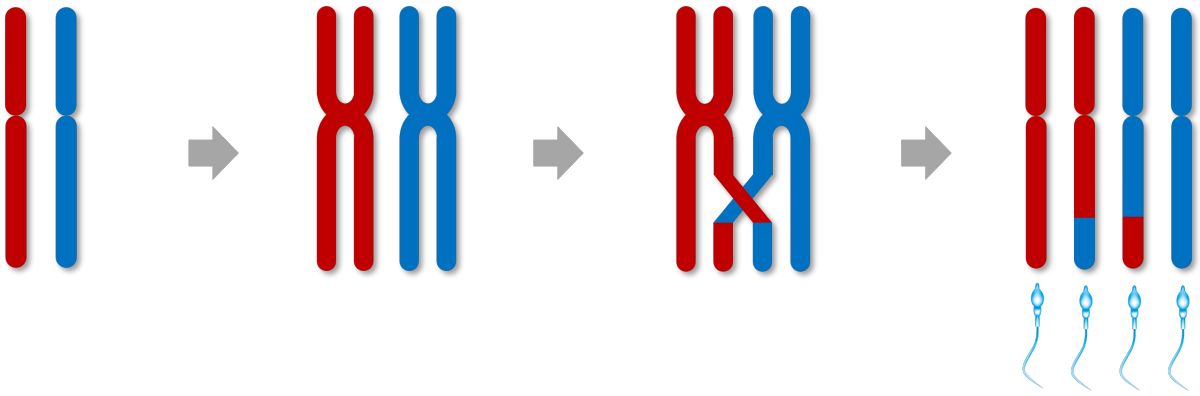
```
## 5 8118435 F              18.8          0.0285          900122488          7.72
## 6 8168601 M              13.5          0.0169          896135125.         4.67
## # i 3 more variables: var_intra_shuff <dbl>, var_total_coverage <dbl>, n <int>
```

In this document, I will solely focus on crossover count.

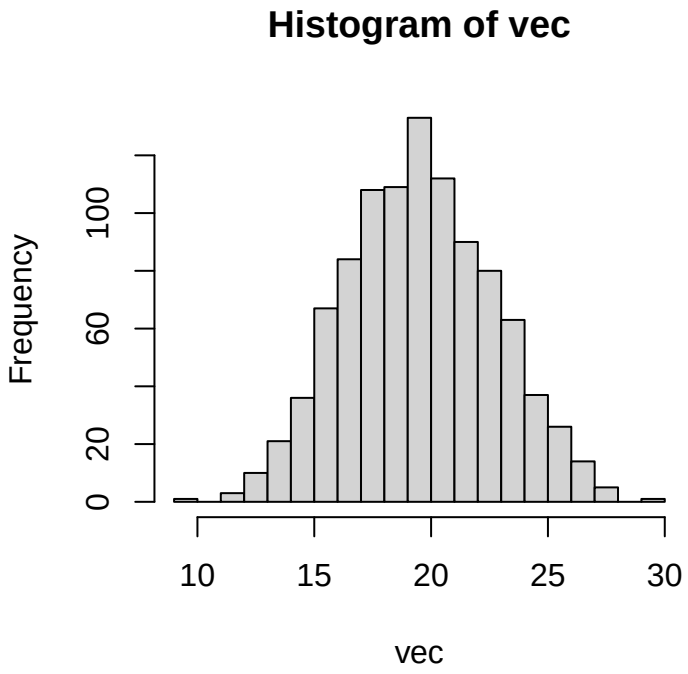
This is the distribution of crossover counts across all gametes (A) vs an individual's mean crossover count (B). As you can see, there is a clear sex difference in the trait distribution:



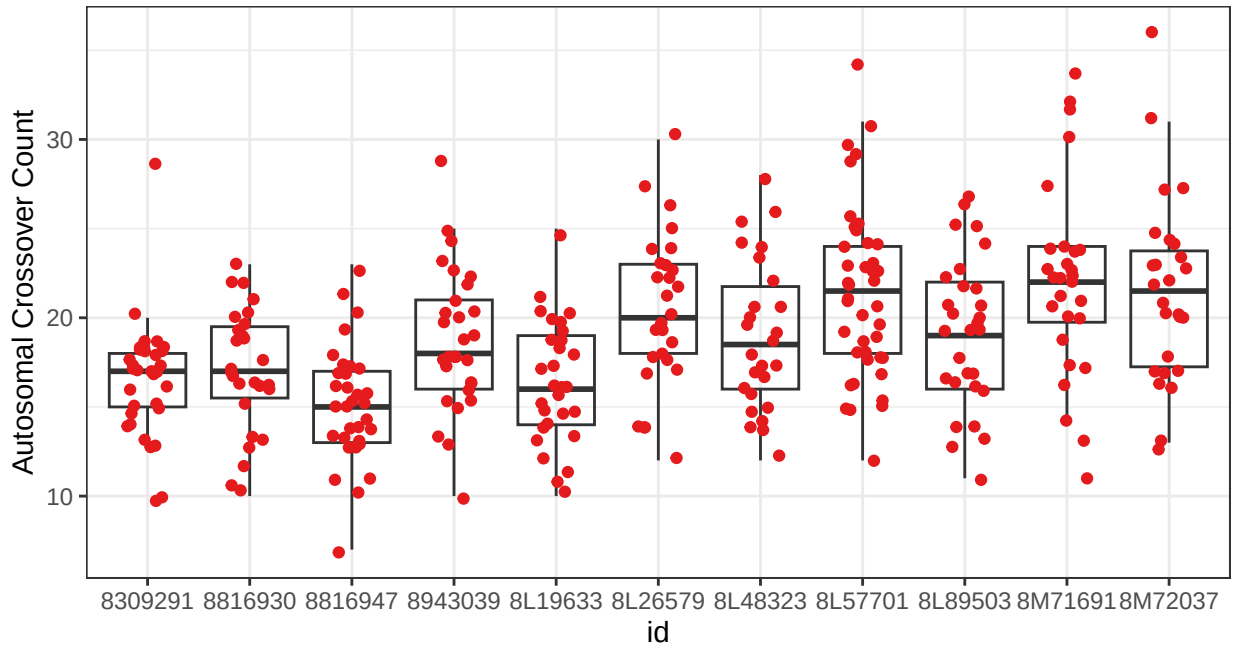
2. Crossover count has a high sampling variance, which leads to weird quirks.



Crossover count is an unusual phenotype as there is more intra-individual variation than inter-individual variation. This is because for each crossover event that happens during meiosis, we can assume that is ~50:50 chance that it segregates into the egg or sperm. If you have 40 crossovers, an average of 20 will end up in the gamete. But that still means there is decent variation around that mean. For example, for 1000 gametes coming from an individual with 40 crossovers in every meiosis, the distribution of crossovers in the gametes would be:



As I stated above, this translates into more intra-individual variation than inter-individual variation. Let's look at the female sparrows with the most unique measures:

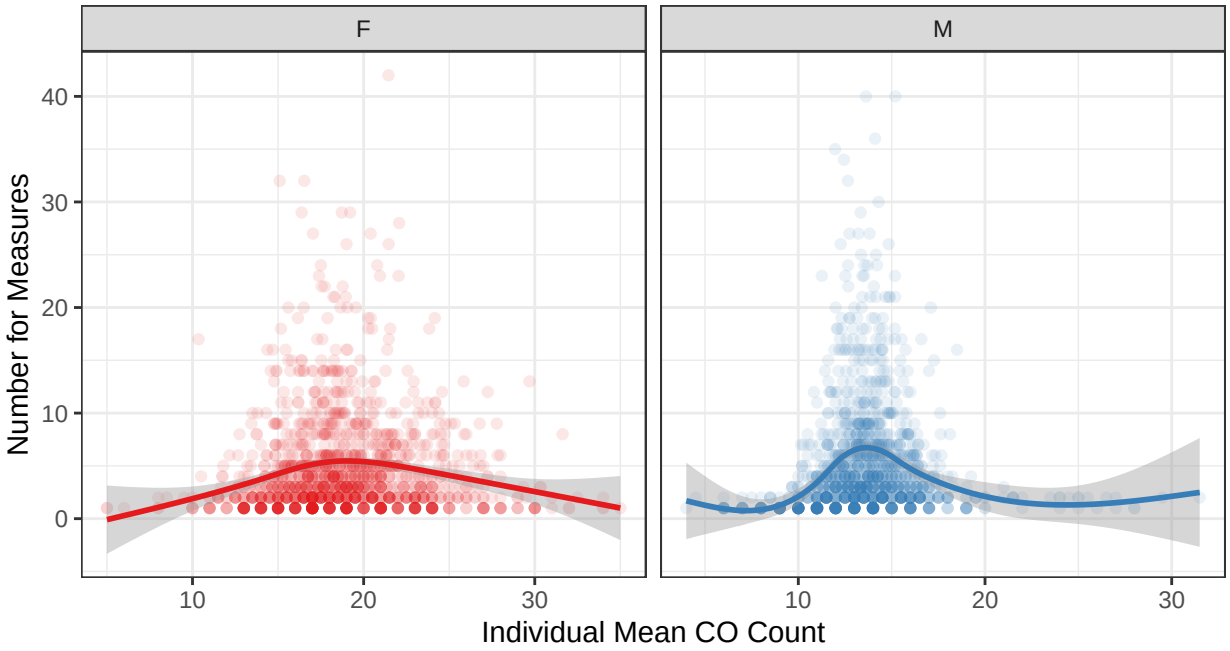


The variance across individual means is 5.5438493, whereas the variance within individuals is:

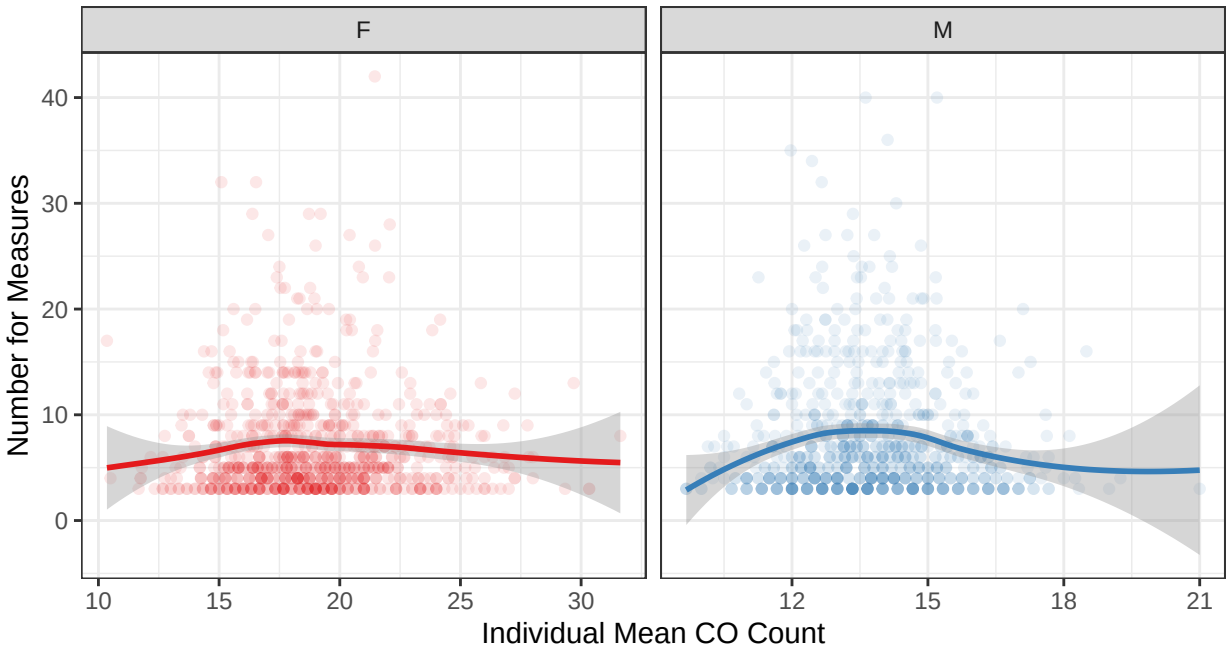
```
## # A tibble: 11 x 2
##    id      `var(co_count)`
##    <fct>             <dbl>
```

```
##   1 8309291            11.1
##   2 8816930            12.0
##   3 8816947            10.4
##   4 8943039            16.2
##   5 8L19633            12.0
##   6 8L26579            18.6
##   7 8L48323            17.7
##   8 8L57701            23.1
##   9 8L89503            17.5
##  10 8M71691            30.5
##  11 8M72037            27.3
```

We also observe that individuals with more measures will have means close to the population mean, which has implications for further analysis of fitness!



In females, once you have more than 3 measures, this effect starts to flatten out. In males it doesn't, perhaps they are more sensitive to this problem because they have fewer crossovers. . . ?



2. How heritable is crossover rate?

Because there are differences in distribution between the sexes, let's just focus on **females** for the moment. Heritabilities are estimated using a pedigree-based relationship matrix (this gives very similar results to the GRM used in the McAuley et al 2024 MBE paper). I use `library(asreml)` for the following models.

Below I consider two phenotypes:

1. What is the heritability of the number of crossovers in a sampled gamete?
2. What is the heritability of the mean number of crossovers across all gametes?

Let's begin. Here, the heritability is estimated using the crossovers in all sampled gametes (value for `vm(id, ainv)`):

```
recsumm_f <- subset(recsumm, sex == "F")

female.acc.ped <- asreml(fixed = co_count ~ total_coverage,
                         random = ~ vm(id, ainv) + ide(id),
                         residual = ~idv(units),
                         data = recsumm_f,
                         trace = F)

asreml4pin(female.acc.ped)
```

```
##      Estimate         SE       Effect
## h1 0.25948288 0.03196861 vm(id, ainv)
## h2 0.03335797 0.02544646      ide(id)
## h3 0.70715915 0.01633110  units!units
```

And here is the mean rate per individual (value for `vm(id, ainv)`):

```
recmeans_f <- subset(recmeans, sex == "F")

female.acc.ped.mean <- asreml(fixed = mean_co_count ~ mean_total_coverage + n,
                              random = ~ vm(id, ainv),
```

```
                              residual = ~idv(units),
                              data = recmeans_f,
                              trace = F)

asreml4pin(female.acc.ped.mean)
```

```
##     Estimate         SE       Effect
## h1 0.4993435 0.06423255 vm(id, ainv)
## h2 0.5006565 0.06423255  units!units
```

And here is the mean rate per individual with >2 measures (value for `vm(id, ainv)`):

```
##    Estimate         SE       Effect
## h1 0.607963 0.08171855 vm(id, ainv)
## h2 0.392037 0.08171855  units!units
```

As you can see, the heritability basically doubles when using the mean measures!

Here is the same repeated for males...

```
recsumm_m <- subset(recsumm, sex == "M")

male.acc.ped <- asreml(fixed = co_count ~ total_coverage,
                       random = ~ vm(id, ainv) + ide(id),
                       residual = ~idv(units),
                       data = recsumm_m,
                       trace = F)
```

```
## Warning in asreml(fixed = co_count ~ total_coverage, random = ~vm(id, ainv) + :
## Some components changed by more than 1% on the last iteration
```

```
asreml4pin(male.acc.ped)
```

```
##        Estimate         SE       Effect
## h1 0.1854285071 0.02897579 vm(id, ainv)
## h2 0.0008263872 0.02219539      ide(id)
## h3 0.8137451057 0.01609015  units!units
```

```
recmeans_m <- subset(recmeans, sex == "M")

male.acc.ped.mean <- asreml(fixed = mean_co_count ~ mean_total_coverage + n,
                            random = ~ vm(id, ainv),
                            residual = ~idv(units),
                            data = recmeans_m,
                            trace = F)

asreml4pin(male.acc.ped.mean)
```

```
##     Estimate         SE       Effect
## h1 0.4642414 0.07332471 vm(id, ainv)
## h2 0.5357586 0.07332471  units!units
```

```
male.acc.ped.mean <- asreml(fixed = mean_co_count ~ mean_total_coverage + n,
                            random = ~ vm(id, ainv),
                            residual = ~idv(units),
                            data = subset(recmeans_m, n > 2),
                            trace = F)
```

```
asreml4pin(male.acc.ped.mean)
```

```
##     Estimate         SE       Effect
## h1 0.4412126 0.09068989 vm(id, ainv)
## h2 0.5587874 0.09068989  units!units
```

What about the bivariate analysis?

```
recsumm$co_count_m <- ifelse(recsumm$sex == "M", recsumm$co_count, NA)
recsumm$co_count_f <- ifelse(recsumm$sex == "F", recsumm$co_count, NA)


biv.acc.ped <- asreml(fixed = cbind(co_count_f, co_count_m) ~ trait + trait:total_coverage,
                      random = ~ corgh(trait):vm(id, ainv) + idh(trait):ide(id),
                      residual = ~ units:us(trait, init = NA),
                      data = recsumm,
                      workspace = "1000mb")
```

```
## ASReml Version 4.2 01/01/2025 19:54:37
##           LogLik        Sigma2     DF     wall
##  1     -24552.72           1.0  13038   19:54:37
##  2     -24221.51           1.0  13038   19:54:38
##  3     -23928.10           1.0  13038   19:54:38
##  4     -23814.94           1.0  13038   19:54:38
##  5     -23796.51           1.0  13038   19:54:39
##  6     -23796.21           1.0  13038   19:54:39
##  7     -23796.21           1.0  13038   19:54:40
```

```
## Warning in asreml(fixed = cbind(co_count_f, co_count_m) ~ trait +
## trait:total_coverage, : Some components changed by more than 1% on the last
## iteration
```

```
summary(biv.acc.ped)
```

```
## $call
## asreml(fixed = cbind(co_count_f, co_count_m) ~ trait + trait:total_coverage,
##     random = ~corgh(trait):vm(id, ainv) + idh(trait):ide(id),
##     residual = ~units:us(trait, init = NA), data = recsumm, workspace = "1000mb")
##
## $loglik
## [1] -23796.21
##
## $nedf
## [1] 13038
##
## $sigma
## [1] 1
##
## $varcomp
##                                                          component std.error
## trait:vm(id, ainv)!trait!co_count_m:!trait!co_count_f.cor 0.3349214 0.1105707
## trait:vm(id, ainv)!trait_co_count_f                       6.2325348 0.8461876
## trait:vm(id, ainv)!trait_co_count_m                       1.8417211 0.3218359
## trait:ide(id)!trait_co_count_f                            0.8182391 0.6039845
## trait:ide(id)!trait_co_count_m                            0.1257990 0.2387246
## units:trait!R                                             1.0000000        NA
```

```
## units:trait!trait_co_count_f:co_count_f                    17.0218359 0.3334904
## units:trait!trait_co_count_m:co_count_f                     0.0000500        NA
## units:trait!trait_co_count_m:co_count_m                     8.7430510 0.1681271
##                                                               z.ratio bound %ch
## trait:vm(id, ainv)!trait!co_count_m:!trait!co_count_f.cor   3.0290240     U 0.3
## trait:vm(id, ainv)!trait_co_count_f                         7.3654295     P 0.0
## trait:vm(id, ainv)!trait_co_count_m                         5.7225463     P 0.2
## trait:ide(id)!trait_co_count_f                              1.3547352     P 0.1
## trait:ide(id)!trait_co_count_m                              0.5269626     P 4.0
## units:trait!R                                                      NA     F 0.0
## units:trait!trait_co_count_f:co_count_f                    51.0414572     P 0.0
## units:trait!trait_co_count_m:co_count_f                            NA     F 0.0
## units:trait!trait_co_count_m:co_count_m                    52.0026325     P 0.0
##
## $bic
## [1] 47658.75
## attr(,"parameters")
## [1] 7
##
## $aic
## [1] 47606.42
## attr(,"parameters")
## [1] 7
##
## attr(,"class")
## [1] "summary.asreml"
```

```
recmeans$mean_co_count_m <- ifelse(recmeans$sex == "M", recmeans$mean_co_count, NA)
recmeans$mean_co_count_f <- ifelse(recmeans$sex == "F", recmeans$mean_co_count, NA)


biv.acc.ped.mean <- asreml(fixed = cbind(mean_co_count_f, mean_co_count_m) ~ trait + trait:mean_total_c
                           random = ~ corgh(trait):vm(id, ainv),
                           residual = ~ units:us(trait, init = NA),
                           data = recmeans,
                           workspace = "1000mb")
```

```
## ASReml Version 4.2 01/01/2025 19:54:40
##           LogLik        Sigma2     DF     wall
##  1     -4615.822           1.0   2647   19:54:41
##  2     -4543.239           1.0   2647   19:54:41
##  3     -4477.422           1.0   2647   19:54:42
##  4     -4449.783           1.0   2647   19:54:42
##  5     -4443.651           1.0   2647   19:54:43
##  6     -4443.154           1.0   2647   19:54:43
##  7     -4443.090           1.0   2647   19:54:43
##  8     -4443.079           1.0   2647   19:54:44
##  9     -4443.077           1.0   2647   19:54:44
## 10     -4443.077           1.0   2647   19:54:45
```

```
summary(biv.acc.ped.mean)
```

```
## $call
## asreml(fixed = cbind(mean_co_count_f, mean_co_count_m) ~ trait +
##     trait:mean_total_coverage + n, random = ~corgh(trait):vm(id,
```

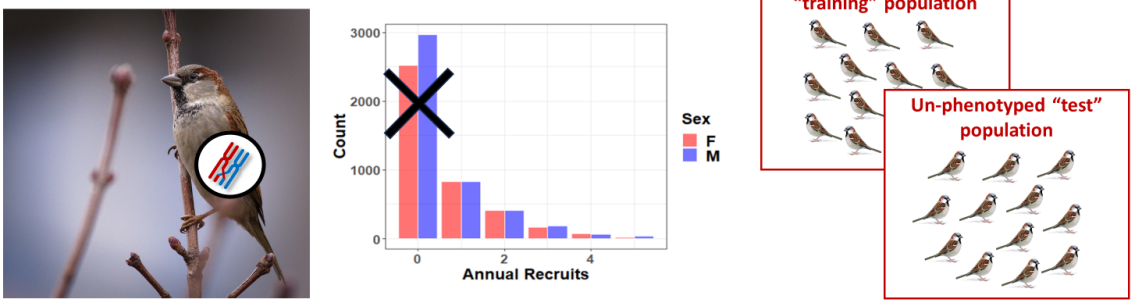
```
##     ainv), residual = ~units:us(trait, init = NA), data = recmeans,
##     workspace = "1000mb")
##
## $loglik
## [1] -4443.077
##
## $nedf
## [1] 2647
##
## $sigma
## [1] 1
##
## $varcomp
##                                                                    component
## trait:vm(id, ainv)!trait!mean_co_count_m:!trait!mean_co_count_f.cor 0.3080475
## trait:vm(id, ainv)!trait_mean_co_count_f                           7.4457906
## trait:vm(id, ainv)!trait_mean_co_count_m                           3.5854462
## units:trait!R                                                      1.0000000
## units:trait!trait_mean_co_count_f:mean_co_count_f                  7.3274502
## units:trait!trait_mean_co_count_m:mean_co_count_f                  0.0000500
## units:trait!trait_mean_co_count_m:mean_co_count_m                  4.3435639
##                                                                    std.error
## trait:vm(id, ainv)!trait!mean_co_count_m:!trait!mean_co_count_f.cor 0.1174680
## trait:vm(id, ainv)!trait_mean_co_count_f                           1.1068130
## trait:vm(id, ainv)!trait_mean_co_count_m                           0.6558918
## units:trait!R                                                             NA
## units:trait!trait_mean_co_count_f:mean_co_count_f                  0.8616160
## units:trait!trait_mean_co_count_m:mean_co_count_f                         NA
## units:trait!trait_mean_co_count_m:mean_co_count_m                  0.5294538
##                                                                     z.ratio
## trait:vm(id, ainv)!trait!mean_co_count_m:!trait!mean_co_count_f.cor 2.622395
## trait:vm(id, ainv)!trait_mean_co_count_f                           6.727235
## trait:vm(id, ainv)!trait_mean_co_count_m                           5.466521
## units:trait!R                                                            NA
## units:trait!trait_mean_co_count_f:mean_co_count_f                  8.504310
## units:trait!trait_mean_co_count_m:mean_co_count_f                        NA
## units:trait!trait_mean_co_count_m:mean_co_count_m                  8.203858
##                                                                    bound %ch
## trait:vm(id, ainv)!trait!mean_co_count_m:!trait!mean_co_count_f.cor     U 0.0
## trait:vm(id, ainv)!trait_mean_co_count_f                                P 0.0
## trait:vm(id, ainv)!trait_mean_co_count_m                                P 0.2
## units:trait!R                                                           F 0.0
## units:trait!trait_mean_co_count_f:mean_co_count_f                       P 0.0
## units:trait!trait_mean_co_count_m:mean_co_count_f                       F 0.0
## units:trait!trait_mean_co_count_m:mean_co_count_m                       P 0.1
##
## $bic
## [1] 8925.56
## attr(,"parameters")
## [1] 5
##
## $aic
## [1] 8896.154
## attr(,"parameters")
```

```
## [1] 5
##
## attr(,"class")
## [1] "summary.asreml"
```

3. Genomic prediction for crossover count

Our motivation for using genomic prediction is as follows:

Is there selection on recombination rates at the individual level?



Does individual recombination rate affect reproductive success?

Problem: Recombination rate is measured using genotyped offspring.

Solution: Use genomic data to predict each bird's recombination phenotype.

I ran a basic demo model of genomic prediction using the package `hibayes`. I used the `BayesCpi()` function which assumes a small proportion of SNPs (1-Pi) have non-zero effects, and share the same variance, but Pi is not fixed. (*NB. We know from the MBE paper that crossover count is likely to be polygenic so this might not be the best model!*). We ran four prediction models:

1. FULL dataset (i.e. individual gametes)
2. MEAN dataset (i.e. mean values per individual with `n` fit as a fixed effect)
3. MEAN n>2 dataset (i.e. mean values for individuals that had 3 or more offspring), and
4. SAMPLE 1 (i.e. only one gamete was sampled from each individual)

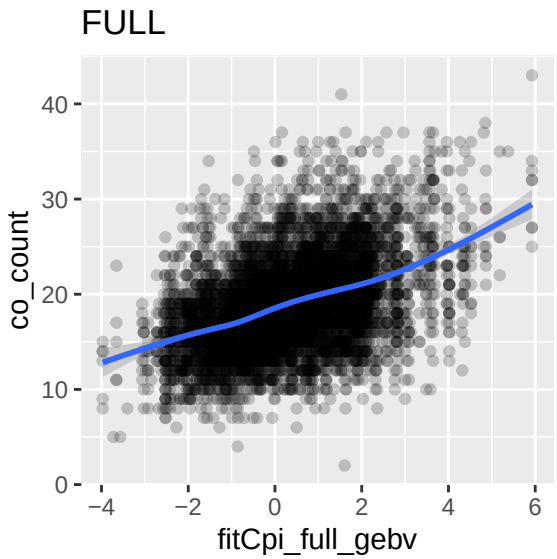
Here is an example of the model structure, please see `2_Sparrow_hibayes_run_SEJ.R` for more details.

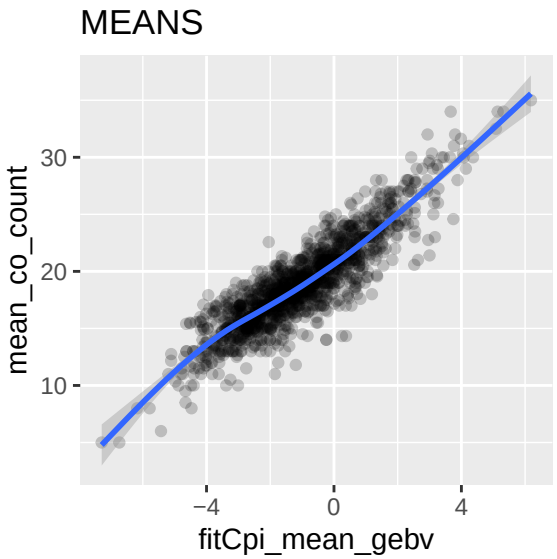
```
fitCpi_full <- ibrm(co_count ~ total_coverage,
                    data = recfull,
                    M = geno,
                    M.id = geno.id,
                    method = "BayesCpi",
                    Pi = c(0.98, 0.02), niter = 20000, nburn = 16000, thin = 5,
                    printfreq = 100, seed = 666666, verbose = TRUE)
```

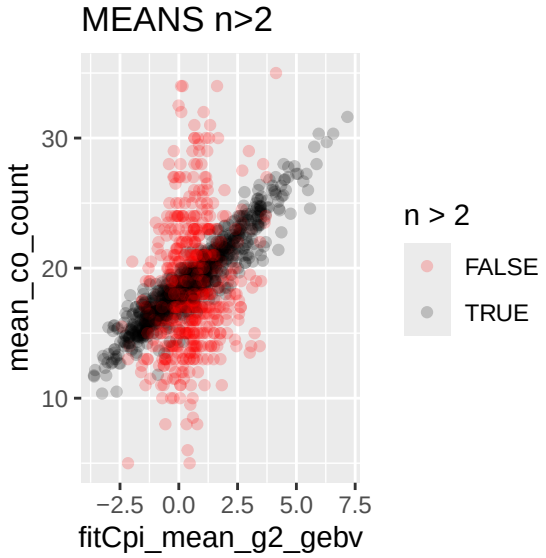
I did some plots of the GEBVs vs. True Phenotypes in the "training" population (NB. the red points on the third plot are the individuals with 2 or fewer measures, so we have phenotypes, they are analagous to a "test" population here)

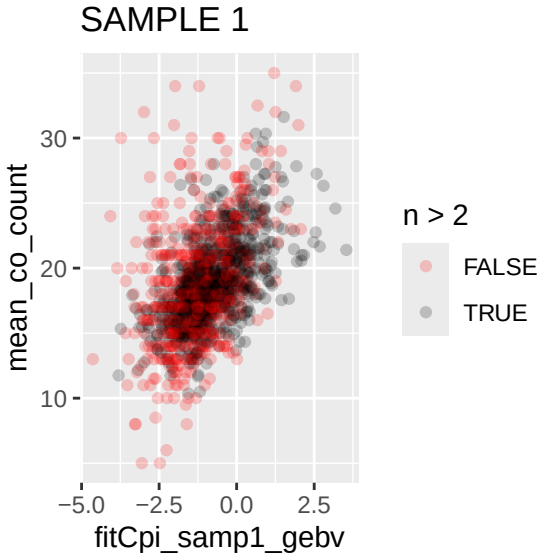
```
## Loading objects:
```

```
##   recfull
##   full_gebvs
##   fitCpi_full
##   .Random.seed
##   fitCpi_mean_g2
##   recsamp1
##   recmeans
##   fitCpi_mean
##   fitCpi_samp1
```

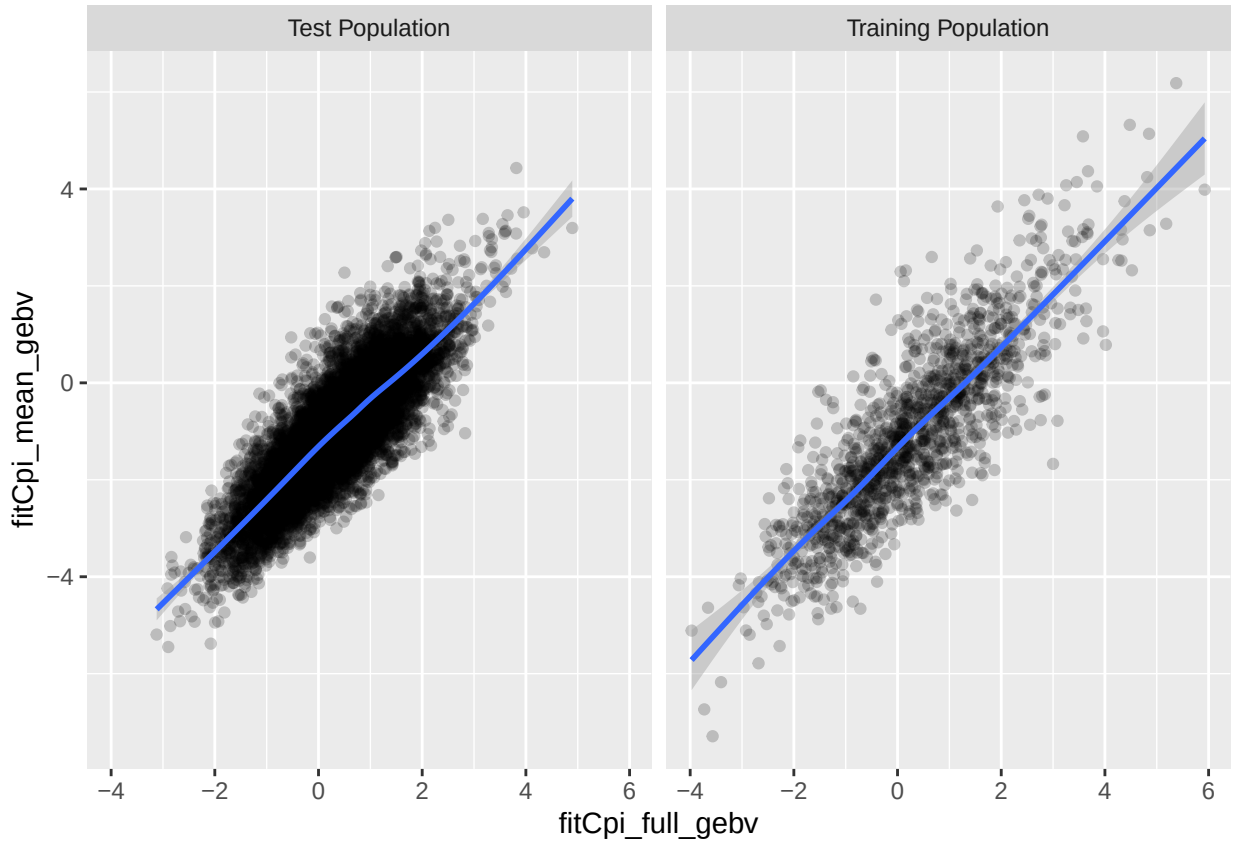








How do the FULL and MEANS GEBVs correlate?



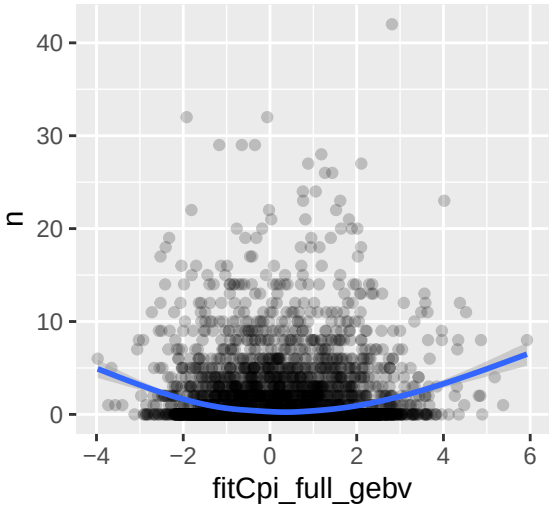

```
##
##  Pearson's product-moment correlation
##
## data:  full_gebvs_tab$fitCpi_full_gebv and full_gebvs_tab$fitCpi_mean_gebv
## t = 171.67, df = 12963, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8280353 0.8385547
## sample estimates:
##       cor
## 0.8333704
```

4. Genomic Prediction may give a false signal of disruptive selection on recombination rate.

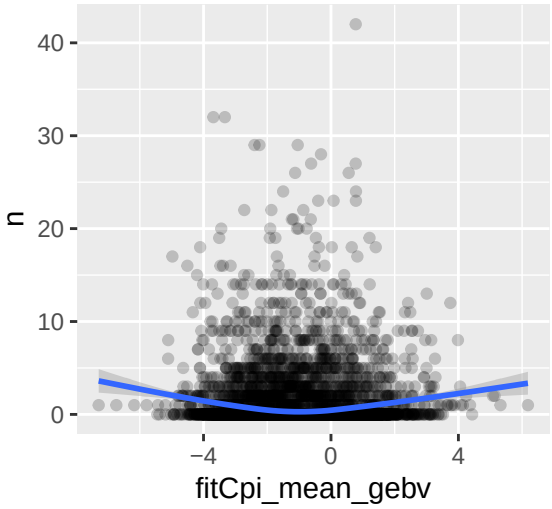
As shown in the figure above, one issue we have in the sparrows is that we cannot estimate recombination rates in individuals that did not have offspring. *Note: in our dataset, we have reproductive success information that doesn't necessarily correlate tightly with the number of gametes here! We know that some individuals will have had more offspring, but we just sampled fewer of them for genetic analysis and therefore have fewer estimates of their recombination. However, in this exploration, we will just focus on the number of gametes* `n` *as this still illustrates the problem effectively.*

Our analysis above calculated with GEBVs for all 12965 individuals in the genomic dataset. So the vast majority of these will have `n = 0`. Let's look at the relationship between GEBVs and n:

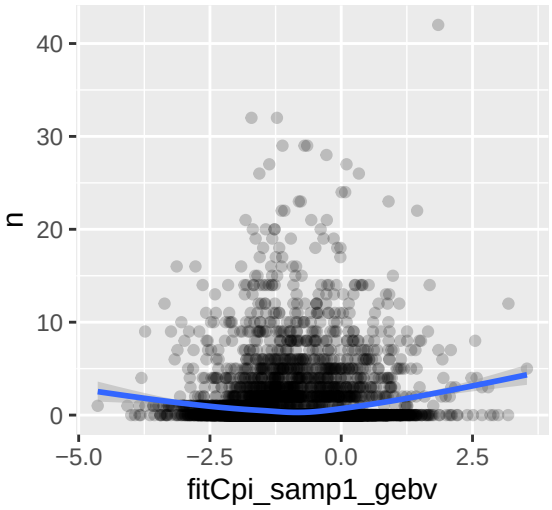
FULL

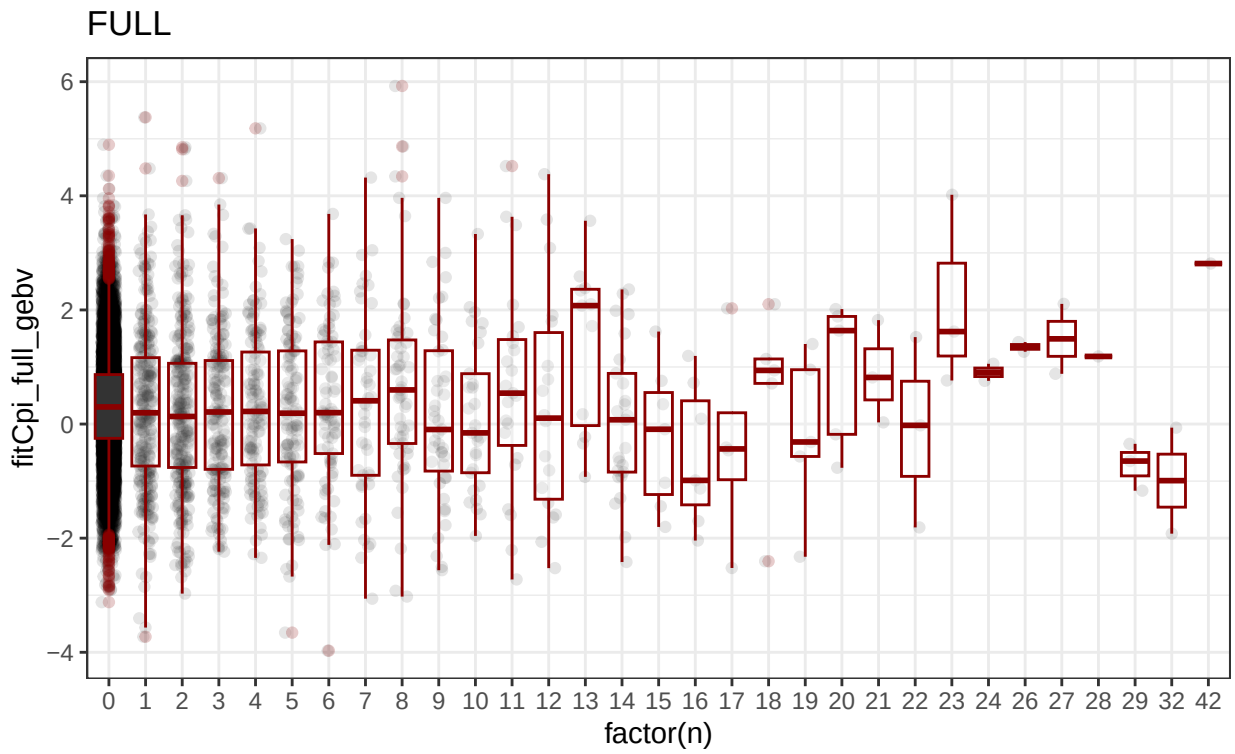


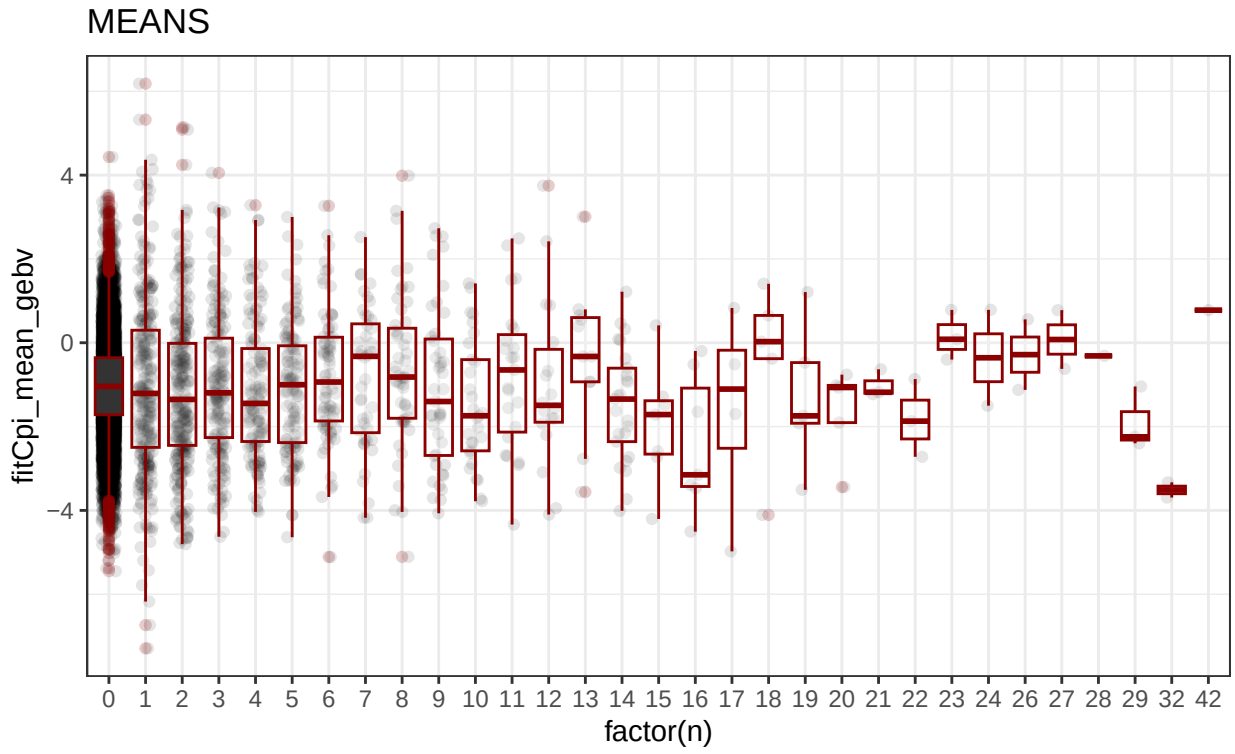
MEAN

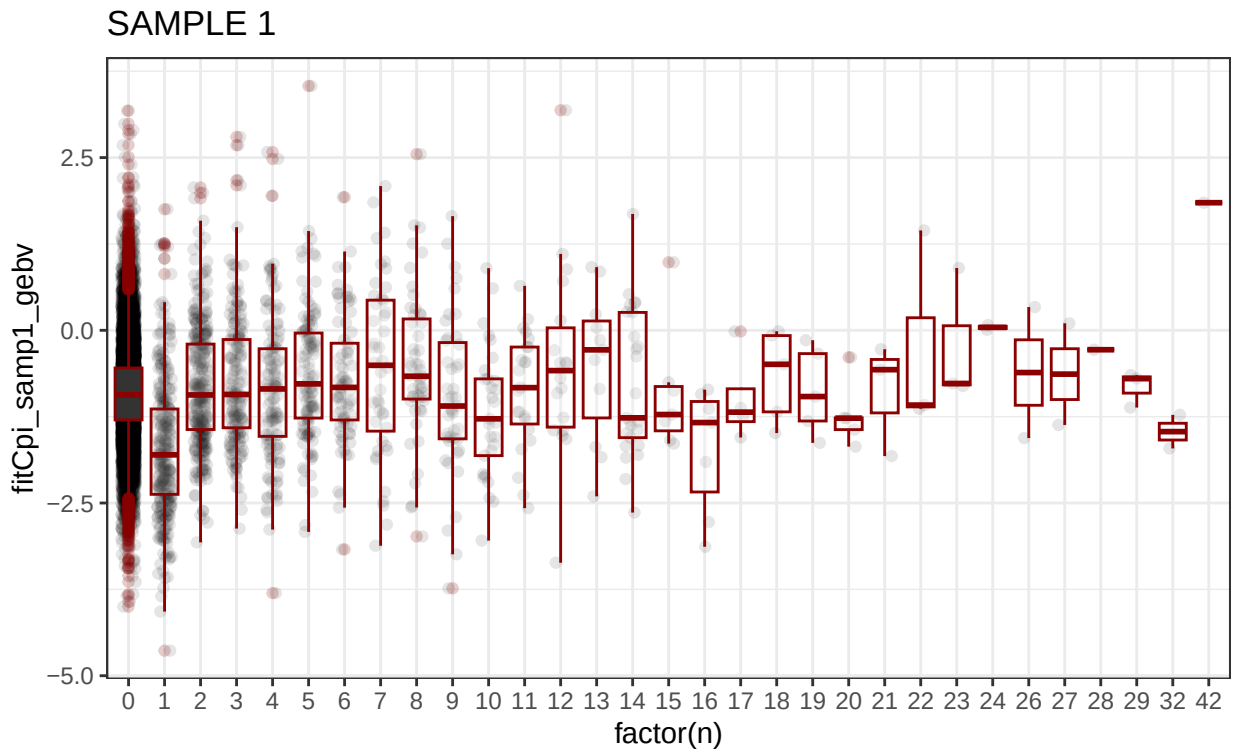


SAMPLE 1




FULL
fitCpi_full_gebv
6
4
2
0
−2
−4
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 26 27 28 29 32 42
factor(n)


MEANS
fitCpi_mean_gebv
4
0
−4
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 26 27 28 29 32 42
factor(n)

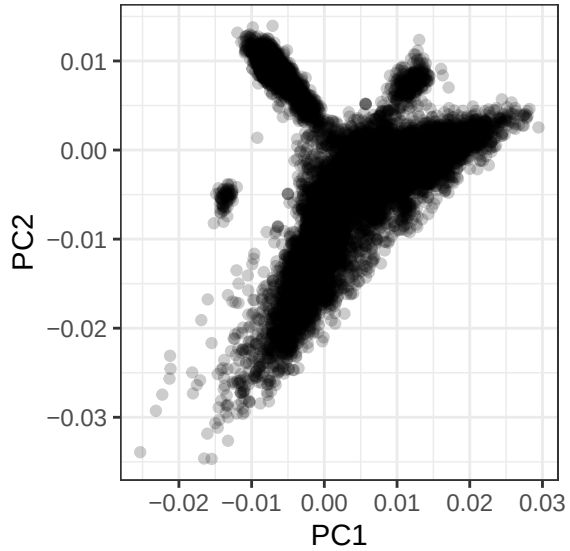


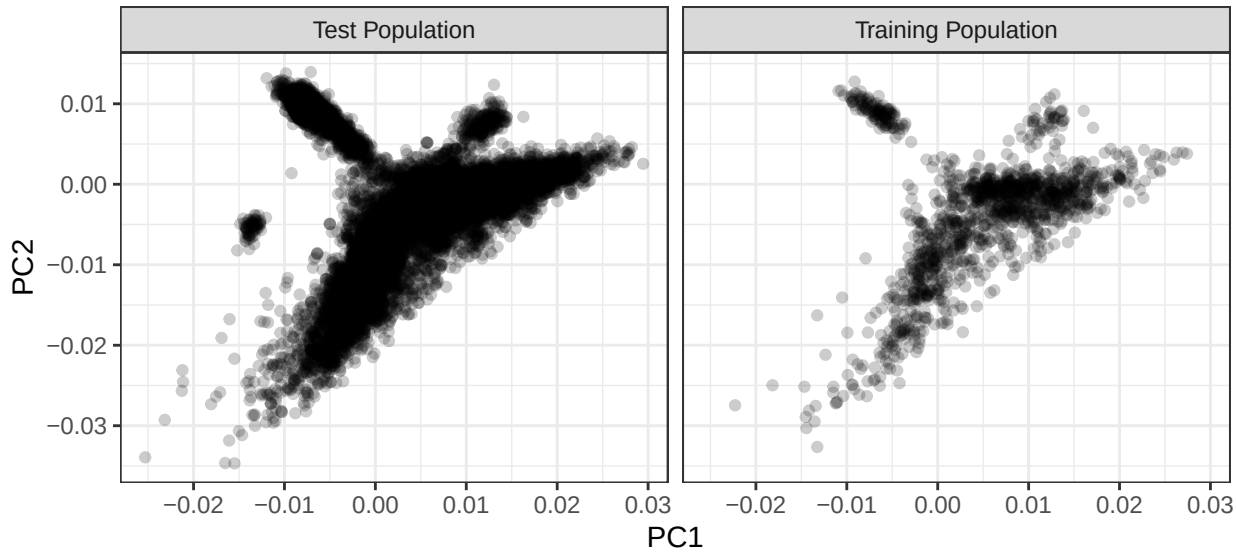
Why is there a signal of disruptive selection? Are zeros less related to the population? unconnected in the pedigree? CLUES?

5. Population Structure in the house sparrows

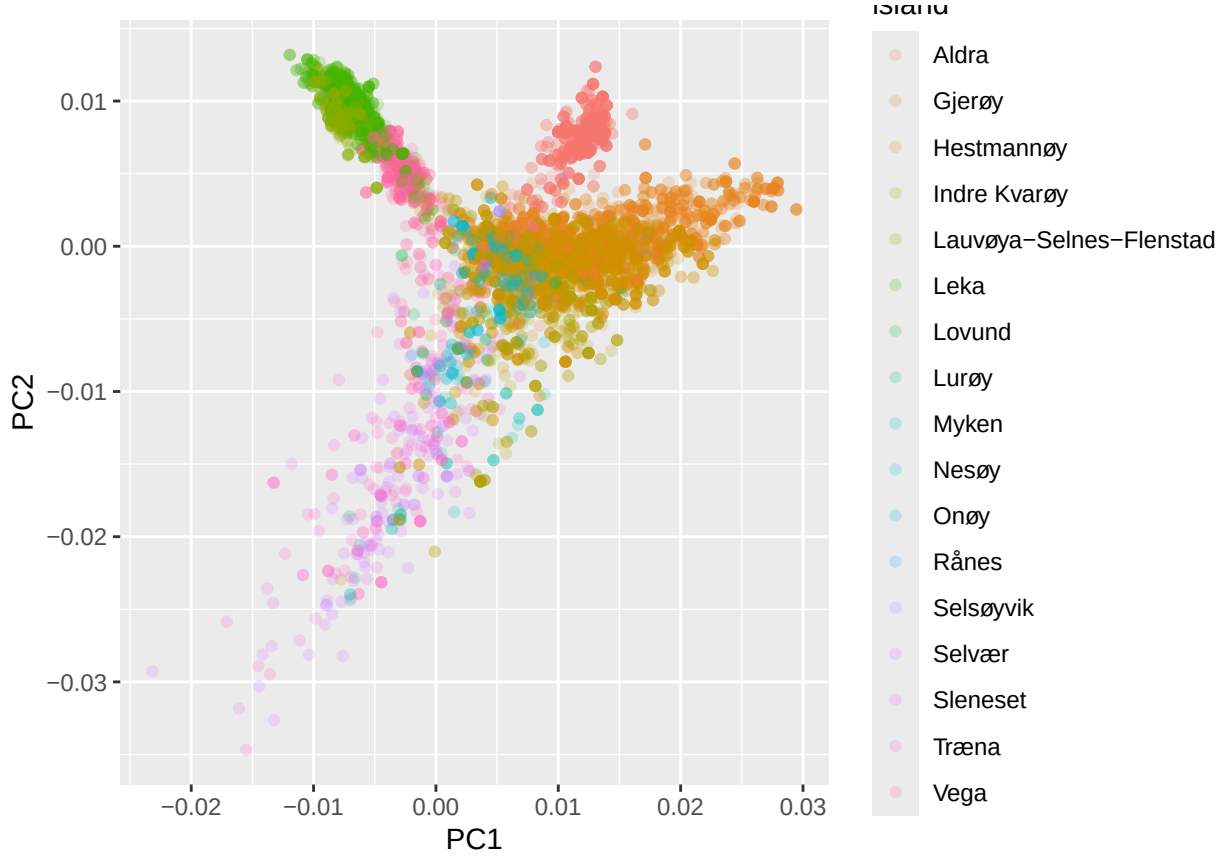
I ran a PCA in plink, and there is definitely a bit of population structure, which is similar in the training and test population:

(PLINK function: `plink --bfile data/70K_200K_maf_geno_mind_v5 --cow --autosome --pca --out data/70K_200K_maf_geno_mind_v5`)

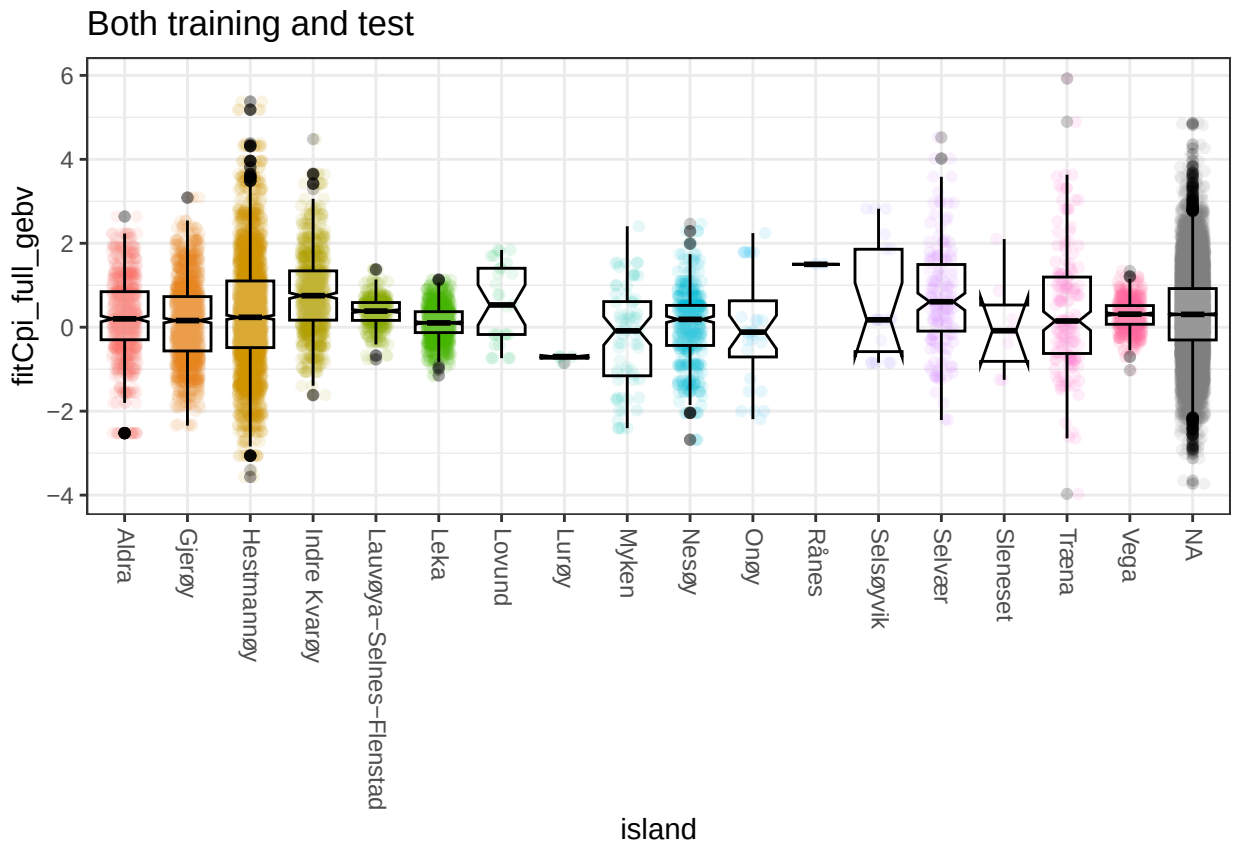


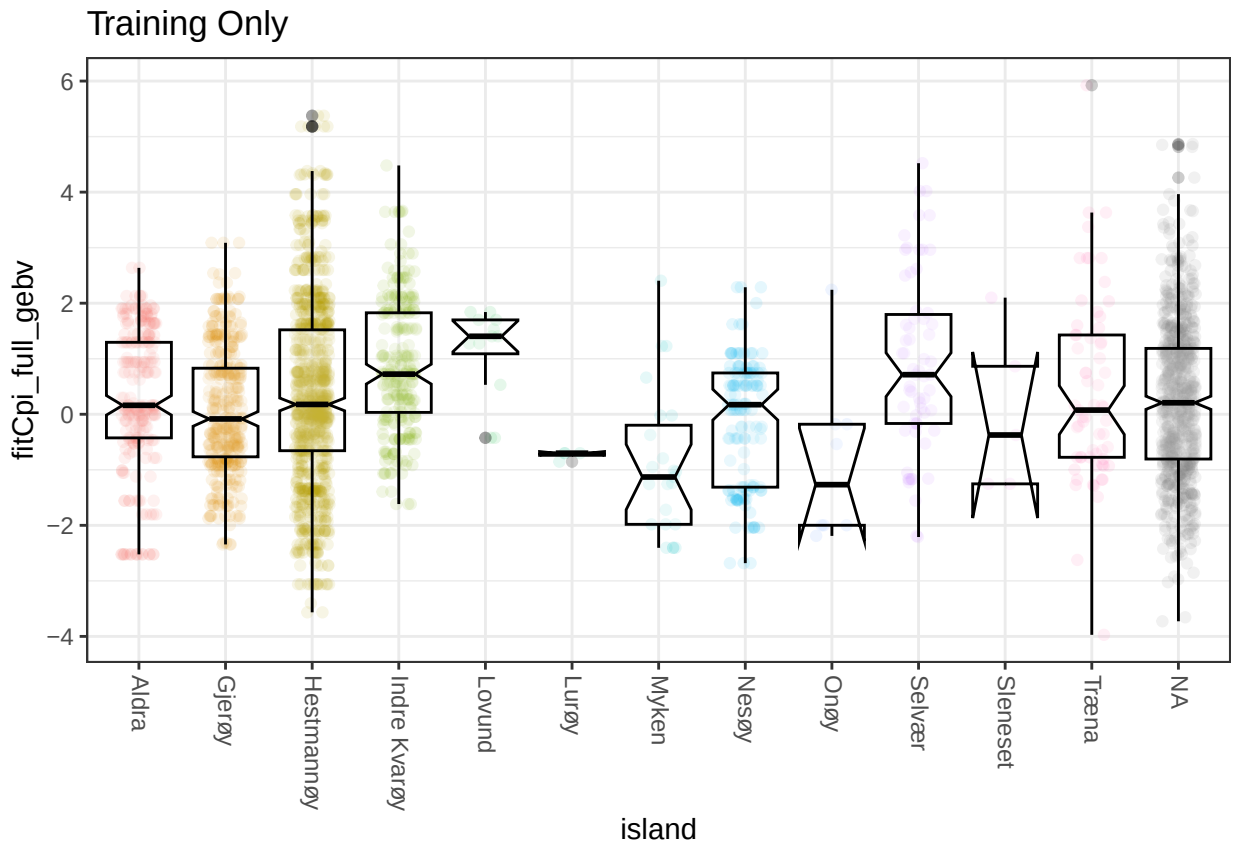


Based on information on locality for some birds, there is some island structure:



What are the differences in GEBVs?


Both training and test
fitCpi_full_gebv
6
4
2
0
−2
−4
Aldra
Gjerøy
Hestmannøy
Indre Kvarøy
Lauvøya−Selnes−Flenstad
Leka
Lovund
Lurøy
Myken
Nesøy
Onøy
Rånes
Selsøyvik
Selvær
Sleneset
Træna
Vega
NA
island


Training Only
fitCpi_full_gebv
6
4
2
0
−2
−4
Aldra
Gjerøy
Hestmannøy
Indre Kvarøy
Lovund
Lurøy
Myken
Nesøy
Onøy
Selvær
Sleneset
Træna
NA
island