pqa

LMN

23/04/2021

1: concordance in RIL QTL effects and population allele frequency effects

fertility We have estimates of SNV effects on fitness (which I'll call s) from both poolseq, over generations 0-240, and from RIL (mostly A6140) fertility (which I'll call α). Are they correlated?

Here's one approach:

```
# load everything
load('~/Documents/cemee/poolseq/v2/fertility_tests.rda', verbose=T)
```

```
## Loading objects:
##   rilsnps
##   rilgt
##   X
##   Xld
##   K
##   phe
##   padj
##   refcadj
##   altcadj
##   gs
##   reps
##   popn
```

```
snps = rilsnps
refaf = refcadj/(refcadj+altcadj)
### phenotypes
kable(h(phe), digits=3)
```

	line	fertility	Std. Error	z value	Pr(>|z|)	pop	popr	env	L1
NGM.1	A6140L1	0.493	0.022	22.101	0.000	A6140	A6140	NGM	1.638
NGM.2	A6140L10	0.881	0.017	52.845	0.000	A6140	A6140	NGM	2.414
NGM.3	A6140L101	0.079	0.026	3.014	0.003	A6140	A6140	NGM	1.082
NGM.4	A6140L102	0.069	0.023	2.952	0.003	A6140	A6140	NGM	1.072
NGM.6	A6140L105	-0.339	0.042	-7.996	0.000	A6140	A6140	NGM	0.713
NGM.7	A6140L106	-0.015	0.025	-0.595	0.552	A6140	A6140	NGM	0.985

```
### poolseq data
# SNVs tested (CeMEE diallelic v2, inc. denovo mutations)
kable(h(rilsnps))
```

chrom	pos	ref	alt	cM
1	1222	A	C	0
1	1291	G	T	0
1	1761	G	A	0
1	1799	T	C	0
1	1902	C	G	0
1	1933	C	G	0

```
# read counts (adjusted for Neff, as in Feder 2012, using the mean value for CA pops vs A6140)
# and allele frequencies
kable(hh(refcadj))
```

A0-5	A00	A110	A210	A310	A410
96	72	91	115	137	61
98	62	93	97	93	50
117	73	100	82	95	32
137	64	95	80	92	35
121	94	85	112	135	55
97	47	55	54	58	24

```
# p-values from quasibinomial tests
# `full, int, g` are -log10 LRT p-vals for nested tests estimating effects of
# full : SNV + generation*replicate lineage
# int  : lineage-specific SNV effects (generation:replicate)
# g    : deterministic SNV effects (generation)
kable(h(padj), digits = 2)
```

chrom	pos	ref	alt	cM	qual	maf	mafq	full	int	g	b
1	1222	A	C	0	3290.78	0.12	[0.11947,0.1252)	0.61	0.92	0.64	0
1	1291	G	T	0	4477.06	0.21	[0.21090,0.2197)	0.01	0.90	3.86	0
1	1761	G	A	0	103503.00	0.20	[0.19525,0.2027)	0.00	0.88	29.24	0
1	1799	T	C	0	136403.00	0.29	[0.28478,0.2970)	0.00	0.96	28.82	0
1	1902	C	G	0	8541.10	0.19	[0.18790,0.1952)	0.00	0.91	4.80	0
1	1933	C	G	0	190332.00	0.46	[0.45385,0.4694)	0.00	0.97	30.85	0

```
# the structure of the poolseq data (generations and reps)
kable(table(gs, reps))
```

	0	1	2	3	4	5	6
0	2	0	0	0	0	0	0
10	0	1	1	1	1	1	1
30	0	1	1	1	0	0	0
60	0	1	1	0	1	0	1
100	0	1	1	1	1	1	1
140	0	0	0	0	1	1	1
150	0	0	1	1	1	1	1
172	0	0	0	0	1	1	1
176	0	1	1	1	0	0	0
190	0	1	1	1	0	0	0

	0	1	2	3	4	5	6
206	0	0	0	0	1	1	0
208	0	1	1	1	0	0	0
240	0	1	1	1	1	1	0

```
### genotypes, and a GRM (K) built from LD-pruned genotypes
kable(hh(rilgt))
```

A6140L1	A6140L10	A6140L100	A6140L101	A6140L102	A6140L104
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
0	0	0	0	0	0
1	1	1	1	1	1

```
kable(hh(K), digits=3)
```

	A6140L1	A6140L10	A6140L101	A6140L102	A6140L105	A6140L106
A6140L1	1.007	-0.051	-0.011	-0.009	-0.070	0.039
A6140L10	-0.051	0.919	0.027	-0.021	0.035	0.003
A6140L101	-0.011	0.027	0.924	-0.047	0.072	-0.011
A6140L102	-0.009	-0.021	-0.047	1.011	-0.034	-0.024
A6140L105	-0.070	0.035	0.072	-0.034	0.876	-0.062
A6140L106	0.039	0.003	-0.011	-0.024	-0.062	0.959

The models fit to obtain SNV selection coefficients were:

- $fit0 = glm(cbind(refcadj[i,], altcadj[i,])\ reps, family =' quasibinomial')$
- $fit1 = glm(cbind(refcadj[i,], altcadj[i,])\ gs + reps, family =' quasibinomial')$
- $fit2 = glm(cbind(refcadj[i,], altcadj[i,])\ gs * reps, family =' quasibinomial')$

Where `full` compares the saturated model 2 to model 0 by likelihood ratio, `int` compares the saturated model 2 to model 1, and `g` compares model 1 to model 0.

At a given poolseq threshold, measure SNV effects on fertility by LMM. This is not a full test of concordance, just how well RIL fertility α at candidate poolseq loci predict the estimated selection coefficients s.

For illustrative purposes, we'll take a single threshold here. This threshold could be optimised later by cross-validation on predictive power.

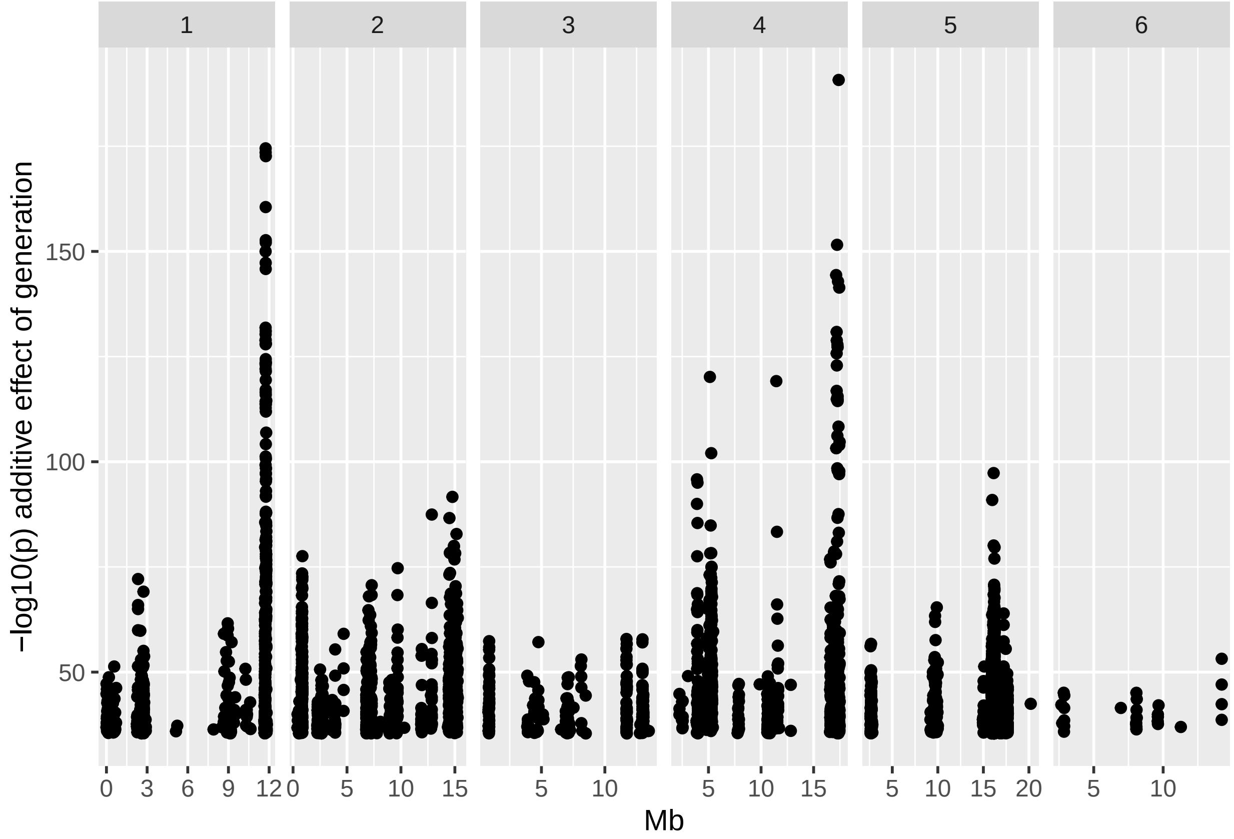
I use a LD threshold to initially bin SNVs into somewhat independent loci (e.g., 0.5cM or 1cM, based on F_2 map length). This is not ideal, in that LD obviously changes during the experiment, and a single threshold doesn't account for variability in LD. An improvement is to use the realized RIL LD between these bins in a mixed model.

```
# p-value threshold
(cut = -log10(1e-30/nrow(padj)))
```

```
## [1] 35.42301
```

```
# initial bin width in cM (F2 map scale)
gwin = 0.5
```

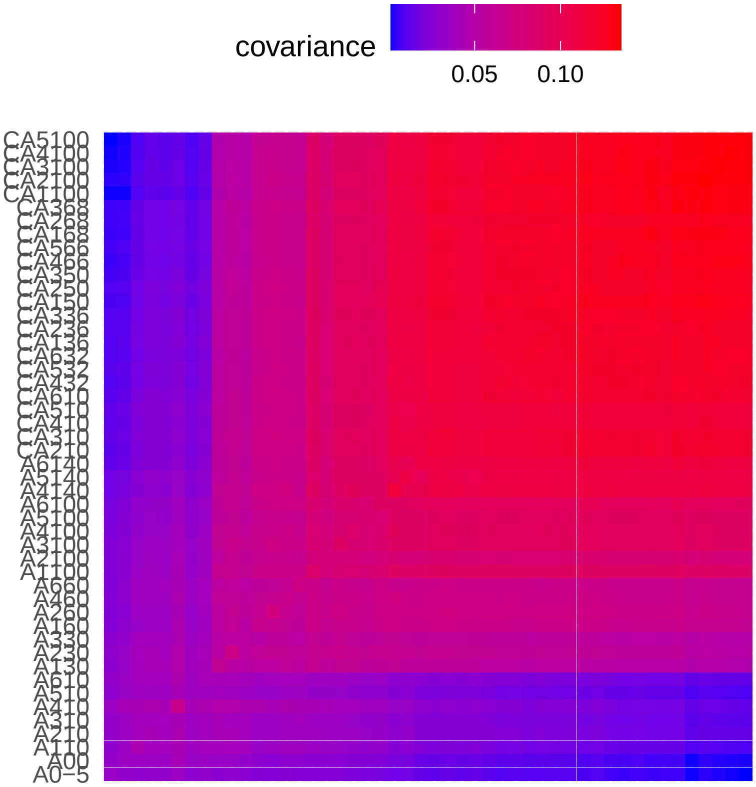
```
# illustrative sample covariance at thresholded loci (min p per gwin cM block)
# a few low Ne outliers based on allele frequencies have been removed
pc = subset(padj, g > cut)
ggplot(pc, aes(pos/1e6, g)) + geom_point() + facet_grid(.~chrom, scales = 'free') + labs(x='Mb', y = '-
```



```
pc$b = round(pc$cM*(1/gwin))/(1/gwin)
pcw = merge(pc, aggregate(data = pc, g~chrom+b, max))
cat(sprintf('-log10(p) > %.2f: %s markers in %s (%scM) bins pass\n', cut, nrow(pc), nrow(pcw), gwin))
```

```
## -log10(p) > 35.42: 4166 markers in 99 (0.5cM) bins pass
```

```
ix = which(paste(padj$chrom, padj$pos) %in% paste(pcw$chrom, pcw$pos))

covc = reshape2::melt(cov(refaf[ix,]))
ggplot(covc, aes(Var1, Var2, fill=value)) + geom_tile() +
  scale_fill_gradient('covariance', high='red', low='blue') +
  theme(legend.position = 'top', axis.ticks = element_blank(), axis.text.x = element_blank()) +
  labs(x='', y='') + coord_equal()
```

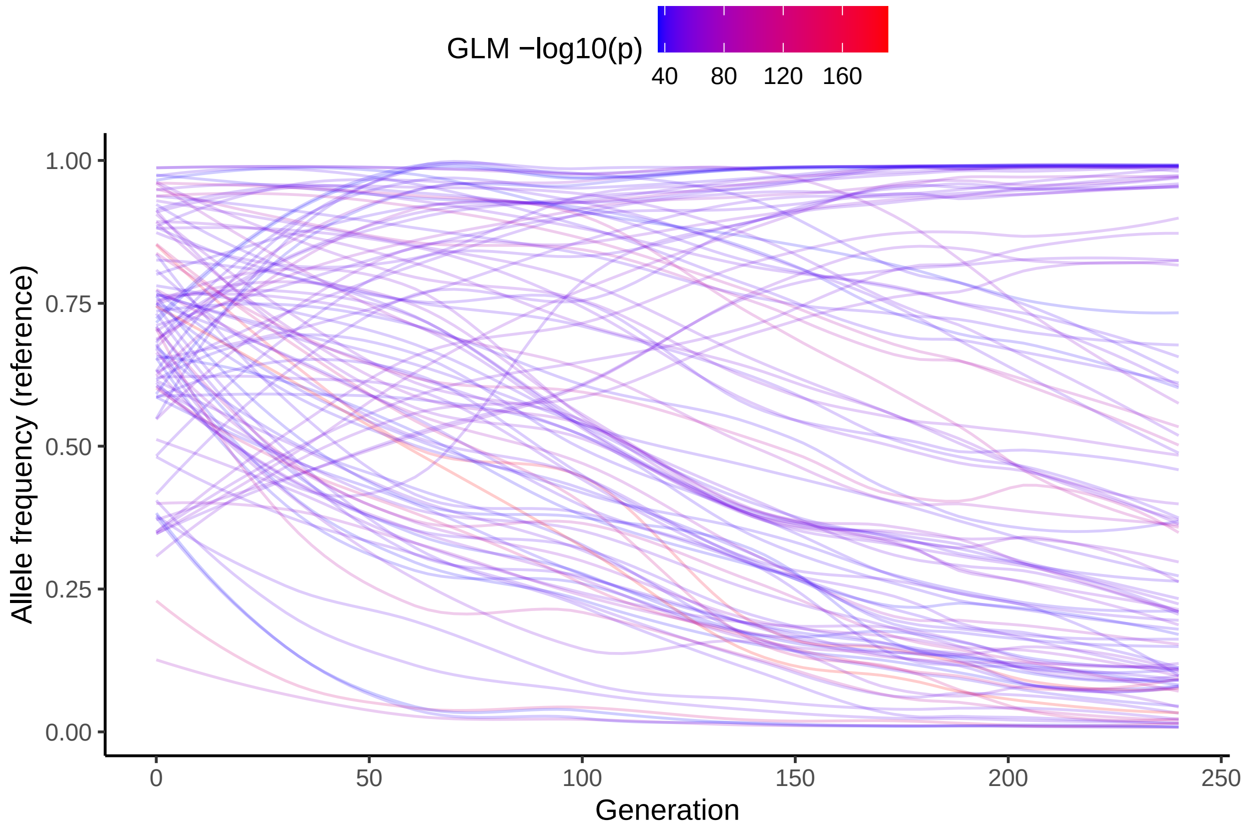


```
# most outliers removed, A660 could be dropped too...
```

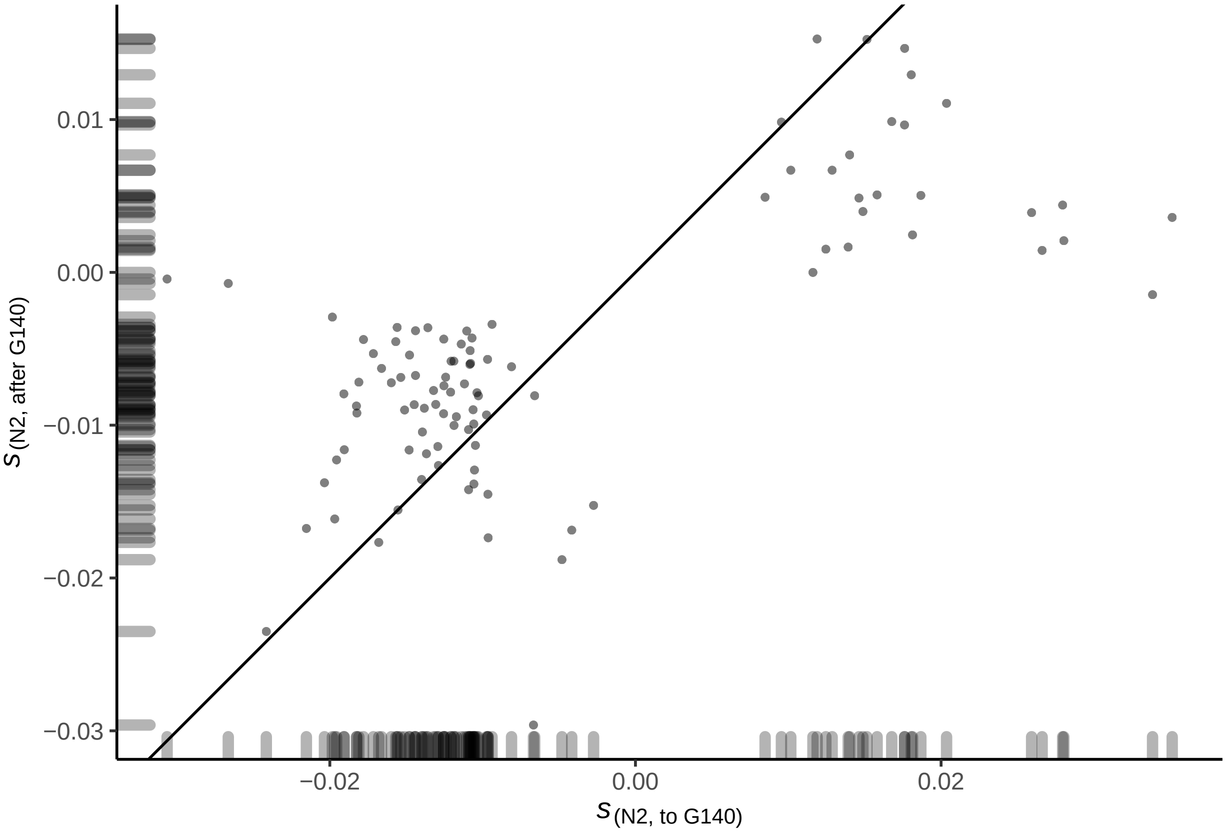
Most of the deterministic allele frequency change happens before G140. So I'll look at s estimated from all data, and from early (G<=140) and late samples (G>140), where n=24 each.

```
# allele frequency trajectories
# again, just taking the site with min p per gwin cM block here
# ignoring for now the fact that SNVs with opposing effects (and -log10p > cut) may
# be found within a block
refafc = reshape2::melt(cbind(pcw[,c('chrom', 'pos', 'g', 'b')], refaf[ix,]),
                        1:4, variable.name = 'pop', value.name = 'N2')
refafc <- merge(refafc, data.frame(pop = popn, gen = gs, rep = reps))
refafc$site = paste(refafc$chrom, refafc$b)
ggplot(refafc, aes(gen, N2, col = g)) +
  geom_line(stat='smooth', aes(group = site), se=F, alpha=0.2, method='loess', span=0.5) +
  theme_classic() + labs(x = 'Generation', y = 'Allele frequency (reference)') +
  scale_color_gradient('GLM -log10(p)', low = 'blue', high = 'red') +
  theme(legend.position = 'top')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# distribution of selection coefficients (min p per bin)
early=140
ss = sapply(ix, function(i) coef(glm(refaf[i,]~gs+reps, family = 'quasibinomial'))[2])
ssEarly = sapply(ix, function(i) coef(glm(refaf[i,gs<=early]~gs[gs<=early]+reps[gs<=early], family = 'qu
ssLate = sapply(ix, function(i) coef(glm(refaf[i,gs>early]~gs[gs>early]+reps[gs>early], family = 'quasib
ss = cbind(snps[ix,], data.frame(s = ss, early = ssEarly, late = ssLate))
ggplot(ss, aes(early, late)) + geom_point(stroke=0, alpha=0.5) +
  geom_abline(aes(intercept=0, slope=1)) +
  theme_classic() + coord_equal() + geom_rug(size=2, alpha=0.3) +
  labs(x = expression(italic(s)["(N2, to G140)"]), y = expression(italic(s)["(N2, after G140)"]))
```



s-1.pdf

Subset the associated SNVs at our threshold, estimate the poolseq selection coefficients and the RIL effects at these SNVs.

```
dump = sprintf("~/Documents/cemee/poolseq/pqa/res_%scM_%2.fp.RData", gwin, cut)
if(file.exists(dump)){
  load(dump, verbose=T)
} else {

  ix = which(padj$g > cut)
  pc = padj[ix,]
  # s all samples: redo pvals using intercept only null
  sfit = mclapply(ix, mc.cores = np, function(i) {
    m0 = glm(cbind(refcadj[i,],altcadj[i,])~reps, family = 'quasibinomial')
    # m0 = glm(cbind(refcadj[i,],altcadj[i,])~1, family = 'quasibinomial')
    m1 = glm(cbind(refcadj[i,],altcadj[i,])~gs+reps, family = 'quasibinomial')
    list(anova(m0, m1, test="LRT"), coef(m1)[2])
  })
  # effects (s) and pvalues
  ss = unlist(lapply(sfit, '[[', 2))
  ss_p = unlist(lapply(sfit, function(x) -log10(x[[1]][2,5])))

  # s early (G0-140)
  six = gs<=early
  sfit = mclapply(ix, mc.cores = np, function(i) {
    m0 = glm(cbind(refcadj[i,six],altcadj[i,six])~reps[six], family = 'quasibinomial')
    m1 = glm(cbind(refcadj[i,six],altcadj[i,six])~gs[six]+reps[six], family = 'quasibinomial')
    list(anova(m0, m1, test="LRT"), coef(m1)[2])
  })
```

```
ssEarly = unlist(lapply(sfit, '[[', 2))
ssEarly_p = unlist(lapply(sfit, function(x) -log10(x[[1]][2,5])))

# s late (G140-)
six = gs>early
sfit = mclapply(ix, mc.cores = np, function(i) {
  m0 = glm(cbind(refcadj[i,six],altcadj[i,six])~reps[six], family = 'quasibinomial')
  m1 = glm(cbind(refcadj[i,six],altcadj[i,six])~gs[six]+reps[six], family = 'quasibinomial')
  list(anova(m0, m1, test="LRT"), coef(m1)[2])
})
ssLate = unlist(lapply(sfit, '[[', 2))
ssLate_p = unlist(lapply(sfit, function(x) -log10(x[[1]][2,5])))

pc = cbind(pc, s = ss, s_p = ss_p,
           s_early = ssEarly, s_early_p = ssEarly_p,
           s_late = ssLate, s_late_p = ssLate_p)

X$chrom = as.character(X$chrom)
Xc = as.data.frame(merge(pc, X, sort=F))
Xcm = t(as.matrix(Xc[,phe$line]))
# remove NAs
nas = apply(Xcm, 2, function(x) sum(is.na(x))>0)
Xcm = Xcm[,!nas]
Xc = Xc[!nas,]
# round intermediate HMM probs
Xcm[Xcm>0 & Xcm<1] = round(Xcm[Xcm>0 & Xcm<1])
cat(sprintf('p %.2f: %s/%s selected sites fixed in phenotyped lines\n', cut, nrow(pc)-nrow(Xc), nrow(p

ixf = paste(snps$chrom, snps$pos) %in% paste(Xc$chrom, Xc$pos)
reff = data.frame(refaf[ixf,])

# get RIL snp effects, against intercept only null
# test all but record the RIL MAF for filtering
fert_fit = GridLMM_GWAS(L1~pop+(1|line), test_formula = ~1, reduced_formula = ~1,
                        data = phe, X = Xcm, X_ID = 'line', relmat = list(line=K),
                        method = 'REML', verbose = F)

# fert_fit$results$beta.lm = unlist(mclapply(1:ncol(Xcm), mc.cores = np, function(i) coef(lm(phe$L1~X
# qplot(fert_fit$results$beta.4, fert_fit$results$beta.lm)
# fixed pop null
# fitp = GridLMM_GWAS(L1~pop+(1|line), test_formula = ~pop, reduced_formula = ~1,
#                     data = phe, X = Xcm, X_ID = 'line', relmat = list(line=K), method = 'REML')


# CF current, G100, G0 allele frequencies
# all reference-based
# `a` is the estimated effect in RILs (the GWAS beta)
res = merge(pc, cbind(Xc[,1:2], a = fert_fit$results$beta.4,
                      pval = -log10(fert_fit$results$p_value_REML),
                      afRIL = 1-(apply(Xc[,phe$line], 1, sum)/nrow(phe)),
                      af = reff$A6140,
                      afA0 = apply(reff[,gs==0], 1, mean),
                      afA100 = reff$A6100,
                        afCA100 = apply(reff[,gs==240], 1, mean)
  ), sort=F)
  # res = merge(pc, cbind(Xc[,1:2], a = fitp$results$beta.2, pval = -log10(fitp$results$p_value_REML.1)


  # correlation between SNP effects on fertility and poolseq selection coef
  # bin by genetic distance
  res$b = round(res$cM*(1/gwin))/(1/gwin)
  res$binf = paste(res$chrom, res$b); len(table(res$binf))

  # fit using RIL A6140/CA* RIL LD matrix between bins
  res_snvs = paste(res$chrom, res$pos)
  snpix = paste(snps$chrom, snps$pos) %in% res_snvs
  snpld = t(as.matrix(rilgt[snpix, grep('^A6|^CA', colnames(rilgt))]))
  # snpld = t(as.matrix(rilgt[snpix, grep('^A6', colnames(rilgt))]))
  colnames(snpld) = paste(res$chrom, res$pos)
  # sites that are invariant in A6140(+CAs), but called in pool
  snpv = apply(snpld, 2, var); sum(snpv==0)
  # downsample if very large
  if(ncol(snpld)>30000) snpv[seq(1, nrow(snpld), 2)]=0
  snpld = cor(snpld[,snpv>0])
  blockld = diag(length(unique(res$binf)))
  rownames(blockld) = colnames(blockld) = unique(res$binf)
  for(i in 1:ncol(blockld)){
    blocki = colnames(blockld)[i]
    for(j in 1:(i-1)){
      blockj = colnames(blockld)[j]
      corij = snpld[colnames(snpld) %in% res_snvs[res$binf==blocki],
                    colnames(snpld) %in% res_snvs[res$binf==blockj]]^2
      # take mean correlation seen between SNPs in blocks
      if(len(corij)==0) {
        # no variable snps in this block
        print(c(i,j))
        blockld[i,j] = blockld[j,i] = NA
      } else {
        # blockld[i,j] = blockld[j,i] = max(corij)
        blockld[i,j] = blockld[j,i] = mean(corij)
      }
    }
  }
  # use global mean for NA (<= 1 marker per bin)
  blockld[is.na(blockld)] = round(mean(ut(blockld), na.rm=T), 5)
  # plot(prcomp(blockld))

  save(res, blockld, gwin, cut, file = dump)
}
```

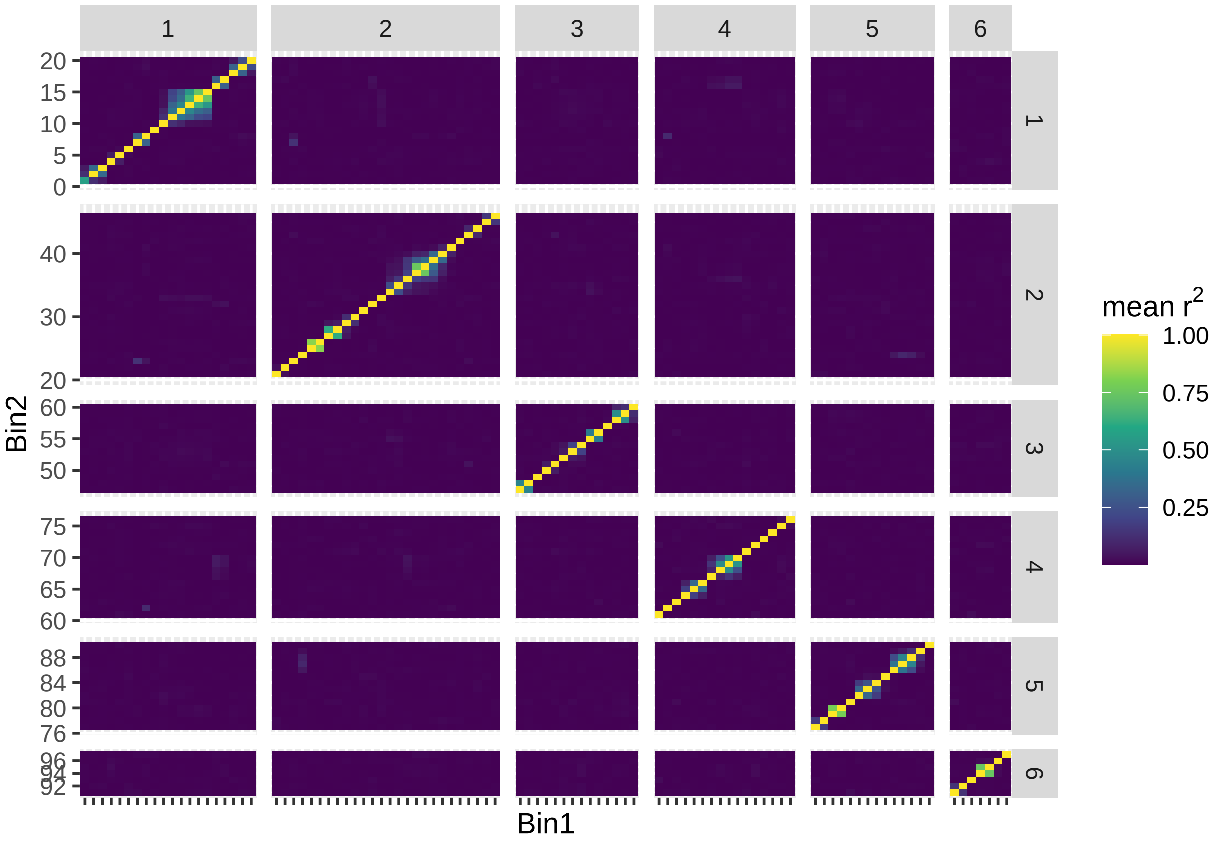
```
## Loading objects:
##   res
##   blockld
##   gwin
##   cut
```

Full LMM at single SNP level, fitting bin as a random effect (I'm not confident this handles pseudoreplication well enough), and testing poolseq selection coefficient overall, early (<=G140) and late (>G140), given that RIL effects are estimated at G140.

```
fit_null = lmer(s~1+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa"))
fit = lmer(s~a+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa"))
fit_earlynull = lmer(s_early~1+(1|binf), res, weights = s_early_p, control = lmerControl(optimizer ="bo
fit_early = lmer(s_early~a+(1|binf), res, weights = s_early_p, control = lmerControl(optimizer ="bobyqa
fit_latenull = lmer(s_late~1+(1|binf), res, weights = s_late_p, control = lmerControl(optimizer ="bobyq
fit_late = lmer(s_late~a+(1|binf), res, weights = s_late_p, control = lmerControl(optimizer ="bobyqa"))

lls = sapply(list(fit_null, fit, fit_earlynull, fit_early, fit_latenull, fit_late), function(x) summary
# difference in log likelihood for overall, early, late selection coefs
# against each (intercept only) null
print(data.frame(s=c('overall', 'early', 'late'), LR=round(lls[seq(2, 6, 2)]-lls[seq(1, 6, 2)],1)))
```

```
##         s    LR
## 1 overall 903.2
## 2   early 848.0
## 3    late  67.0
```

```
blockr = reshape2::melt(blockld)
blockr$y = factorToInt(factor(blockr$Var2, labels = 1:length(levels(blockr$Var2))))
blockr$chrom1 = tstrsplit(blockr$Var1, " ")[[1]]
blockr$chrom2 = tstrsplit(blockr$Var2, " ")[[1]]

ggplot(blockr, aes(Var1, y, fill=value)) + geom_tile() +
  scale_fill_viridis_c(expression(paste(mean~r^2))) +
  facet_grid(chrom2~chrom1, scales='free', space='free') +
  theme(axis.text.x = element_blank()) + labs(x='Bin1', y='Bin2')
```



```
# slow: 20 minutes per model at ~100 bins/4k markers
# ldfit <- mmer(s~a, random=~vs(binf, Gu=blockld), rcov=~units, data = res, tolparinv = 1e-4, method='A
# ldfit_null <- mmer(s~1, random=~vs(binf, Gu=blockld), rcov=~units, data = res, tolparinv = 1e-4, meth
# ldfit_early <- mmer(s_early~a, random=~vs(binf, Gu=blockld), rcov=~units, data = res, tolparinv = 1e-
# ldfit_late <- mmer(s_late~a, random=~vs(binf, Gu=blockld), rcov=~units, data = res, tolparinv = 1e-4,

# there's still a strong positive association between RIL and poolseq estimated additive effects.
# and not much different to that ignoring LD
# at p<10-35, gwin=0.5: t=44, vs t=48 for full glm

# summary(fit)
# summary(ldfit)
# 0.5cm bin, cut=p<10^-35, mean LD
# ============================================================
#        logLik       AIC       BIC Method Converge
# Value 872.504 -1741.008 -1728.349     NR     TRUE
# ============================================================
# Variance-Covariance components:
#             VarComp VarCompSE Zratio Constraint
# u:binf.s-s 0.0001096 1.691e-05  6.477   Positive
# units.s-s  0.0000284 6.313e-07 44.989   Positive
# ============================================================
# Fixed effects:
#   Trait      Effect  Estimate Std.Error t.value
# 1     s (Intercept) -0.005341 0.0013384  -3.991
# 2     s           a  0.036949 0.0008373  44.129
# ============================================================
# Groups and observations:
#          s
# u:binf 97
# ============================================================
# binned LD ~ 79% of the variance in s
# ldfit$sigma$`u:binf`/sum(unlist(ldfit$sigma))
```

2: change in allele frequency as a function of estimated SNV effect

$\sigma^2\alpha = d_w/d_t = 2pq(p\alpha + q\alpha)^2$ (Robertson) $= 2\alpha^2 pq$ (Barton)
where α = average effect (difference in fitness of a single allele copy, $wAA - wAa$) $= \frac{pqs^2}{2}$

if $dt = 1 : s^2 = (2 * \sigma^2\alpha)/pq$

```
# A6140 reference
res$p = res$af
res$ssq = (2*res$p*(1-res$p)) * (res$a*res$p + res$a*(1-res$p))^2
# A0 reference
res$p = res$afA0
res$ssqA0 = (2*res$p*(1-res$p)) * (res$a*res$p + res$a*(1-res$p))^2
# A6100 reference
res$p = res$afA100
res$ssqA100 = (2*res$p*(1-res$p)) * (res$a*res$p + res$a*(1-res$p))^2
# CA*100 reference
res$p = res$afCA100
res$ssqCA100 = (2*res$p*(1-res$p)) * (res$a*res$p + res$a*(1-res$p))^2
```

For speed, ignore LD between genetic bins here (results are not much inflated above).

Does including allele frequency change the already strong association between RIL a and poolseq s?

Fit LMMs to each of the above timepoints (A0, A6100, A6140, CA100), asking if the inclusion of $\sigma^2\alpha$ improves fit to a model regressing s on a.

Actually, this doesn't make sense. We know s is correlated with α, so this, below, is just testing whether allele frequency is also related to s, which it is by definition since we won't detect anything that doesn't change in frequency. So what is it adding? I must be doing/thinking it wrong.

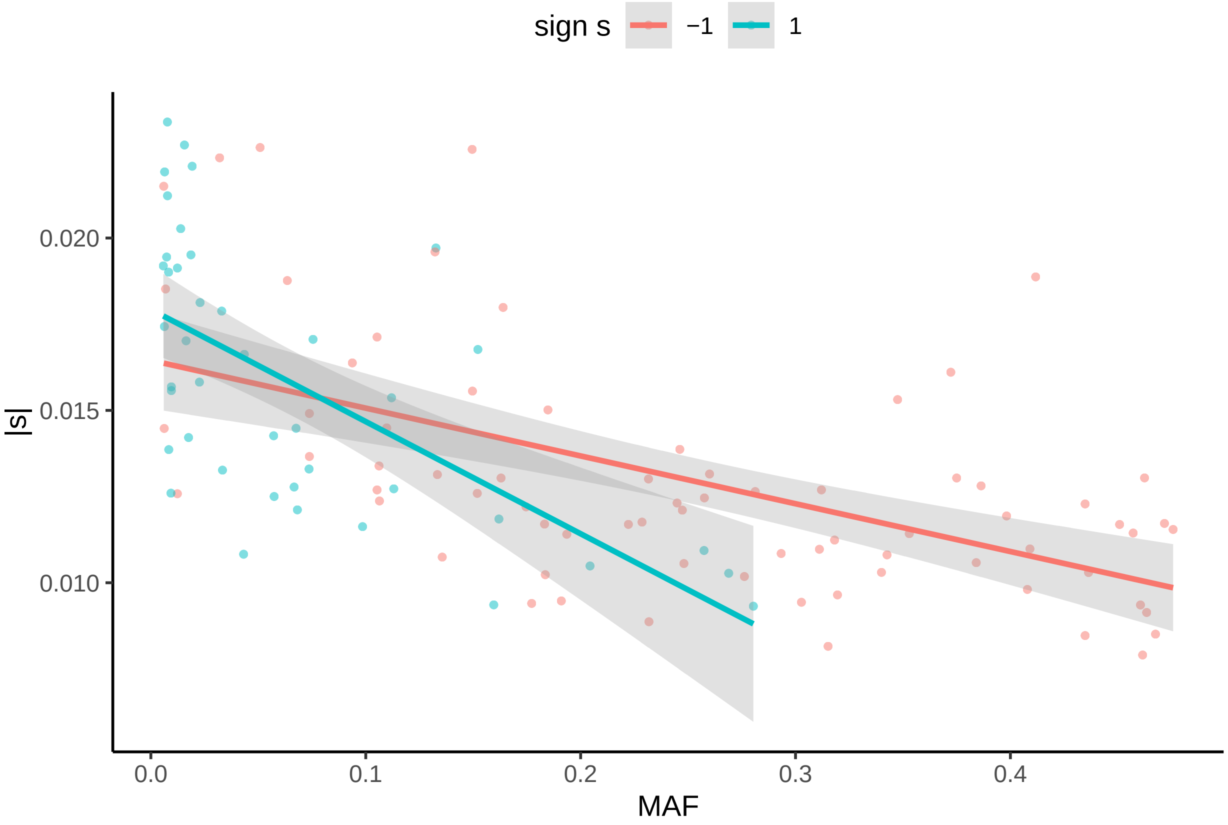
```
pqa_null = lmer(s~a+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa"), REML = F)
pqa_fit_A0 = lmer(s~a+ssqA0+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa"), R
pqa_fit_A6100 = lmer(s~a+ssqA100+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa
pqa_fit_A6140 = lmer(s~a+ssq+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyqa"),
pqa_fit_CA100 = lmer(s~a+ssqCA100+(1|binf), res, weights = s_p, control = lmerControl(optimizer ="bobyq

# s (all samples) v MAF (A6140)
binmax = merge(res, aggregate(data = res, g~binf+sign(s), max))
binmax$maf = afsToMafs(binmax$af)
```

And here is the bin level correlations between s and minor allele frequency. Correlation between $|s|$ and MAF is r =-0.6262294 overall.

But note that low-frequency stuff will be estimated very unreliably in the RILs.

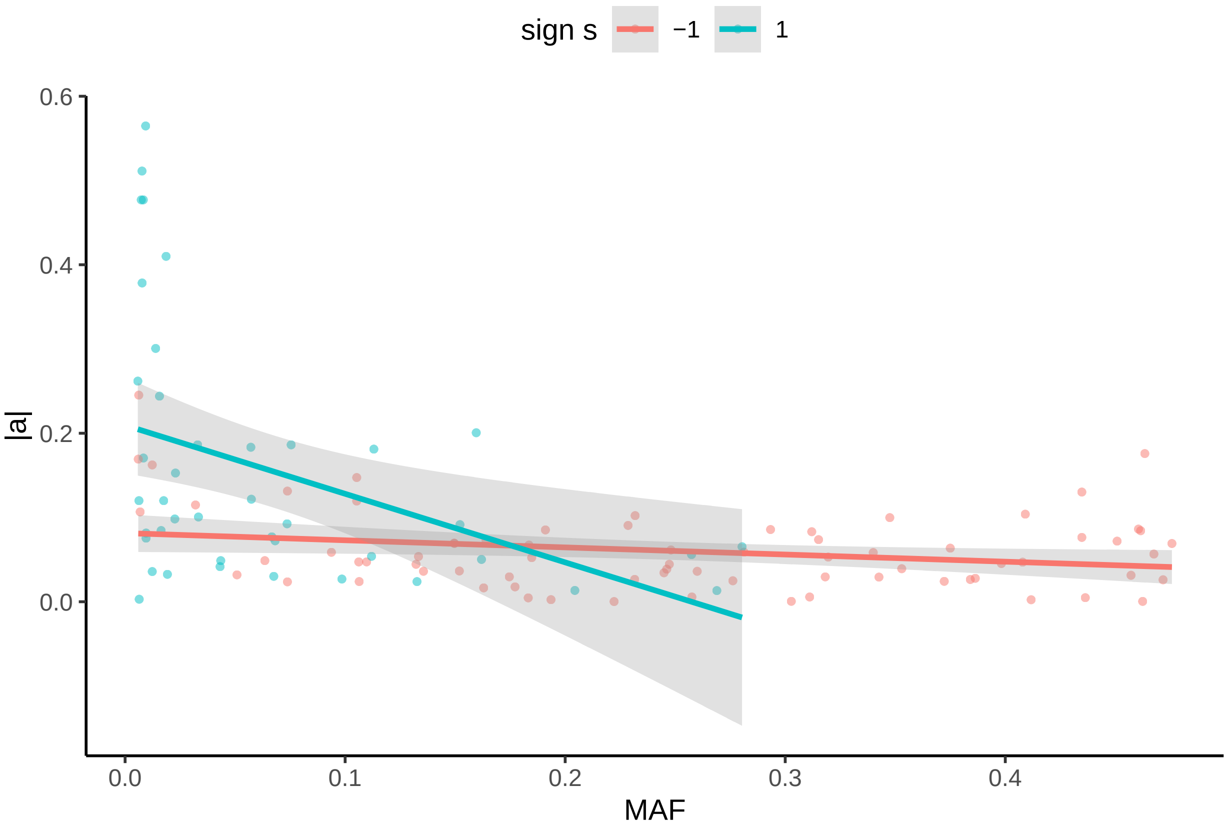
```
ggplot(binmax, aes(maf, abs(s), col = factor(sign(s)))) + geom_point(stroke=0, alpha=0.5) +
  theme_classic() + labs(x='MAF', y = "|s|") + scale_color_discrete("sign s") +
  theme(legend.position = 'top')  + geom_smooth(method='lm', alpha=0.3)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
ggplot(binmax, aes(maf, abs(a), col = factor(sign(s)))) + geom_point(stroke=0, alpha=0.5) +
  theme_classic() + labs(x='MAF', y = "|a|") + scale_color_discrete("sign s") +
  theme(legend.position = 'top')  + geom_smooth(method='lm', alpha=0.3)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Model fits are all better than the α null, of course.

And the model coefficients for $\sigma^2\alpha$ increase with time.

```
data.frame(gen=c(0, 100, 140, 240),
           d_aic=unlist(lapply(list(pqa_fit_A0, pqa_fit_A6100, pqa_fit_A6140, pqa_fit_CA100), function(
           coef_ssq=unlist(lapply(list(pqa_fit_A0, pqa_fit_A6100, pqa_fit_A6140, pqa_fit_CA100), functi
)
```

```
##   gen      d_aic  coef_ssq
## 1   0 -928.30583 0.2966063
## 2 100 -416.46610 0.4815856
## 3 140  -81.35203 0.5409662
## 4 240 -129.22004 0.6864669
```

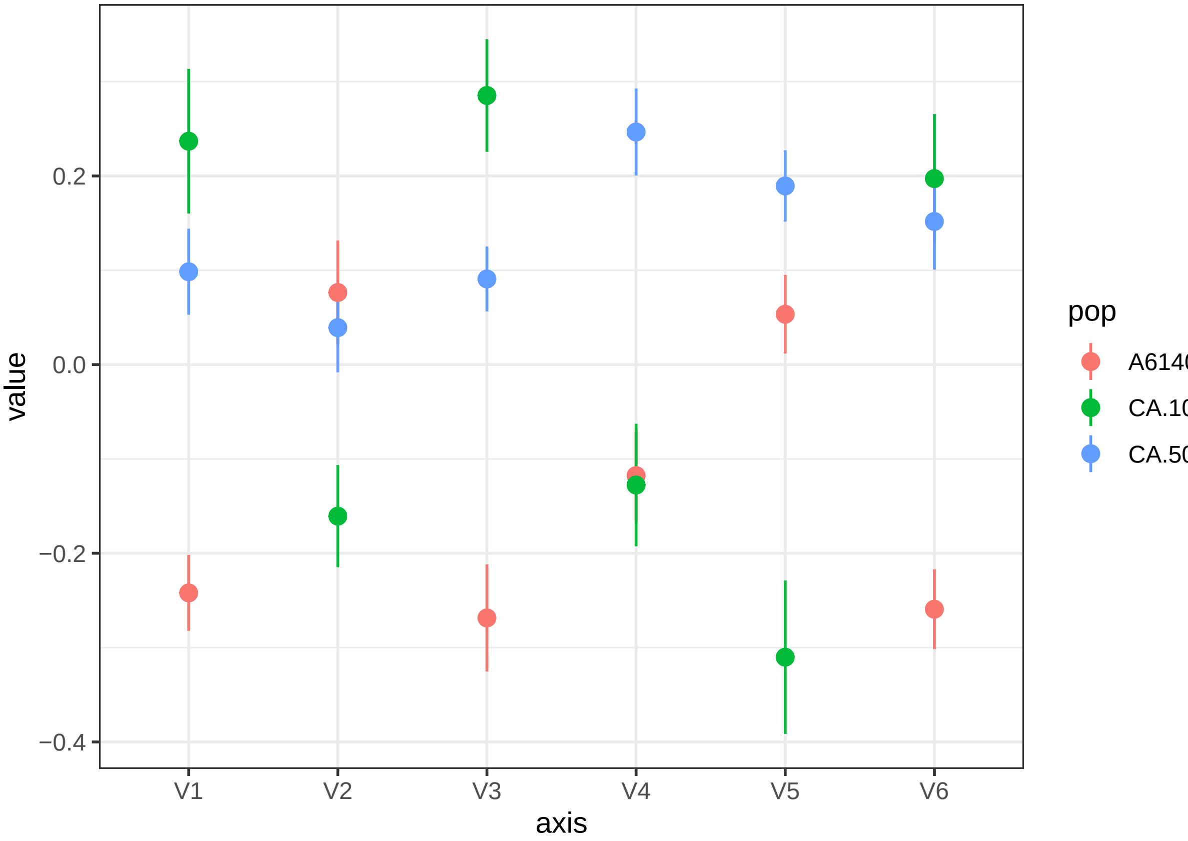
```

gamma_dump = "~/Documents/cemee/poolseq/pqa/gamma_gwas_data.RData"
if(!file.exists(gamma_dump)){

  gammas <- fread('~/Downloads/gamma.txt')
  M=eigen(gammas)$vector
  load('~/Documents/cemee/phenotypes/transitionRates.rda', verbose = T)
  traits = c('sf', 'sb', 'fs', 'fb', 'bs', 'bf')
  ngm <- data.frame(subset(mov, env=='NGM' & line %in% colnames(rilgt)))
  y = t(apply(as.matrix(ngm[,traits]), 1, function(x) t(M) %*% x))
  y = scale(y)
  d = mahalanobis(y, center = F, cov = cov(y))
  qplot(d, -log10(dchisq(d, df = 6)))
  # drop 3 A6140 outliers
  ngm <- subset(ngm, ! line %in% ngm$line[d > 70])
  ngm = ngm[grep('GA', ngm$line, invert = T),]
  y = t(apply(as.matrix(ngm[,traits]), 1, function(x) t(M) %*% x))
  # note: scaling traits to unit variance
  y_df = cbind(line = ngm$line,
               pop = cemeePopsFromLines(ngm$line),
               popr = cemeePopRepsFromLines(ngm$line),
               date = ngm$date,
               as.data.frame(scale(y)))
  y_df = y_df[order(y_df$line, y_df$date),]
  Xgamma = filterMAF(cbind(rilsnps[,1:2], rilgt), MAFgt = 0, header = 2, lines=unique(ngm$line))
  Xld_gamma = doPrune(Xgamma[,1:2], Xgamma[,-(1:2)], r2 = 0.99, np=6, window=4000, step=2500)
  K_gamma = hgsm(Xld_gamma[[2]])
  # check snps == X
  snps_gamma = rilsnps[paste(rilsnps$chrom, rilsnps$pos) %in% paste(Xld_gamma[[1]]$chrom,Xld_gamma[[1]]$
  Xld_gamma=t(Xld_gamma[[2]])
  # remove NAs
  nas = apply(Xld_gamma, 2, function(x) sum(is.na(x))>0)
  Xld_gamma = Xld_gamma[,!nas]
  snps_gamma = snps_gamma[!nas,]
  # round intermediate HMM probs
  Xld_gamma[Xld_gamma>0 & Xld_gamma<1] = round(Xld_gamma[Xld_gamma>0 & Xld_gamma<1])

  # get RIL snp effects, against intercept + pop replicate null
  # nope: intercept only
  # test all but record the RIL MAF for filtering
  # greene! or drop popr
  gamma_fits = do.call(rbind, lapply(1:6, function(i) {
    # fiti = GridLMM_GWAS(as.formula(sprintf("V%s~popr+(1|line)", i)),
    #                     test_formula = ~1+popr, reduced_formula = ~1,
    #                     data = y_df, X = Xld_gamma,
    #                     X_ID = 'line', relmat = list(line=K_gamma),
    #                     method = 'REML', verbose = T)
    fiti = GridLMM_GWAS(as.formula(sprintf("V%s~1+(1|line)", i)),
                        test_formula = ~1, reduced_formula = ~1,
                        data = y_df, X = Xld_gamma,
                        X_ID = 'line', relmat = list(line=K_gamma),
                        method = 'REML', verbose = T)
    cbind(snps_gamma[,-(3:4)], data.frame(dim=i, a = fiti$results$beta.2,
                                          pval = -log10(fiti$results$p_value_REML)))
  }))

  save(snps_gamma, K_gamma, Xld_gamma, y_df, gamma_fits, file = gamma_dump)
} else {
  load(gamma_dump, verbose=T)
}
```

selection axes.

```
## Loading objects:
##   snps_gamma
##   K_gamma
##   Xld_gamma
##   y_df
##   gamma_fits
```

```
# mean trait values by generation: there's strong differentiation
# for the GWAS, I've ignored this (intercept only null)
ym = as.data.frame(melt(as.data.table(y_df), 1:4, variable.name='axis'))
ggplot(ym, aes(axis, value, col = pop)) + stat_summary() + theme_bw()
```

```
## No summary function supplied, defaulting to `mean_se()`
```



munge and gwas-1.pdf

```
# repeatabilities for lines and populations
lapply(split(ym, ym$axis), function(x) rpt(value~1+(1|line)+(1|popr), grname = c('line', 'popr'), data
```

```
## $V1
##       line       popr
## 1 0.403872 0.03127882
##
## $V2
##        line       popr
## 1 0.3384731 0.01178134
##
## $V3
##        line       popr
## 1 0.2670656 0.06067762
##
## $V4
##        line       popr
## 1 0.3635541 0.04981233
##
```

```
## $V5
##        line       popr
## 1 0.2032516 0.06143917
##
## $V6
##       line       popr
## 1 0.454038 0.06778049
```

```
# correlation between SNP effects on fertility and poolseq selection coef
# bin by genetic distance
gamma_fits$b = round(gamma_fits$cM*(1/gwin))/(1/gwin)
gamma_fits$binf = paste(gamma_fits$chrom, gamma_fits$b); len(table(gamma_fits$binf))
```

```
## [1] 604
```

```
# subset to thresholded poolseq tests
gamma_fits$cut = paste(gamma_fits$chrom, gamma_fits$pos) %in% paste(res$chrom, res$pos)
gamma_s = merge(as.data.frame(subset(gamma_fits, cut)), res[,c('chrom', 'pos', 'g', 's', 's_p', 's_early

p = gamma_s$af
gamma_s$ssq = (2*p*(1-p)) * (gamma_s$a*p + gamma_s$a*(1-p))^2
# A0 reference
p = gamma_s$afA0
gamma_s$ssqA0 = (2*p*(1-p)) * (gamma_s$a*p + gamma_s$a*(1-p))^2
# A6100 reference
p = gamma_s$afA100
gamma_s$ssqA100 = (2*p*(1-p)) * (gamma_s$a*p + gamma_s$a*(1-p))^2
# CA*100 reference
p = gamma_s$afCA100
gamma_s$ssqCA100 = (2*p*(1-p)) * (gamma_s$a*p + gamma_s$a*(1-p))^2
```

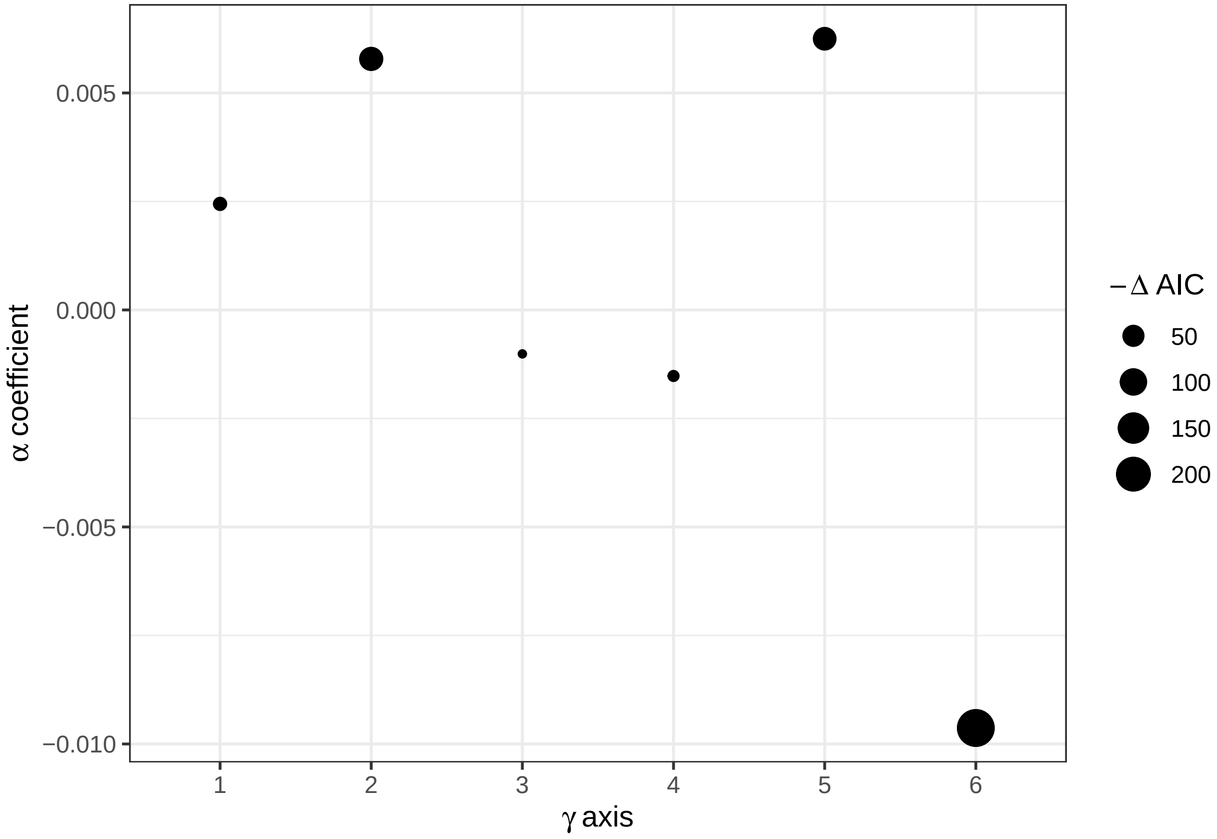
```
# fits by trait, SNP-level
# inclusion of sigma^2a over a alone
axis_fits_ssq = do.call(rbind, lapply(split(gamma_s, gamma_s$dim), function(x){

  pqa_null = lmer(s~a+(1|binf), x, weights = s_p,
                  control = lmerControl(optimizer ="bobyqa"), REML = F)
  pqa_fit_A0 = lmer(s~a+ssqA0+(1|binf), x, weights = s_p,
                    control = lmerControl(optimizer ="bobyqa"), REML = F)
  pqa_fit_A6100 = lmer(s~a+ssqA100+(1|binf), x, weights = s_p,
                       control = lmerControl(optimizer ="bobyqa"), REML = F)
  pqa_fit_A6140 = lmer(s~a+ssq+(1|binf), x, weights = s_p,
                       control = lmerControl(optimizer ="bobyqa"), REML = F)
  pqa_fit_CA100 = lmer(s~a+ssqCA100+(1|binf), x, weights = s_p,
                     control = lmerControl(optimizer ="bobyqa"), REML = F)

  data.frame(dim = x$dim[1], gen=c(0, 100, 140, 240),
             d_aic=unlist(lapply(list(pqa_fit_A0, pqa_fit_A6100, pqa_fit_A6140, pqa_fit_CA100), functio
             coef_ssq=unlist(lapply(list(pqa_fit_A0, pqa_fit_A6100, pqa_fit_A6140, pqa_fit_CA100), func
  )

}))

# inclusion of a alone
axis_fits_a = do.call(rbind, lapply(split(gamma_s, gamma_s$dim), function(x){
  pqa_null = lmer(s~1+(1|binf), x,  REML = F)
  pqa_fit = lmer(s~a+(1|binf), x, REML = F)
  data.frame(dim = x$dim[1], d_aic = diff(anova(pqa_null, pqa_fit, test="LRT")$AIC),
             coef_a=coef(summary(pqa_fit))[2,1])
}))

# bin level
# s (all samples) v MAF (A6140)
binmax = merge(gamma_s, aggregate(data = gamma_s, g~binf+sign(s)+dim, max))
```

The correlations between s and α vary with γ dimension. At the bin level (conservative): -0.2287243, 0.3001934, -0.3577318, -0.1976464, 0.0451889, -0.4283464.

In the plot below (SNP level regression on 3109 markers thresholded by s p-value, with a random bin effect), all axes but 3 and 4 show some association ($\Delta AIC > 3$), with axis 6 under stabilising selection by far the strongest!

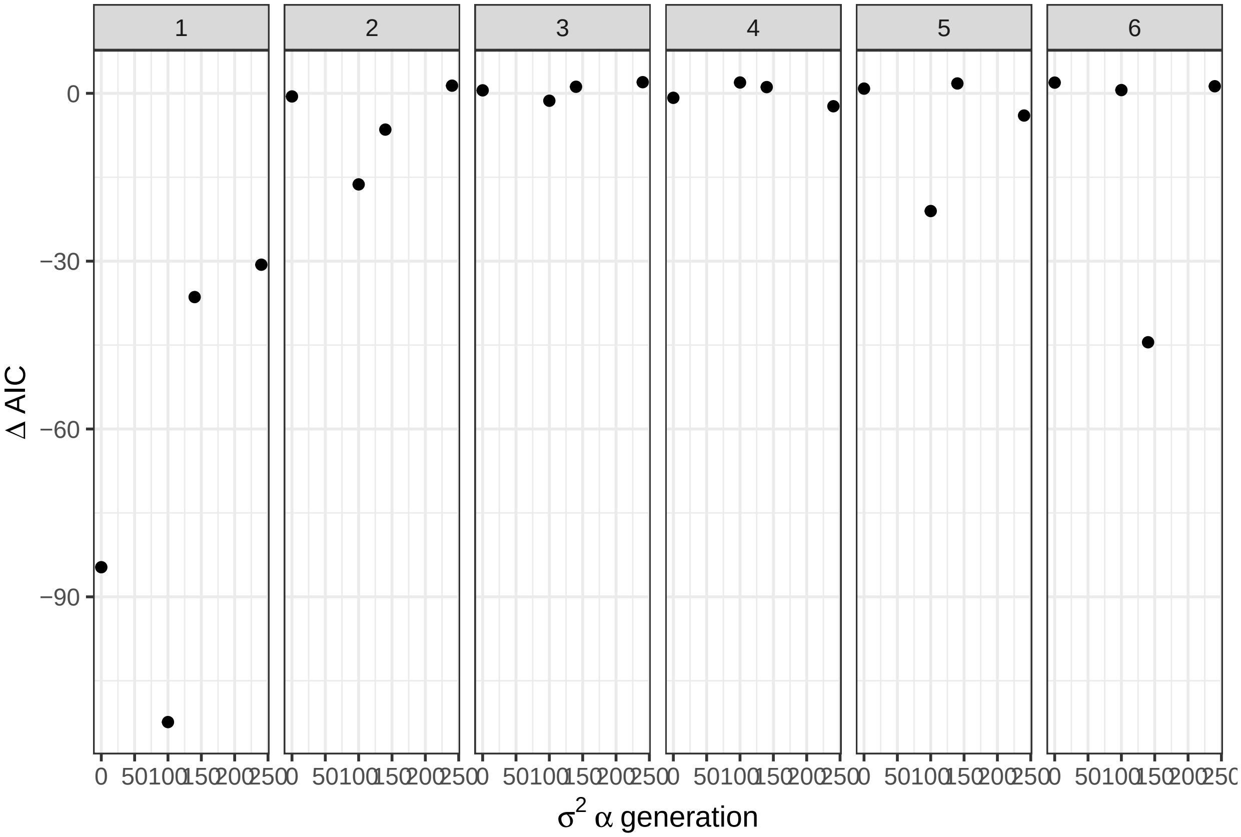
Inclusion of $\sigma^2\alpha$ rarely improves the model fit over α alone, except for the first dimension. A6100 allele frequency seems to be the most correlated.

```
# not traits were variance standardised, so the coefficients are relative
ggplot(axis_fits_a, aes(factor(dim), coef_a, size=-d_aic)) + geom_point() + theme_bw() +
  labs(x = expression(paste(gamma~'axis')), y = expression(paste(alpha~'coefficient'))) +
  scale_size(expression(paste(-Delta~"AIC")))
```



pqa fits-1.pdf

```
ggplot(axis_fits_ssq, aes(gen, d_aic)) + geom_point() + facet_grid(.~dim) + theme_bw() +
  labs(x = expression(paste(sigma^2~alpha~'generation')), y = expression(paste(Delta~"AIC")))
```



pqa fits-2.pdf

3: GxGxE

The above is all on additive effects, but we know that there's abundant epistasis for fertility (including between alleles with weak additive effects).

The amount of negative epistasis for new mutations is an important quantity for evolution. We can make the case that moving to a new environment (NGMN-adapted RILs tested on NaCl) is similar to (multiple) new mutations.

Below, we test for directional epistasis between all markers with additive effects (again, at a given threshold, and using either RIL a or poolseq s).

Connection between above and below needs work!

```
# using BLUPs in NGM (quasipoisson) and NaCl (negative binomial)
load('~/Documents/cemee/phenotypes/fertility.rda', verbose = T)
```

```
## Loading objects:
##   ecoefs
##   feno
```

```
fert = ecoefs[order(ecoefs$line),]
fert = subset(fert, line %in% colnames(X))

# this includes 58 G* RILs as well as 160 A6140
# confirm the below holds for A6140 only
table(cemeePopRepsFromLines(unique(fert$line)))
```

```
##
```

```
## A6140 GA150 GA250 GT150 GT250
##   160    26    20     7     5
```

```
ldsnps = Xld[[1]]
ldsnps$cM = X$cM[paste(X$chrom, X$pos) %in% paste(ldsnps$chrom, ldsnps$pos)]
Xt = t(Xld[[2]])
# remove NAs
nas = apply(Xt, 2, function(x) sum(is.na(x))>0)
Xt = Xt[,!nas]
ldsnps = ldsnps[!nas,]
# convert intermediate HMM probs
Xt[Xt>0 & Xt<1] = round(Xt[Xt>0 & Xt<1])

doGWAS = function(){
  # estimate RIL SNP effects on NGM and NaCl
  # using MAF>5% and minimally pruned of LD (r2<0.99)

  # exclude markers within 1cM of the focal marker from the GRM
  prox = unlist(parallel::mclapply(split(cbind(ix=1:ncol(Xt), ldsnps), ldsnps$chrom), mc.cores = np, fu

  # snp h2 = 0.3
  fitngm <- GridLMM_GWAS(fertility~1 + (1|line), test_formula = ~1, reduced_formula = ~1,
                         data=subset(fert, env=='NGM'), X = Xt, X_ID = 'line',
                         relmat = list(line=K), proximal_markers = prox)

  # snp h2 = 0.19
  fitnacl <- GridLMM_GWAS(fertility~1 + (1|line), test_formula = ~1, reduced_formula = ~1,
                          data=subset(fert, env!='NGM'), X = Xt, X_ID = 'line',
                          relmat = list(line=K), proximal_markers = prox)
  # snp h2 = 0.69
  # fitngm_a <- GridLMM_GWAS(fertility~1 + (1|line), test_formula = ~1, reduced_formula = ~1,
  #                          data=subset(fert, env=="NGM"), X = Xt, X_ID = 'line',
  #                          relmat = list(line=K^2))
  # snp h2 = 1
  # fitnacl_a <- GridLMM_GWAS(fertility~1 + (1|line), test_formula = ~1, reduced_formula = ~1,
  #                           data=subset(fert, env!="NGM"), X = Xt, X_ID = 'line',
  #                           relmat = list(line=K^2))

  ril_beta = rbind(cbind(ldsnps, beta = fitngm$results$beta.2, p = fitngm$results$p_value_REML, env = "
                   cbind(ldsnps, beta = fitnacl$results$beta.2, p = fitnacl$results$p_value_REML, env =

  ggplot(subset(ril_beta, p<0.1), aes(pos/1e6, -log10(p))) + geom_point(alpha=0.5, stroke=F) +
    facet_grid(env~chrom, scales = 'free_x') + theme_classic()
  save(ril_beta, file = '~/Documents/cemee/qtl/cemee_v2_fertility_gwas_ld.RData')
}

load('~/Documents/cemee/qtl/cemee_v2_fertility_gwas_ld.RData', verbose=T)
```

```
## Loading objects:
##   ril_beta
```

The function `syne` below test all pairwise interactions between a set of diallelic markers at a given threshold in a given environment. As before, to approximately account for LD we take the smallest p-value in bin=0.5 cM blocks (i.e., ignoring the mean additive effect within a bin).

I split effects by effect direction for reference alleles and alternate alleles, which might differ due to lab adaptation, or because this correlates with allele frequency. So in the below **ref** means QTL with effects where the reference allele decreases fertility, **alt** means the non-N2 allele decreases fertility.

Still need to compare with QTL from poolseq.

```
syne <- function(ldsnps, ril_beta, fert, Xt, thresh=0.01, bin=0.5, minclass=4, np=6, ev='NGM'){

  # return: interaction coefficients, separately by direction of (reference-based)
  # additive effect (in case commoner N2 alleles are doing something different to rare wild alleles)

  fertd = subset(fert, env==ev)
  fertd = fertd[order(fertd$line),]

  # alt deleterious
  sub = subset(ril_beta, p<thresh & beta<0 & env==ev)
  sub$b = round(sub$cM*(1/bin))/(1/bin)
  subdel = merge(sub, aggregate(data = sub, p~b+chrom, min))
  df = subdel[!duplicated(subdel[,c('chrom','p', 'b')]),]
  ix = which(paste(ldsnps$chrom, ldsnps$pos) %in% paste(df$chrom, df$pos))
  snps_ = ldsnps[ix,]
  X_ = Xt[,ix]
  pw = expand.grid(1:nrow(df), 1:nrow(df))
  pw = pw[pw$Var1!=pw$Var2,]
  altd = do.call(rbind, mclapply(1:nrow(pw), mc.cores=np, function(x) {
    i = pw$Var1[x]
    j = pw$Var2[x]
    ix = which((snps_$chrom==df$chrom[i] & snps_$pos==df$pos[i])|(snps_$chrom==df$chrom[j] &
                                                                    snps_$pos==df$pos[j]))
    dx = cbind(data.frame(X_[,ix]), y=fertd$fertility)
    dx = dx[dx$X1%%1==0 & dx$X2%%1==0,]
    if((prod(dim(table(dx[,-3])))==4) & (min(table(dx[,-3]))>=minclass)){
      # ggplot(dx, aes(X1, y, col=factor(X2))) + stat_summary() +
      #  geom_smooth(method='lm', se=F)
      fit = lm(y~X1*X2, dx)
      fit0 = lm(y~X1+X2, dx)
      data.frame(r_full = summary(fit)$adj,
                 r_add = summary(fit0)$adj,
                 coef = coef(fit)[4])
    } else {
      data.frame(r_full = NA, r_add = NA, coef = NA)
    }
  }))
  altd = cbind(pw, altd)
  alt = df

  # N2 del
  sub = subset(ril_beta, p<thresh & beta>0 & env==ev)
  sub$b = round(sub$cM*(1/bin))/(1/bin)
  subdel = merge(sub, aggregate(data = sub, p~b+chrom, min))
  df = subdel[!duplicated(subdel[,c('chrom','p', 'b')]),]
  ix = which(paste(ldsnps$chrom, ldsnps$pos) %in% paste(df$chrom, df$pos))
  snps_ = ldsnps[ix,]
  X_ = Xt[,ix]
  pw = expand.grid(1:nrow(df), 1:nrow(df))
  pw = pw[pw$Var1!=pw$Var2,]
```

```r
  refd = do.call(rbind, mclapply(1:nrow(pw), mc.cores=np, function(x) {
    i = pw$Var1[x]
    j = pw$Var2[x]
    ix = which((snps_$chrom==df$chrom[i] & snps_$pos==df$pos[i])|(snps_$chrom==df$chrom[j] &
                                                                  snps_$pos==df$pos[j]))
    dx = cbind(data.frame(X_[,ix]), y=fertd$fertility)
    dx = dx[dx$X1%%1==0 & dx$X2%%1==0,]
    if((prod(dim(table(dx[,-3])))==4) & (min(table(dx[,-3]))>=minclass)){
      # ggplot(dx, aes(X1, y, col=factor(X2))) + stat_summary() +
      #  geom_smooth(method='lm', se=F)
      fit = lm(y~X1*X2, dx)
      fit0 = lm(y~X1+X2, dx)
      data.frame(r_full = summary(fit)$adj,
                 r_add = summary(fit0)$adj,
                 coef = coef(fit)[4])
    } else {
      data.frame(r_full = NA, r_add = NA, coef = NA)
    }
  }))
  refd = cbind(pw, refd)
  ref = df

  o = list(alt = list(cbind(altd, env=ev, cut=thresh, pol='alt'), alt),
           ref = list(cbind(refd, env=ev, cut=thresh, pol='ref'), ref))


  d = rbind(o$alt[[1]], o$ref[[1]])
  pd <- ggplot(d, aes(coef, r_full-r_add, col = factor(env))) + geom_point()
  pmarg = ggMarginal(pd, type="violin", margins='x', draw_quantiles = c(0.25, 0.5, 0.75))
  o = c(o, pmarg)
  o
}
```

In general:

– interactions between additive QTL are clearly negative on average – interactions between QTL (in RILs, for that environment) are much more negative on NaCl (and vary more) – interactions between non-reference alleles on NaCl are the most negative/variable of all (though there are many more than for reference alleles).

```r
# look at three thresholds, across NGM and NaCl

syn_eff = list(syne(ldsnps, ril_beta, fert, Xt, thresh = 0.001),
               syne(ldsnps, ril_beta, fert, Xt, thresh = 0.005),
               syne(ldsnps, ril_beta, fert, Xt, thresh = 0.01),
               syne(ldsnps, ril_beta, fert, Xt, thresh = 0.001, ev = "NaCl"),
               syne(ldsnps, ril_beta, fert, Xt, thresh = 0.005, ev = "NaCl"),
               syne(ldsnps, ril_beta, fert, Xt, thresh = 0.01, ev = "NaCl"))
```

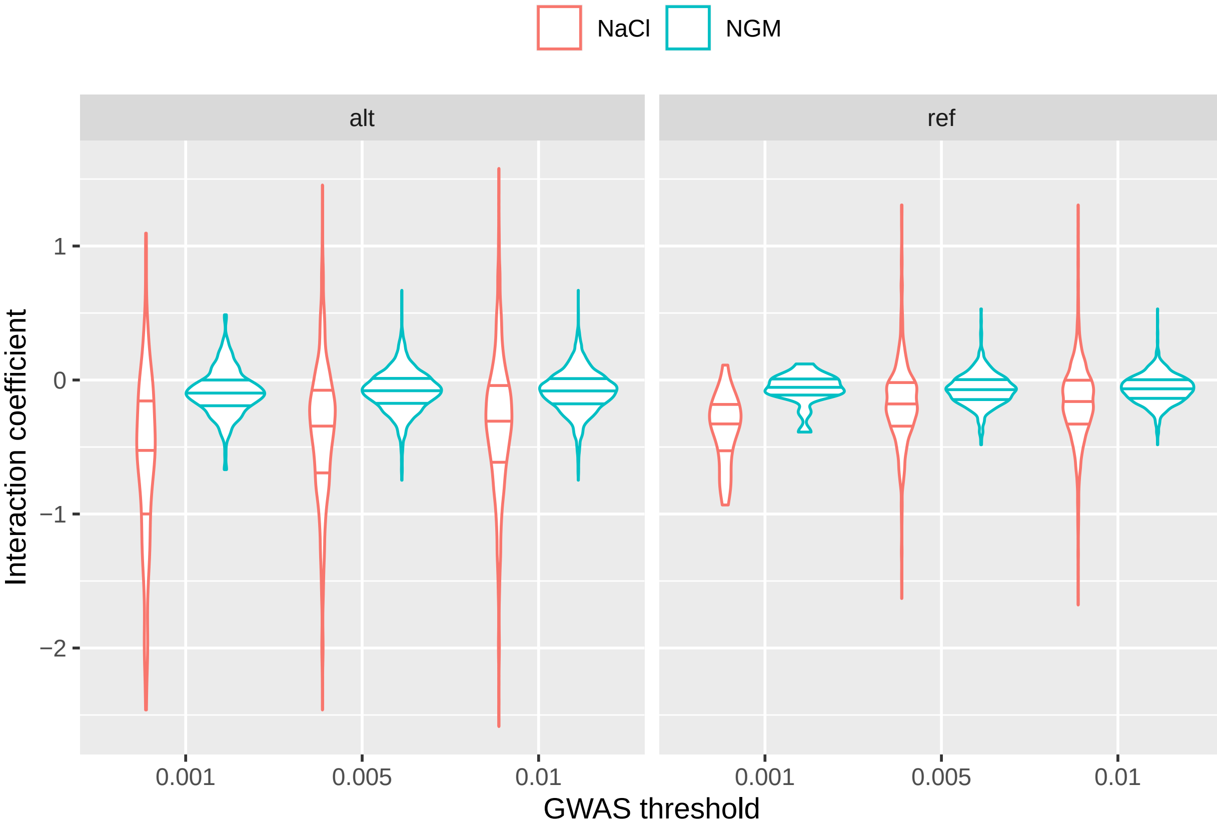
```
## Warning: Removed 482 rows containing missing values (geom_point).

## Warning: Removed 5064 rows containing missing values (geom_point).

## Warning: Removed 11088 rows containing missing values (geom_point).

## Warning: Removed 1064 rows containing missing values (geom_point).

## Warning: Removed 5694 rows containing missing values (geom_point).
## Warning: Removed 11598 rows containing missing values (geom_point).
```

```
effs = do.call(rbind, lapply(syn_eff, function(x) rbind(x$alt[[1]], x$ref[[1]])))

# number of bins tested per threshold/env
do.call(rbind, lapply(syn_eff, function(x) data.frame(n_alt=nrow(x$alt[[2]]),
                                                      n_ref=nrow(x$ref[[2]]),
                                                      env=x$alt[[1]]$env[1],
                                                      threshold=x$alt[[1]]$cut[1])))
```

```
##   n_alt n_ref  env threshold
## 1    32     7  NGM     0.001
## 2   100    39  NGM     0.005
## 3   143    64  NGM     0.010
## 4    39     8 NaCl     0.001
## 5    94    47 NaCl     0.005
## 6   130    77 NaCl     0.010
```

```
ggplot(effs, aes(factor(cut), coef, col = env)) +
  geom_violin(draw_quantiles = c(0.25, 0.5, 0.75)) +
  labs(x = 'GWAS threshold', y = 'Interaction coefficient') + theme(legend.position = 'top') +
  scale_color_discrete("") + facet_grid(.~pol)
```

```
## Warning: Removed 34990 rows containing non-finite values (stat_ydensity).
```



distributions-1.pdf

If we want to look at change in fitness, we're going to need to revert to using worm size. We can at least justify it directly as a good proxy in the A6140, and then indirectly based on the increase in size across populations and derived RILs.

```
load('~/Documents/cemee/phenotypes/CeMEE_simpleLocomotionTraits_correctedPlateMeansERes.RData', verbose
```

```
## Loading objects:
##   dfoo
##   dfov
##   dfoorp
##   dfoory
##   dfoorvp
##   dfoorvy
##   traito
```

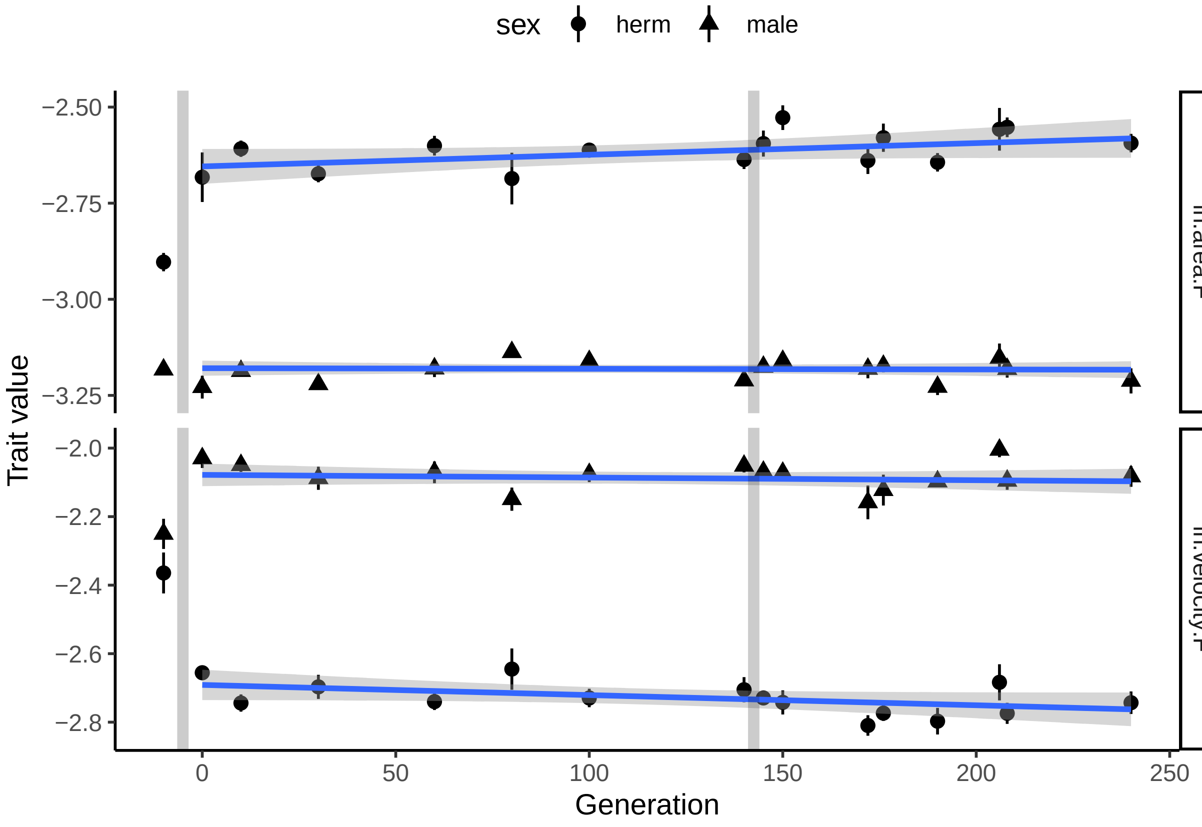
```
# among replicate variance is decreasing and varies with sex
# sex~g is weak for mean
# but stronger for dispersion
ltraits = c('ln.area.F', # though males differ in length v width relative to founders
            'ln.velocity.F')   # dramatic divergence from founders, and herms (only) declining
popm = subset(dfoorp, pop=='populations')
fnd = subset(dfoorp, pop=='founders')
popm = rbind(popm, fnd)
# these two traits show strong phenotypic correlation, yet divergent sex-specific evolution
round(cor(popm[,c('ln.area.F', 'length.F', 'width.F', 'ln.velocity.F')])^2, 3)
```

```
##               ln.area.F length.F width.F ln.velocity.F
## ln.area.F         1.000    0.966   0.967         0.736
## length.F          0.966    1.000   0.884         0.673
## width.F           0.967    0.884   1.000         0.761
## ln.velocity.F     0.736    0.673   0.761         1.000
```

```
popm$maleDep = F; popm$maleDep[grep('noM', popm$line)] = T
popm$line = gsub('noM', '', popm$line)
# exclude male depleted samples
popm = subset(popm, !maleDep)
popm$g = 0; popm$g[grep("CA", popm$line)] = 140+as.numeric(substr(popm$line[grep("CA", popm$line)], 4, 
popm$g = as.numeric(popm$g)
popm$g[popm$pop=='founders']= -10
popm = popm[grep('^M', popm$line, invert = T),]; popm$g[popm$line=='A0']=0
popm = melt(as.data.table(popm[,c('line', 'pop', 'g', 'sex', ltraits)]), 1:4, variable.name='trait')
popm$x = paste(popm$pop, popm$sex)
ggplot(popm, aes(g, value, shape=sex, group=x)) + stat_summary() +
  geom_smooth(method='lm') + facet_grid(trait~., scales='free') +
  geom_vline(aes(xintercept=-5), alpha=0.2, size=2) +
  geom_vline(aes(xintercept=142.5), alpha=0.2, size=2) +
  theme_classic() +
  theme(legend.position = 'top') + labs(x = 'Generation', y = 'Trait value')
```

```
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
```

```
## `geom_smooth()` using formula 'y ~ x'
```



in populations-1.pdf

Clear evidence for population hermaphrodite size increasing from founders. Suggestive mean effects only when excluding the founders.

```
lapply(split(popm, popm$trait), function(x) summary(lmer(value~sex*g+(1|line), data=x)))
```

```
## $ln.area.F
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: value ~ sex * g + (1 | line)
##    Data: x
##
## REML criterion at convergence: -404.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.5692 -0.3826  0.0363  0.5197  2.2433
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  line     (Intercept) 0.001363 0.03692
##  Residual             0.012035 0.10970
## Number of obs: 300, groups:  line, 67
##
## Fixed effects:
##               Estimate Std. Error         df  t value Pr(>|t|)
## (Intercept) -2.768e+00  1.517e-02  1.555e+02 -182.468  < 2e-16 ***
## sexmale     -4.144e-01  1.921e-02  2.252e+02  -21.567  < 2e-16 ***
## g            1.051e-03  1.274e-04  1.640e+02    8.247 5.02e-14 ***
## sexmale:g   -1.054e-03  1.623e-04  2.252e+02   -6.493 5.31e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) sexmal g
## sexmale   -0.633
## g         -0.745  0.479
## sexmale:g  0.476 -0.752 -0.637
##
## $ln.velocity.F
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: value ~ sex * g + (1 | line)
##    Data: x
##
## REML criterion at convergence: -180.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.7799 -0.5857 -0.0413  0.5074  4.2713
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  line     (Intercept) 0.009886 0.09943
##  Residual             0.022136 0.14878
## Number of obs: 300, groups:  line, 67
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept) -2.543e+00  2.574e-02  1.123e+02 -98.799  < 2e-16 ***
## sexmale      3.862e-01  2.606e-02  2.287e+02  14.820  < 2e-16 ***
## g           -1.293e-03  2.150e-04  1.154e+02  -6.014 2.18e-08 ***
## sexmale:g    1.727e-03  2.201e-04  2.287e+02   7.843 1.67e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) sexmal g
## sexmale   -0.506
## g         -0.740  0.385
## sexmale:g  0.381 -0.752 -0.512
```

```
# excluding founders
lapply(split(popm, popm$trait), function(x) summary(lmer(value~sex*g+(1|line), data=subset(x, g>0))))
```

```
## boundary (singular) fit: see ?isSingular
```

```
## $ln.area.F
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: value ~ sex * g + (1 | line)
##    Data: subset(x, g > 0)
##
## REML criterion at convergence: -362.9
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.4219 -0.3656  0.0923  0.4998  1.8624
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  line     (Intercept) 2.266e-17 4.760e-09
##  Residual             1.050e-02 1.024e-01
## Number of obs: 236, groups:  line, 51
##
## Fixed effects:
##               Estimate Std. Error         df  t value Pr(>|t|)
## (Intercept) -2.651e+00  1.887e-02  2.320e+02 -140.489   <2e-16 ***
## sexmale     -5.220e-01  2.668e-02  2.320e+02  -19.566   <2e-16 ***
## g            2.786e-04  1.415e-04  2.320e+02    1.969   0.0501 .
## sexmale:g   -3.384e-04  2.001e-04  2.320e+02   -1.691   0.0921 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) sexmal g
## sexmale   -0.707
## g         -0.866  0.612
## sexmale:g  0.612 -0.866 -0.707
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see ?isSingular
##
##
## $ln.velocity.F
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: value ~ sex * g + (1 | line)
##    Data: subset(x, g > 0)
##
## REML criterion at convergence: -307.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.7744 -0.5702 -0.0107  0.5111  4.8780
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  line     (Intercept) 5.171e-05 0.007191
##  Residual             1.328e-02 0.115259
## Number of obs: 236, groups:  line, 51
##
## Fixed effects:
##               Estimate Std. Error         df  t value Pr(>|t|)
## (Intercept) -2.696e+00  2.133e-02  1.686e+02 -126.392   <2e-16 ***
## sexmale      6.114e-01  3.002e-02  2.036e+02   20.368   <2e-16 ***
## g           -2.660e-04  1.598e-04  1.890e+02   -1.664   0.0977 .
## sexmale:g    2.300e-04  2.251e-04  2.036e+02    1.022   0.3080
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation of Fixed Effects:
##           (Intr) sexmal g
## sexmale   -0.704
## g         -0.866  0.610
## sexmale:g  0.609 -0.866 -0.704
```

Population (within-plate) variance in hermaphrodite size is increasing. But it is clearly decreasing for RILs (see below), which I'm more inclined to believe.

Variance among replicates decreases with time, but I'm not sure what to make of that.

```
# these are the per track variances for locomotion traits
ppsv = subset(dfoorvp, popr %in% c('populations', 'founders'))
# these are the per track means
ppsm = subset(dfoorp, popr %in% c('populations', 'founders'))
# take mean per block, and get residual variance from nonlinear regression on mean, by sex
mres <- lapply(traito, function(i) {
  df = na.exclude(cbind(ppsm[,c('line', 'sex', 'pop', 'date', 'block')], y = ppsv[,i], x = ppsm[,i]))
  df = do.call(rbind, lapply(split(df, paste(df$line, df$sex, df$date)), function(x) {
    if(nrow(x)>1){
      cbind(x[1,c('line', 'sex', 'pop', 'date', 'block')], y=mean(x$y), x = mean(x$x))
    } else {
      x
    }
  }))
  df = do.call(rbind, lapply(split(df, df$sex), function(z) {
    fit = mgcv::gam(y~s(x, bs="cs"), z, family = 'gaussian')
    print(c(i, z$sex[1]))
    print(anova(fit)$p.table[4])
    z$res <- mean(z$y) + resid(fit)
    z
  }))
  names(df)[names(df)=='res'] = i
  df[,c('line', 'sex', 'pop', 'date', 'block', i)]
})
```

```
## [1] "ln.area.F" "herm"
## [1] 3.445227e-84
## [1] "ln.area.F" "male"
## [1] 6.747426e-59
## [1] "ln.area.F.var" "herm"
## [1] 5.92094e-130
## [1] "ln.area.F.var" "male"
## [1] 1.886279e-83
## [1] "ln.area.S" "herm"
## [1] 1.014256e-89
## [1] "ln.area.S" "male"
## [1] 6.606788e-70
## [1] "ln.area.S.var" "herm"
## [1] 1.914093e-135
## [1] "ln.area.S.var" "male"
## [1] 4.345139e-91
## [1] "length.F" "herm"
## [1] 1.460266e-93
## [1] "length.F" "male"
## [1] 7.117534e-43
## [1] "width.F" "herm"
## [1] 3.042896e-94
## [1] "width.F" "male"
## [1] 1.421562e-34
## [1] "ln.length.F.var" "herm"
## [1] 1.015636e-136
## [1] "ln.length.F.var" "male"
## [1] 8.355772e-109
## [1] "ln.width.F.var" "herm"
## [1] 2.344916e-117
## [1] "ln.width.F.var" "male"
## [1] 9.610735e-116
## [1] "length.S" "herm"
## [1] 2.116377e-96
## [1] "length.S" "male"
## [1] 3.057304e-53
## [1] "width.S" "herm"
## [1] 6.381275e-98
## [1] "width.S" "male"
## [1] 1.993385e-70
## [1] "ln.length.S.var" "herm"
## [1] 3.375638e-114
## [1] "ln.length.S.var" "male"
## [1] 1.01458e-106
## [1] "ln.width.S.var" "herm"
## [1] 3.226643e-127
## [1] "ln.width.S.var" "male"
## [1] 5.791782e-112
## [1] "ln.velocity" "herm"
## [1] 5.001456e-127
## [1] "ln.velocity" "male"
## [1] 9.964474e-49
## [1] "ln.velocity.var" "herm"
## [1] 3.070317e-132
## [1] "ln.velocity.var" "male"
## [1] 2.357866e-70
## [1] "ln.velocity.F" "herm"
## [1] 7.9435e-117
## [1] "ln.velocity.F" "male"
## [1] 4.562625e-89
## [1] "ln.velocity.B" "herm"
## [1] 4.404459e-135
## [1] "ln.velocity.B" "male"
## [1] 1.207309e-109
## [1] "ln.velocity.F.var" "herm"
## [1] 2.962539e-130
## [1] "ln.velocity.F.var" "male"
## [1] 3.935004e-81
## [1] "ln.velocity.B.var" "herm"
## [1] 5.32434e-118
## [1] "ln.velocity.B.var" "male"
## [1] 1.246811e-80
## [1] "acceleration" "herm"
## [1] 9.909876e-58
## [1] "acceleration" "male"
## [1] 1.210668e-80
## [1] "ln.acceleration.var" "herm"
## [1] 3.160585e-106
## [1] "ln.acceleration.var" "male"
## [1] 1.939847e-79
## [1] "ln.acceleration.F" "herm"
## [1] 1.985055e-23
## [1] "ln.acceleration.F" "male"
## [1] 2.000063e-54
## [1] "ln.acceleration.B" "herm"
## [1] 2.176672e-13
## [1] "ln.acceleration.B" "male"
## [1] 1.231438e-65
## [1] "ln.acceleration.F.var" "herm"
## [1] 4.549399e-74
## [1] "ln.acceleration.F.var" "male"
## [1] 1.334464e-69
## [1] "ln.acceleration.B.var" "herm"
## [1] 1.072529e-21
## [1] "ln.acceleration.B.var" "male"
## [1] 1.492581e-18
## [1] "run"  "herm"
## [1] 1.166674e-34
## [1] "run"  "male"
## [1] 1.972094e-48
## [1] "ln.run.var" "herm"
## [1] 1.717414e-176
## [1] "ln.run.var" "male"
## [1] 2.089651e-125
## [1] "ln.run.F" "herm"
## [1] 4.174408e-140
## [1] "ln.run.F" "male"
## [1] 8.614312e-99
## [1] "run.B" "herm"
## [1] 9.057078e-59
## [1] "run.B" "male"
## [1] 2.394965e-08
## [1] "ln.run.F.var" "herm"
## [1] 6.693981e-166
## [1] "ln.run.F.var" "male"
## [1] 1.398854e-109
## [1] "ln.run.B.var" "herm"
## [1] 9.546415e-118
## [1] "ln.run.B.var" "male"
## [1] 1.049343e-52
## [1] "curvature" "herm"
## [1] 7.558327e-105
## [1] "curvature" "male"
## [1] 2.191991e-87
## [1] "ln.curvature.var" "herm"
## [1] 9.053437e-145
## [1] "ln.curvature.var" "male"
## [1] 9.443489e-107
## [1] "curvature.F" "herm"
## [1] 4.834885e-108
## [1] "curvature.F" "male"
## [1] 2.517942e-77
## [1] "curvature.B" "herm"
## [1] 2.299142e-110
## [1] "curvature.B" "male"
## [1] 4.585743e-71
## [1] "curvature.S" "herm"
## [1] 6.123959e-120
## [1] "curvature.S" "male"
## [1] 6.775915e-105
## [1] "ln.curvature.F.var" "herm"
## [1] 2.307189e-137
## [1] "ln.curvature.F.var" "male"
## [1] 1.932557e-89
## [1] "ln.curvature.B.var" "herm"
## [1] 3.3156e-117
## [1] "ln.curvature.B.var" "male"
## [1] 4.180691e-95
## [1] "ln.curvature.S.var" "herm"
## [1] 8.995779e-120
## [1] "ln.curvature.S.var" "male"
## [1] 5.57588e-98
## [1] "ln.angular" "herm"
## [1] 5.991862e-109
## [1] "ln.angular" "male"
## [1] 9.752559e-106
## [1] "ln.angular.var" "herm"
## [1] 1.546827e-126
## [1] "ln.angular.var" "male"
## [1] 8.859632e-113
## [1] "ln.angular.F" "herm"
## [1] 1.350585e-110
## [1] "ln.angular.F" "male"
## [1] 2.693025e-104
## [1] "ln.angular.B" "herm"
## [1] 3.504136e-140
## [1] "ln.angular.B" "male"
## [1] 2.500673e-111
## [1] "ln.angular.S" "herm"
## [1] 6.438016e-107
## [1] "ln.angular.S" "male"
## [1] 3.853781e-100
## [1] "ln.angular.F.var" "herm"
## [1] 3.73409e-123
## [1] "ln.angular.F.var" "male"
## [1] 1.324225e-110
## [1] "ln.angular.B.var" "herm"
## [1] 2.021251e-143
## [1] "ln.angular.B.var" "male"
## [1] 6.453439e-88
## [1] "ln.angular.S.var" "herm"
## [1] 1.293286e-134
## [1] "ln.angular.S.var" "male"
## [1] 3.676547e-105
```

```
pps = mres %>% Reduce(function(d1,d2) inner_join(d1,d2, by=c('line', 'sex', 'pop', 'block', 'date')), .
pps$pop <- pps$popr <- pps$line
subp = pps
subp$pop <- gsub('noM', '', subp$pop)
traits = traito
pmx <- subp %>% group_by(line, sex, pop, popr, block) %>% summarise_at(traits, function(x) mean(x, na.r
pmx$gen <- 0
pmx$gen[grep('A..', pmx$pop)] <- as.numeric(gsub('A.', '', pmx$pop[grep('A..', pmx$pop)]))
```

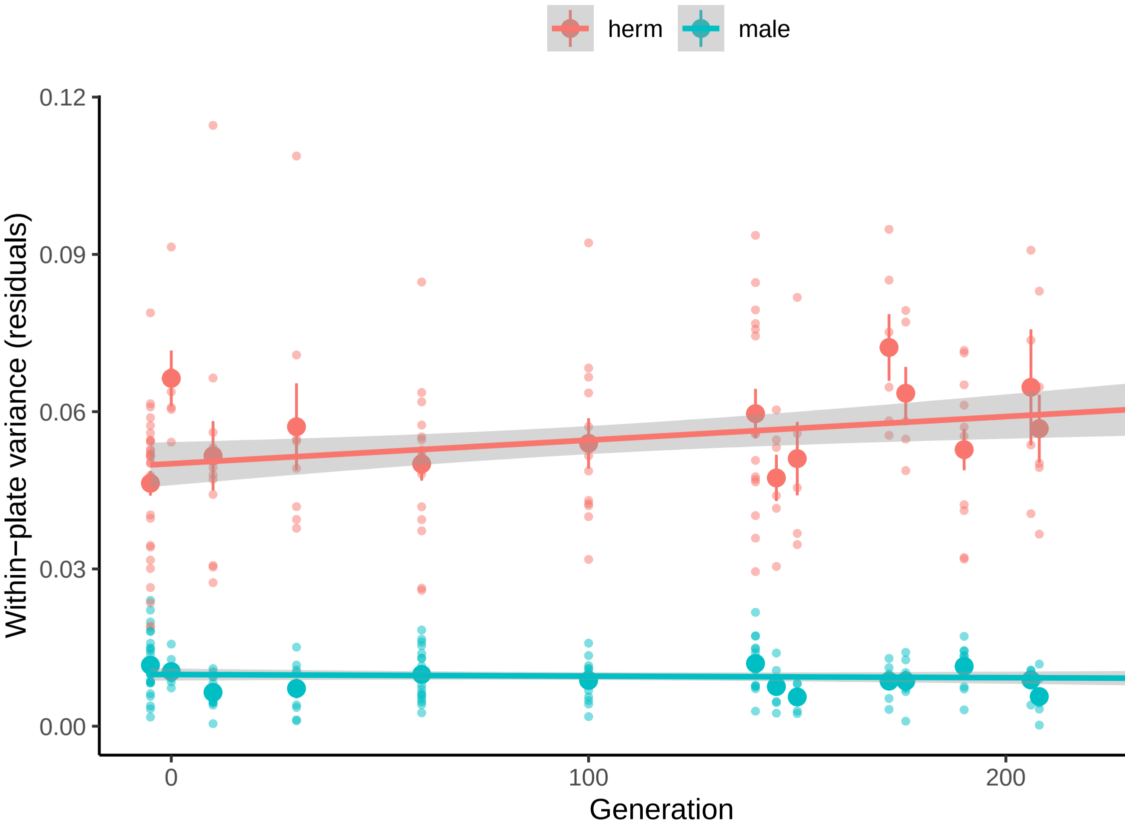
```
## Warning: NAs introduced by coercion
```

```
pmx$gen[grep('CA', pmx$pop)] <- as.numeric(gsub('CA.', '', pmx$pop[grep('CA', pmx$pop)]))+140
pmx$gen[grep('^D', pmx$pop)] <- as.numeric(gsub('^D.', '', pmx$pop[grep('^D', pmx$pop)]))+140
pmx$gen[grep('^M', pmx$pop)] <- as.numeric(gsub('^M.', '', pmx$pop[grep('^M', pmx$pop)]))+140
pmx$gen[pmx$line %in% cemeeFounders] = -5
table(pmx$pop, pmx$gen)
```

```
##
##         -5  0 10 30 60 100 140 145 150 172 176 190 206 208 220 240
##   A0      0  8  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A110    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A1100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A130    0  0  0  4  0   0   0   0   0   0   0   0   0   0   0   0
##   A160    0  0  0  0  4   0   0   0   0   0   0   0   0   0   0   0
##   A210    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A2100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A230    0  0  0  4  0   0   0   0   0   0   0   0   0   0   0   0
##   A260    0  0  0  0  4   0   0   0   0   0   0   0   0   0   0   0
##   A310    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A3100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A330    0  0  0  4  0   0   0   0   0   0   0   0   0   0   0   0
##   A360    0  0  0  0  4   0   0   0   0   0   0   0   0   0   0   0
##   A410    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A4100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A4140   0  0  0  0  0   0   8   0   0   0   0   0   0   0   0   0
##   A430    0  0  0  4  0   0   0   0   0   0   0   0   0   0   0   0
##   A460    0  0  0  0  8   0   0   0   0   0   0   0   0   0   0   0
##   A510    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A5100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A5140   0  0  0  0  0   0   8   0   0   0   0   0   0   0   0   0
##   A560    0  0  0  0  8   0   0   0   0   0   0   0   0   0   0   0
##   A610    0  0  4  0  0   0   0   0   0   0   0   0   0   0   0   0
##   A6100   0  0  0  0  0   4   0   0   0   0   0   0   0   0   0   0
##   A6140   0  0  0  0  0   0  16   0   0   0   0   0   0   0   0   0
##   A660    0  0  0  0  8   0   0   0   0   0   0   0   0   0   0   0
##   AB1     4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   CA110   0  0  0  0  0   0   0   0   4   0   0   0   0   0   0   0
##   CA1100  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   8
##   CA136   0  0  0  0  0   0   0   0   0   0   4   0   0   0   0   0
##   CA15    0  0  0  0  0   0   0   4   0   0   0   0   0   0   0   0
##   CA150   0  0  0  0  0   0   0   0   0   0   0   8   0   0   0   0
##   CA168   0  0  0  0  0   0   0   0   0   0   0   0   0   4   0   0
##   CA210   0  0  0  0  0   0   0   0   4   0   0   0   0   0   0   0
##   CA2100  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   8
##   CA236   0  0  0  0  0   0   0   0   0   0   4   0   0   0   0   0
##   CA25    0  0  0  0  0   0   0   4   0   0   0   0   0   0   0   0
##   CA250   0  0  0  0  0   0   0   0   0   0   0   8   0   0   0   0
##   CA268   0  0  0  0  0   0   0   0   0   0   0   0   0   4   0   0
##   CA310   0  0  0  0  0   0   0   0   4   0   0   0   0   0   0   0
##   CA3100  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   8
##   CA336   0  0  0  0  0   0   0   0   0   0   4   0   0   0   0   0
##   CA35    0  0  0  0  0   0   0   4   0   0   0   0   0   0   0   0
##   CA350   0  0  0  0  0   0   0   0   0   0   0   8   0   0   0   0
##   CA368   0  0  0  0  0   0   0   0   0   0   0   0   0   4   0   0
##   CA432   0  0  0  0  0   0   0   0   0   4   0   0   0   0   0   0
##   CA466   0  0  0  0  0   0   0   0   0   0   0   0   4   0   0   0
##   CA532   0  0  0  0  0   0   0   0   0   4   0   0   0   0   0   0
##   CA566   0  0  0  0  0   0   0   0   0   0   0   0   4   0   0   0
##   CA632   0  0  0  0  0   0   0   0   0   4   0   0   0   0   0   0
##   CB4507  4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   CB4852  4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   CB4855  4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   CB4856  4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   CB4858  4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   D580    0  0  0  0  0   0   0   0   0   0   0   0   0   0   2   0
##   D680    0  0  0  0  0   0   0   0   0   0   0   0   0   0   2   0
##   D780    0  0  0  0  0   0   0   0   0   0   0   0   0   0   2   0
##   JU319   3  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   JU345   3  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   JU400   4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   M480    0  0  0  0  0   0   0   0   0   0   0   0   0   0   1   0
##   M580    0  0  0  0  0   0   0   0   0   0   0   0   0   0   2   0
##   M680    0  0  0  0  0   0   0   0   0   0   0   0   0   0   1   0
##   MY1     4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   MY16    4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   N2anc   3  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   OF5     0  4  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   PB306   4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   PX174   4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   PX179   4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
##   RC301   4  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0
```

```r
# mating system. ignoring M, D here
pmx$mate = 'A'; pmx$mate[grep('^D', pmx$pop)] = "D"; pmx$mate[grep('^M', pmx$pop)] = "M"
pmx$mate[pmx$line %in% cemeeFounders] = 'F'
pmx$mate = factor(pmx$mate)
pmx$mate = factor(pmx$mate, levels=c('F', 'A', 'D', 'M'))
pmx$mate = factor(pmx$mate, labels=c('Founder', 'Andro.', 'Di.', 'Mono.'))

ggplot(subset(pmx, mate %in% c("Founder", "Andro.")), aes(gen, ln.area.F, col=sex)) +
  geom_point(alpha=0.5, stroke=0) + stat_summary() +
  theme_classic() + geom_smooth(method='lm') +
  scale_x_continuous(breaks = scales::pretty_breaks(n=2)) +
  labs(x = "Generation", y = "Within-plate variance (residuals)") +
  theme(legend.position = 'top') + scale_color_discrete("")
```

```
## No summary function supplied, defaulting to `mean_se()`

## `geom_smooth()` using formula 'y ~ x'
```



within-plate variance-1.pdf

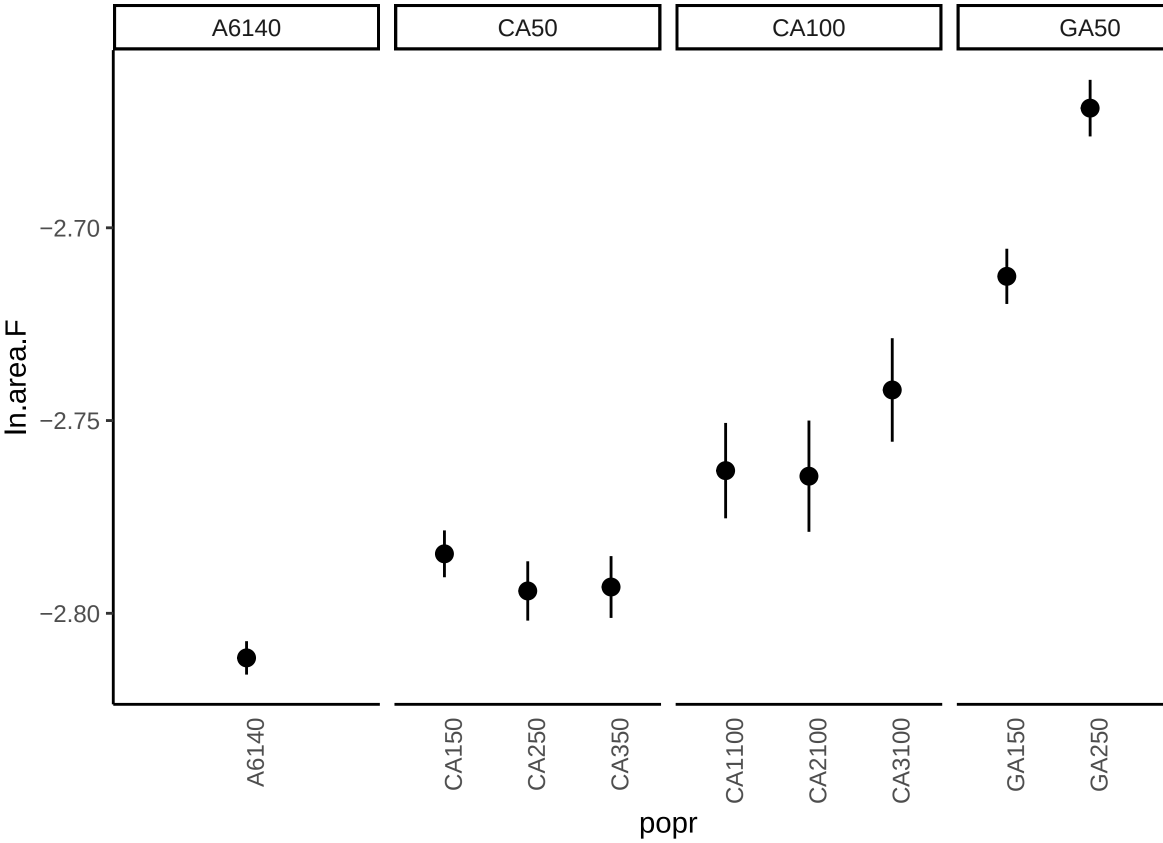
```
# all data. founders are not so different here
summary(lmer(ln.area.F~sex*gen+(1|line), data=subset(pmx, mate %in% c("Founder", "Andro."))))
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ln.area.F ~ sex * gen + (1 | line)
##    Data: subset(pmx, mate %in% c("Founder", "Andro."))
##
## REML criterion at convergence: -1830
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.7640 -0.3760 -0.0490  0.3609  4.9576
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  line     (Intercept) 6.174e-06 0.002485
##  Residual             1.558e-04 0.012483
## Number of obs: 321, groups:  line, 79
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  5.006e-02  1.562e-03  2.211e+02  32.047  < 2e-16 ***
## sexmale     -4.023e-02  2.153e-03  2.458e+02 -18.689  < 2e-16 ***
## gen          4.473e-05  1.214e-05  2.191e+02   3.686 0.000288 ***
## sexmale:gen -4.790e-05  1.664e-05  2.449e+02  -2.879 0.004338 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) sexmal gen
## sexmale     -0.672
## gen         -0.757  0.509
## sexmale:gen  0.512 -0.762 -0.675
```

RIL mean size increases, and within-plate variance decreases (despite the increase in mean, which has not been regressed out here).

```
# CeMEE v1.2 paper: e.g. traits curvature, velocity, size (mean/variance) on NGM

# plate means
pm <- subset(dfoorp, !pop %in% c('founders', 'MA', 'populations'))
pm$popr = factor(pm$popr); pm$popr = factor(pm$popr, levels = unique(pm$popr)[c(1,3,5,7,2,4,6,8:10)])
pm$pop = factor(pm$pop); pm$pop = factor(pm$pop, levels = unique(pm$pop)[c(1,3,2,4)])
pm$gen = as.numeric(substr(pm$pop, 3, 5)); pm$gen[pm$gen==140]=0

ggplot(pm, aes(popr, ln.area.F)) + stat_summary() + facet_grid(.~pop, scales='free') +
  theme_classic() + theme(axis.text.x = element_text(angle=90), axis.ticks.x = element_blank())
```

```
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
```



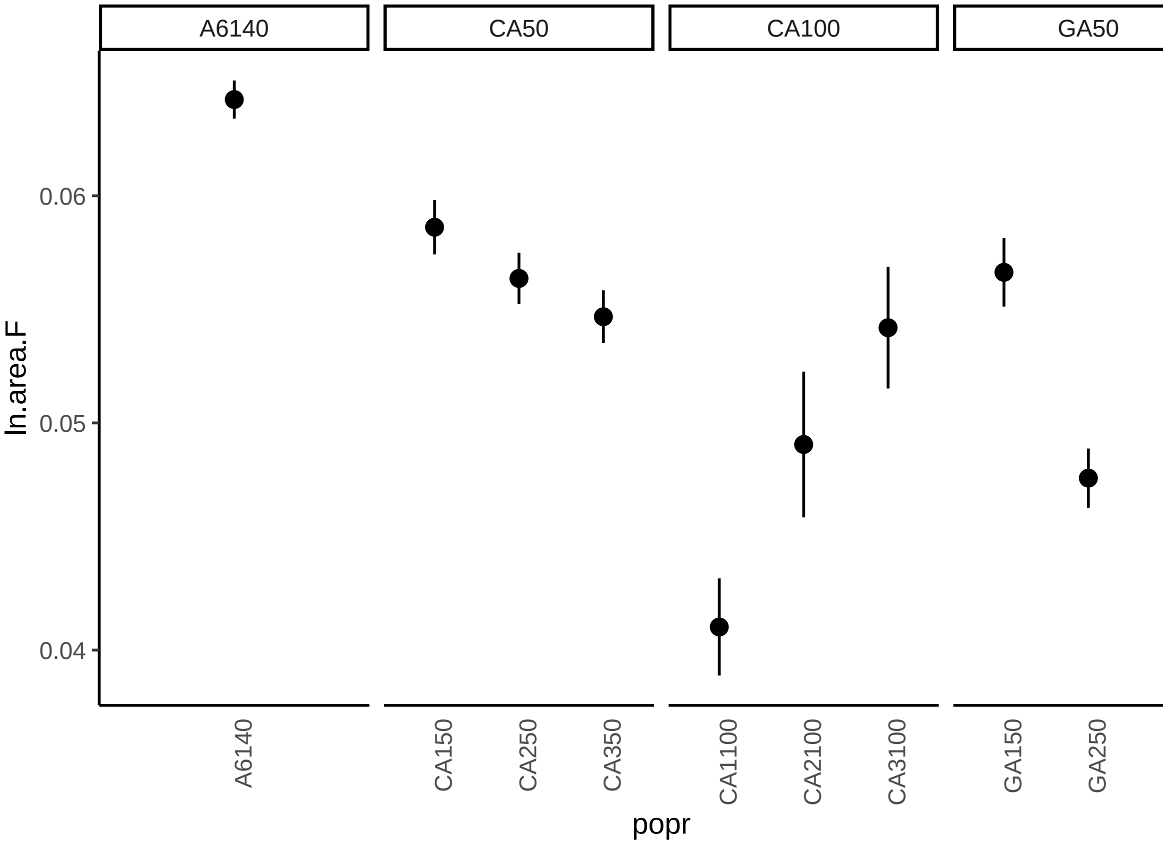
mean and variance-1.pdf

```r
# mean test
summary(lmer(ln.area.F~gen+(1|line)+(1|date), data=subset(pm, !pop=='GA50')))
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ln.area.F ~ gen + (1 | line) + (1 | date)
##    Data: subset(pm, !pop == "GA50")
##
## REML criterion at convergence: -2997.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.6577 -0.5106  0.0303  0.5630  2.8668
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  line     (Intercept) 0.004947 0.07033
##  date     (Intercept) 0.002705 0.05201
##  Residual             0.007693 0.08771
## Number of obs: 1892, groups:  line, 516; date, 186
##
## Fixed effects:
##               Estimate Std. Error         df  t value Pr(>|t|)
## (Intercept) -2.814e+00  7.320e-03  4.054e+02 -384.452  < 2e-16 ***
## gen          6.338e-04  1.321e-04  4.564e+02    4.797 2.19e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## gen -0.658
```

```r
# within-plate variance
pm <- subset(dfoorvp, !pop %in% c('founders', 'MA', 'populations'))
pm$popr = factor(pm$popr); pm$popr = factor(pm$popr, levels = unique(pm$popr)[c(1,3,5,7,2,4,6,8:10)])
pm$pop = factor(pm$pop); pm$pop = factor(pm$pop, levels = unique(pm$pop)[c(1,3,2,4)])
pm$gen = as.numeric(substr(pm$pop, 3, 5)); pm$gen[pm$gen==140]=0

# variance test
ggplot(pm, aes(popr, ln.area.F)) + stat_summary() + facet_grid(.~pop, scales='free') +
  theme_classic() + theme(axis.text.x = element_text(angle=90), axis.ticks.x = element_blank())
```

```
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
```



mean and variance-2.pdf

```
summary(lmer(ln.area.F~gen+(1|line)+(1|date), data=subset(pm, !pop=='GA50')))
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ln.area.F ~ gen + (1 | line) + (1 | date)
##    Data: subset(pm, !pop == "GA50")
##
## REML criterion at convergence: -9190.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.4173 -0.5685 -0.1061  0.4245  6.2955
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  line     (Intercept) 2.112e-04 0.014532
##  date     (Intercept) 5.707e-05 0.007554
##  Residual             2.951e-04 0.017179
## Number of obs: 1892, groups:  line, 516; date, 186
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  6.509e-02  1.337e-03  4.103e+02  48.679  < 2e-16 ***
## gen         -1.671e-04  2.379e-05  4.168e+02  -7.025 8.78e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Correlation of Fixed Effects:
##     (Intr)
## gen -0.687
```