Drosophia Longevity Summary

Eric Weine

3/2/2022

Introduction

The purpose of the paper is to investigate the existence of SNPs effecting longevity. According to standard evolutionary theory, genes that negatively effect longevity would be quickly selected out due to their extremely large effect on fitness. However, in humans and a variety of other species, these longevity effecting genes are observed. There are three explanations that this paper investigates:

(1) Antagonistic Pleiotropy: If a gene increases a fly's ability to produce offspring early in life, but causes it to on average die earlier, this gene could still persist in the population.

(2) Mutation Accumulation: A gene may only effect a fly's longevity late in life. Because selection is week at this point (because flies likely could have mated when younger), then the effect of selection may not be strong enough to eliminate these genes from the population.

(3) Environmental Shift: Genes may only effect longevity in particular environments. If a population evolved when exposed to a different environment, then it's plausible that genes that wouldn't have effected longevity in the prior environment would effect longevity in a different environment.

This summary will mostly concern the author's attempts to test which genes might satisfy condition 3, as that is currently the main purpose of my project.

Experimental Setup

The basic setup is that outbred flies were divided into two groups: one was exposed to a high sugar diet while the other was exposed to a standard laboratory diet. Specifically, groups of 10K flies were divided into 6 different cages. Three of the cages were fed a high sugar diet, while the other three were fed a standard diet. Every 3-7 days, 500 flies were sampled from each cage and allele frequency was measured. Specifically, for the sake of measuring any GxE effect, two times are of interest. T_0, the initial time, and T_N, the time at which only 500 flies were remaining in a given cage, and all 500 flies were measured.

GxE Methods

In order to derive some sort of statistical test to determine if there is a GxE interaction occurring at a particular locus, the authors employ a beta-binomial regression approach. In this case, it seems that the authors would like to fit a logistic regression, but because they are concerned about overdispersion of the data they decided to fit a beta-binomial regression. Because the flies were genotyped using a pool-seq approach, the actual genotypes of each fly are not known. Instead, for fly i, at some given locus, we have r_i read counts, y_i of which are mapped to the alternative allele. Thus, the authors assume that

$$y_i \sim Bin(r_i, \pi_i),$$

where π_i is the true genotype of fly i at the given locus (i.e. $\pi_i = 1$ if there are two copies of the alternative allele, $\pi_i = 0.5$ if there is one copy of the alternative allele, otherwise $\pi_i = 0$).

The authors use a logit link, and thus model π_i as

$$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + pool_i\beta_p + batch_i\beta_b + sex_i\beta_s + control_i\beta_c + treatment_i\beta_h$$

Note that $control_i$ is 1 for a fly on the control diet observed at T_N, and 0 otherwise (even for flies at T_0 who were ultimately put on the control diet). Similarly, $treatment_i$ is 1 for a fly on the high sugar diet observed at T_N, and 0 otherwise.

Note also that the authors incorrectly add an error term ϵ_i to equation for π_i without specifying the distribution for said error. This is incorrect because the error in the logistic regression setup occurs as a result of binomial sampling variance, not an error term for the mean parameter of the binomial distribution.

This model is fit for all sampled flies at time T_0 and T_N. To test for a GxE effect, the authors look at the p-values of β_c and β_h. Conceptually, β_c will be large if having the alternative allele is observed more frequently at T_N than it was at T_0 in the control group. The locus is considered to have a time effect specific to the high sugar environment if the p-value for β_h passes a 10% FDR threshold (it's unclear to me if this was done by calculating q-values or if they performed BH, though I think the former) and the p-value for $\beta_c > .05$ (vice versa for a control specific effect). The authors consider a locus to have a shared effect if at least one of β_h of β_c passes a 10% FDR threshold, and the nominal p-value for the other parameter is $< .05$. This is an interesting approach, and notably has no way to identify loci that have effects in both environments, but the effect is magnified in one environment. This could be a significant problem if we expect that GxE effects primarily occur through amplification, as proposed in our lab's GxSex analysis.

Extensions to the GxE method / comments

I think this paper takes a very interesting approach to modelling GxE interactions. Because I'm not very familiar with the literature, I'm not sure how novel this approach is. In a more general case in which some group is exposed to some treatment and the other is a control group, I would have expected some variable for longevity of fly i, l_i, where this could be modeled as

$$l_i = \beta_0 + y_i\beta_1 + I(treatment = sugar)\beta_2,$$

where y_i is the alternative allele count, as defined above. However, this experimental setup is incompatible with this model, because the longevity of individual flies is not actually measured.

The departure from this model is a bit unfortunate, because it makes it more difficult to identify more General GxE effects. Theoretically, there are 3 different scenarios for a GxE interaction:

(1) $|B_h| > 0$, $B_c = 0$.

(2) $|B_c| > 0$, $B_h = 0$.

(3) $|B_c| > 0$, $|B_h| > 0$, $B_c \neq B_h$.

The approach of the authors is able to identify the first two effects, but not the third. It would be interesting to investigate how one might identify case 3.

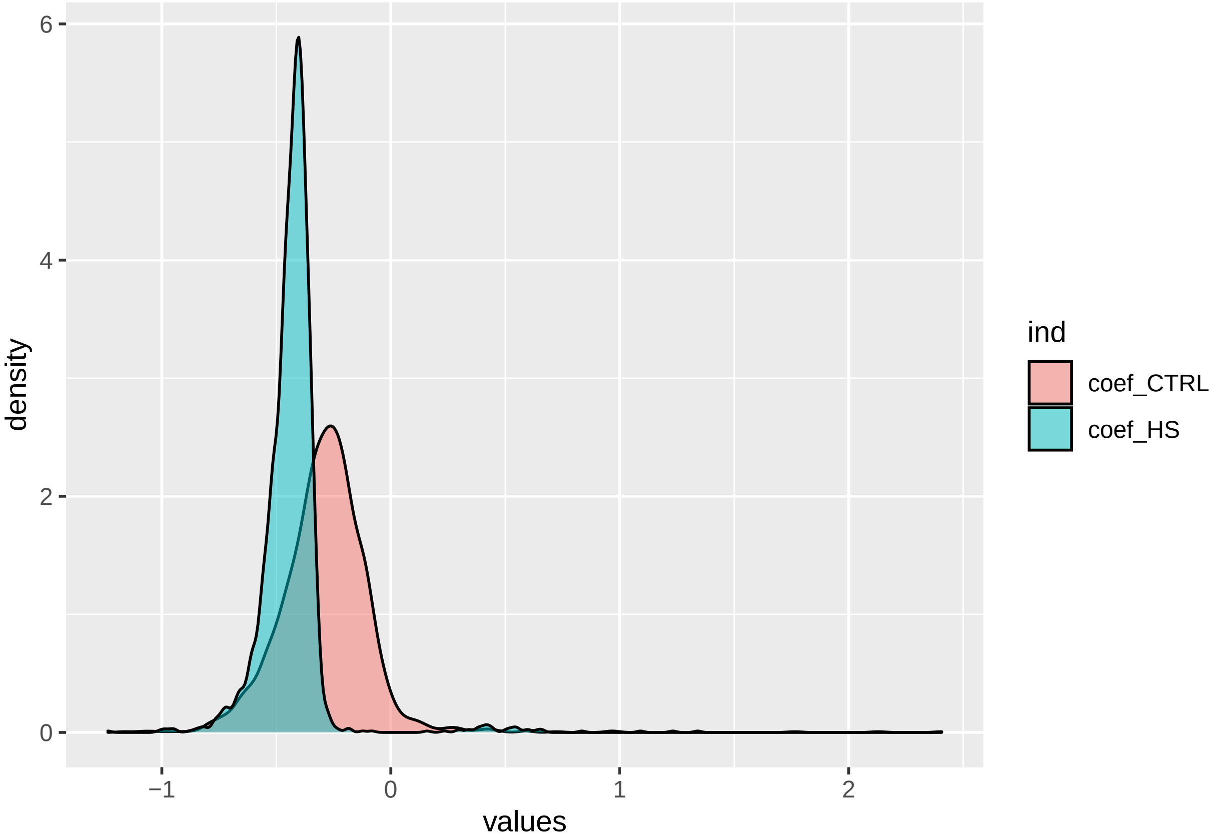
I would like to create some sort of permutation test in order to identify this. We will load in the set of summary statistics provided by the authors.

```
summary_table <- read.delim('data/SummaryTable_allsites_12Nov20.txt')
summary_table <- summary_table %>%
  select(c(site, pval_CTRL, pval_HS, coef_CTRL, coef_HS, q_perm_CTRL, q_perm_HS, sig_cat)) %>%
  filter(sig_cat != 'NS')
```

We filter down to the set of SNPs that is considered significant in at least one environment. Below, we plot the distribution of the coefficients for the different environments.

```
coef_df <- stack(summary_table, select = c(coef_CTRL, coef_HS))

ggplot(data = coef_df, aes(x=values, fill=ind)) +
  geom_density(alpha=0.5)
```

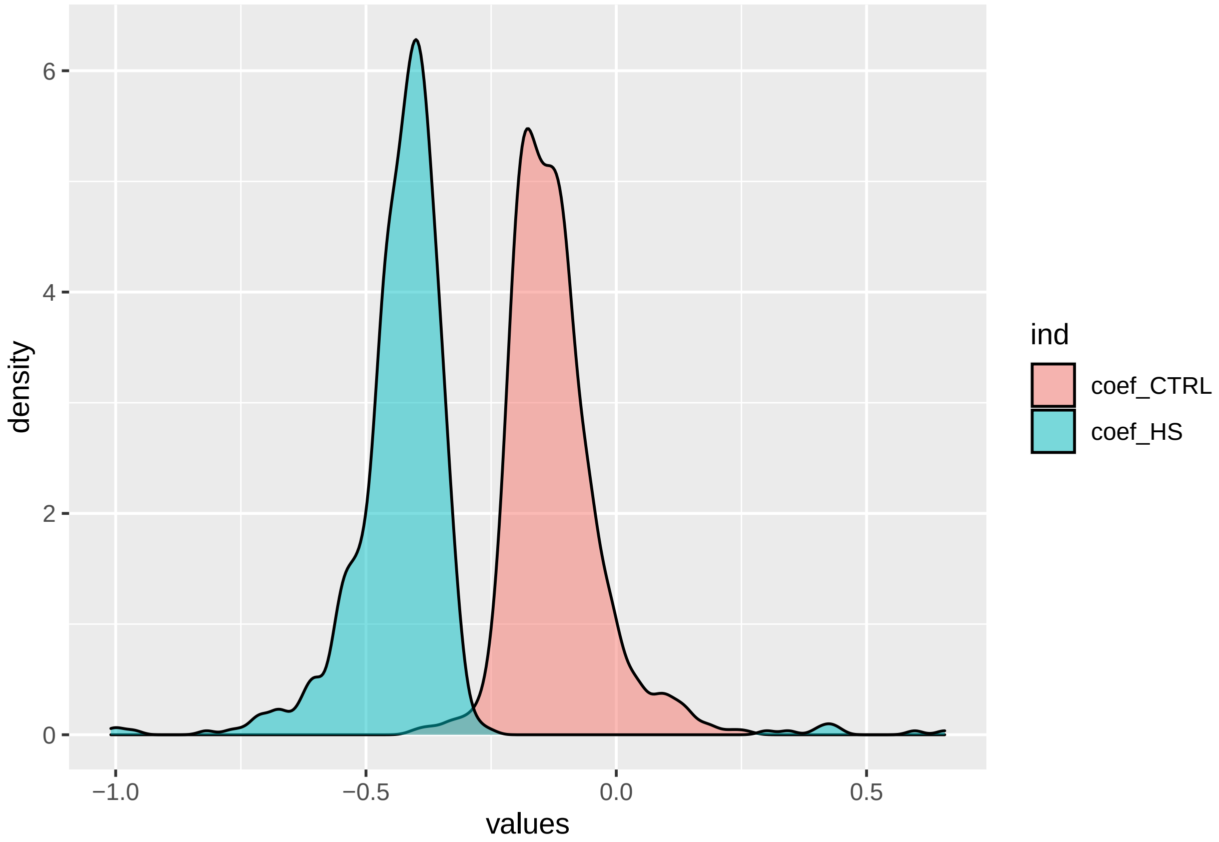


For all but three of the effects, I see that the effect is identified as either shared or HS only. So, it would be interesting to see what the distribution of parameters looks like for shared effects vs. high sugar only effects. First, let's look at high sugar.

```
summary_table_hs <- summary_table %>%
  filter(sig_cat == 'HS')

coef_hs_df <- stack(summary_table_hs, select = c(coef_CTRL, coef_HS))

ggplot(data = coef_hs_df, aes(x=values, fill=ind)) +
  geom_density(alpha=0.5)
```

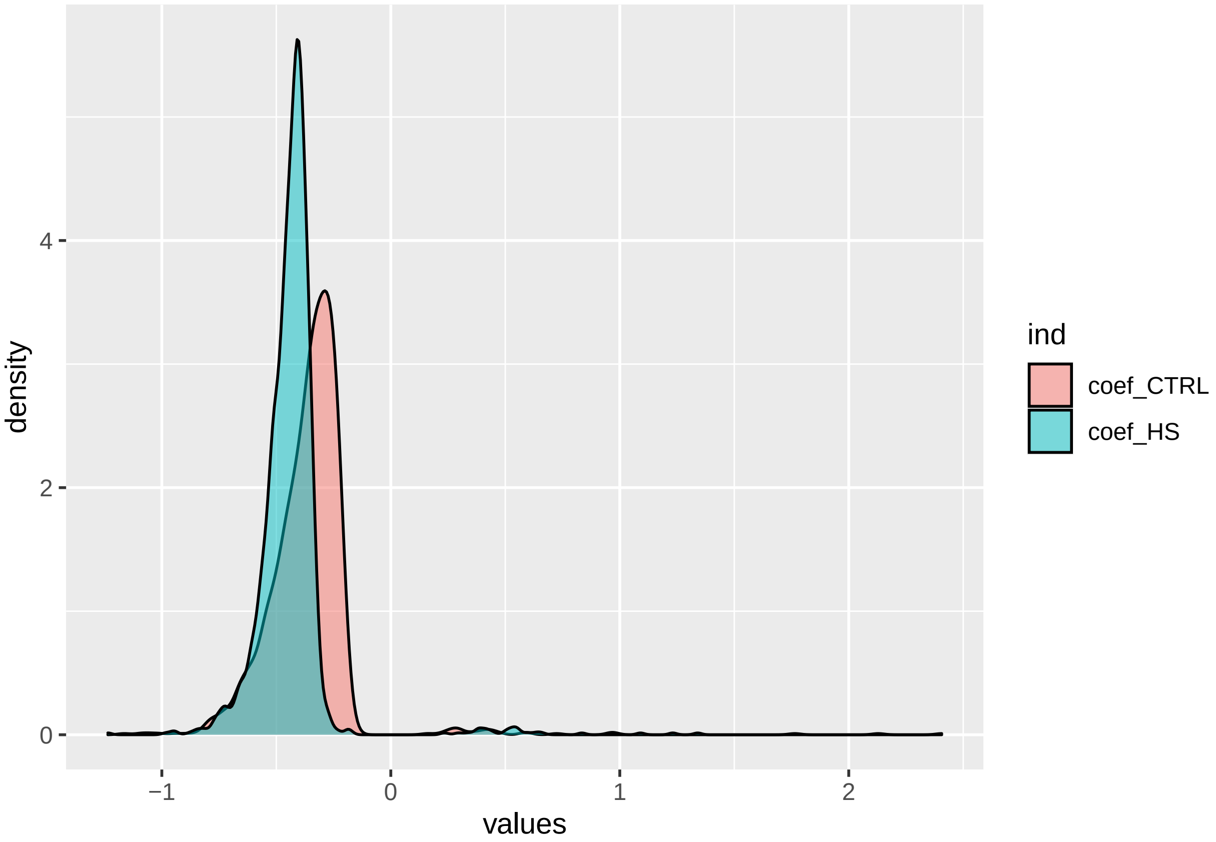


We see very strong division in the effect size here, as would be expected. Notice, though, that the distribution of control coefficients is still highly negative. It's possible that in reality these SNPs do have a negative effect, but it fails to reach the significance level. Now, let's examine the distribution of coefficients for the shared effects.

```
summary_table_shared <- summary_table %>%
  filter(sig_cat == 'shared')

coef_shared_df <- stack(summary_table_shared, select = c(coef_CTRL, coef_HS))

ggplot(data = coef_shared_df, aes(x=values, fill=ind)) +
  geom_density(alpha=0.5)
```



Again, unsurprisingly, the distributions here are much closer. However, the mean of the high sugar effects is clearly larger in magnitude. It would be interesting to see if we could design a statistical test that would determine if $B_h < B_c$, even if both are non-zero.

The big question is how we can develop a null distribution for the case that the two effects are equal.

A naive approach might make a strong parametric assumption about the distribution of regression parameters. I'm not sure how this would perform. Suppose we made the following assumptions:

$$\widehat{\beta_h} \sim N(\beta_h, \sigma^2_{B_h})$$
$$\widehat{\beta_c} \sim N(\beta_c, \sigma^2_{B_c})$$

Then, it's a well known statistical fact that under the null hypothesis that $\beta_h = \beta_c$, the distribution of $\widehat{\beta_h} - \widehat{\beta_c}$ is $N(0, \sigma^2_{B_h} + \sigma^2_{B_c})$. Using the estimated regression parameters and variances, this statistical test could be performed for all regression parameters of shared effects.

I believe a better approach would be to conduct a permutation test similar to what the authors do when evaluating GxE vs. shared genetic effects. For significant all SNPs labelled as "shared" in significance, we could shuffle the labels for HS and C. Then, we could rerun the analysis a number of times and look at the test statistic $\widehat{\beta_h} - \widehat{\beta_c}$ for each simulated dataset. In theory, this provides a null distribution of the test statistic, where the null hypothesis is that $\beta_h = \beta_c$. Then, we could derive a p-value for each test by comparing the calculated values of $\widehat{\beta_h} - \widehat{\beta_c}$ in the actual dataset to the derived null.

```
# Here, load in original datasets and then filter down to only the shared significance sites
shared_signif_df <- summary_table %>%
  filter(sig_cat == "shared")

alt <- fread('data/29Jun20_merged_alt_counts_allCHR.txt')
```

```r
both <- fread('data/29Jun20_merged_both_counts_allCHR.txt')

# filter down to just significant SNPs in the alt and both df
alt <- alt %>%
  filter(site %in% shared_signif_df$site)

both <- both %>%
  filter(site %in% shared_signif_df$site)

info <- read.delim('data/29Jun20_merged_sample_info.txt')
```

```r
# This code has a bug / I don't think it's the best way to do this

# Now, create a null distribution here based on the data
num_sims <- 100

t0_df <- info %>%
  filter(timepoint == "T0")
tn_df <- info %>%
  filter(timepoint == "TN")

for(i in c(1:num_sims)) {

  # first, shuffle the labels for TN
  sim_tn_info_df <- tn_df %>%
    mutate(condition = sample(condition))

  sim_info_df <- rbind(t0_df, sim_tn_info_df)

  # Now, run each regression and get all of the coefficients

  for(j in c(1:nrow(alt))) {

    sim_info_df$tmp_x <- as.vector(t(alt[j,-1]))
    sim_info_df$tmp_y <- as.vector(t(both[j,-1]))

    tmp_info <- subset(sim_info_df, tmp_y>0 & sex!='unknown')

    mod1 <- tryCatch(
      betabin(
        cbind(tmp_x, tmp_y - tmp_x) ~ condition + sequencing_batch + meta_cage + sex,
        ~1,
        data=tmp_info
      ), error=function(x){})

    print(0)

  }


}
```

Scratch

Below is code that I wrote to generate power simulations for the interaction effect in the interaction model.

```r
generate_genotypes <- function(maf, n) {

  sample(
    x=c(0, 0.5, 1),
    size=n,
    prob=c((1 - maf) ^ 2, 2 * maf * (1 - maf), maf ^ 2),
    replace = TRUE
  )

}

# generate df of time, environment, allele count
generate_simulated_dataset <- function(n, maf, reads_mean, hs_perc_dec, c_perc_dec) {

  # first, generate T0
  n_t0 <- floor(n / 2)
  genotypes_t0 <- generate_genotypes(maf, n_t0)

  total_reads_t0 <- pmax(1, rpois(n_t0, reads_mean))
  reads_t0 <- rbinom(n = n_t0, size = total_reads_t0, prob = genotypes_t0)

  t0_df <- data.frame(
    time_lab = "T0",
    condition = "both",
    total_reads = total_reads_t0,
    ma_reads = reads_t0
  )

  # Now, generate TN for the control group.
  maf_c_tn <- maf * (1 - c_perc_dec)
  n_c_tn <- floor((n - n_t0) / 2)
  genotypes_c_tn <- generate_genotypes(maf_c_tn, n_c_tn)

  total_reads_tn_c <- pmax(1, rpois(n_c_tn, reads_mean))
  reads_c_tn <- rbinom(n = n_c_tn, size = total_reads_tn_c, prob = genotypes_c_tn)

  tn_c_df <- data.frame(
    time_lab = "TN",
    condition = "C",
    total_reads = total_reads_tn_c,
    ma_reads = reads_c_tn
  )

  # Now, generate TN for the treatment group.
  maf_hs_tn <- maf * (1 - hs_perc_dec)
  n_hs_tn <- n_c_tn
  genotypes_hs_tn <- generate_genotypes(maf_hs_tn, n_hs_tn)

  total_reads_tn_hs <- pmax(1, rpois(n_hs_tn, reads_mean))
  reads_hs_tn <- rbinom(n = n_hs_tn, size = total_reads_tn_hs, prob = genotypes_hs_tn)
  tn_hs_df <- data.frame(
    time_lab = "TN",
    condition = "HS",
    total_reads = total_reads_tn_hs,
    ma_reads = reads_hs_tn
  )

  sim_df <- rbind(t0_df, tn_c_df, tn_hs_df)
  return(sim_df)

}

do_power_sim_int_model <- function(
  num_sims,
  pval_thresh,
  maf_c_dec_range,
  maf_hs_dec_range,
  maf_dec_range,
  var) {

  is_rejected <- numeric(num_sims)

  for(i in c(1:num_sims)) {

    print(i)
    maf <- runif(1, 0, .5)
    maf_perc_dec <- runif(1, maf_dec_range[1], maf_dec_range[2])
    hs_perc_dec <- runif(1, maf_hs_dec_range[1], maf_hs_dec_range[2]) + maf_perc_dec
    c_perc_dec <- runif(1, maf_c_dec_range[1], maf_c_dec_range[2]) + maf_perc_dec
    sim_df <- generate_simulated_dataset(
      n=2000,
      maf=maf,
      reads_mean=10,
      hs_perc_dec=hs_perc_dec,
      c_perc_dec=c_perc_dec
    )
    sim_df <- sim_df %>%
      mutate(time = case_when(time_lab == "T0" ~ 0, time_lab == "TN" ~ 1),
             env = case_when(condition == "both" ~ 0,
                             condition == "C" ~ 0,
                             condition == "HS" ~ 1))
    mod <- betabin(
      cbind(total_reads, total_reads - ma_reads) ~ time + env:time, ~1,
      data=sim_df,
      control = list(maxit = 4000)
    )
    pval <- attributes(summary(mod))$Coef[var,4]
    reject <- as.numeric(pval < pval_thresh)
    is_rejected[i] <- reject

  }

  power <- mean(is_rejected, na.rm = TRUE)
  return(power)

}
```