Longitudinal Variance GWAS

Correlated phenotypes across time points

There are multiple ways to model correlated phenotypes. One way is to use the function `rmvnorm` from the r package `mvtnorm` which uses single value decomposition to set the correlation across time points. However, this procedure does not readily allow us to concurrently model the effect of an independent variable, X, on the phenotype, Y. To do this, we use linear regression instead in the form

$$Y_{ij} = \beta X + \epsilon_{ij}$$

where,

$$\epsilon \sim N\begin{pmatrix} & 1 & \rho & \rho \\ 0, & \rho & 1 & \rho \\ & \rho & \rho & 1 \end{pmatrix} \tag{1}$$

In this linear equation, β is the effect of the independent variable, X, on phenotype, Y, and ϵ is assumed normally distributed with mean 0 and variance equal to the variance-covariance matrix defining the correlation of phenotypes across time points, ρ.

We generate ϵ by taking the product of an error term generated from a Gaussian distribution and the inverse of the variance-covariance matrix. Let us assume we have correlated phenotypes (r = 0.8) across 3 time points (t) for 50,000 individuals (N).

```r
## Define phenotypic correlation and time points

r <- 0.8
t <- 3

## Create a t x t cov matrix and take its inverse using Cholesky decomposition

covmat <- matrix(c(1,r,r,r,1,r,r,r,1),t,t)
Lambda <- chol(covmat)
```

```
## Generate epsilon by generating an error term and multiplying it by Lambda

N <- 50000
err <- matrix(rnorm(N * t), N, t)
epsilon <- err %*% Lambda

## Define X as Bernoulli variable and set beta

X <- rbinom(N, 2, 0.3)
b <- 0.1

## Model Y using linear regression

Y = X %*% matrix(b, 1, t) + epsilon

## Check the correlation of phenotypes and beta are as simulated

round(cor(Y),2)
```

```
     [,1] [,2] [,3]
[1,]  1.0  0.8  0.8
[2,]  0.8  1.0  0.8
[3,]  0.8  0.8  1.0
```

```
lm(Y~X)
```

```
Call:
lm(formula = Y ~ X)

Coefficients:
             [,1]       [,2]       [,3]     
(Intercept)  -0.005128  -0.008579  -0.011419
X             0.104053   0.110719   0.114458
```

The results show that the phenotypic correlation across time points is as intended and the estimated beta is the same as the simulated beta.

Note that above we defined X as a Bernoulli variable. This is because our aim is to model the effect of a genetic variant, SNP, on phenotype. We used a probability of 0.3, which corresponds to the minor allele frequency (MAF). With the above as our basis, we created 3 functions to more easily simulate a range of scenarios. We use them in succeeding sections to explore a number of questions relevant to our overall research objectives.

Functions

Each function below produces a data frame, dat. It contains uncorrelated (phe) and correlated phenotypes (qt) collected on 3 different time points (time) from unrelated individuals (id), the genetic variant (snp), and for functions 2 and 3, an additional independent genetic variant (snp2) and 3 sets of normally distributed environmental factor (E) – 1 set per time point. These additional variables are used to simulate the variance effect of SNP on phenotype via SNP-by-SNP or SNP-by-E interaction, respectively.

1. simrdatm.R models a phenotype based on main effect of SNP alone;
2. simrdatvq.R models a phenotype based on SNP-by-SNP interaction; and
3. simrdatvE.R models a phenotype based on SNP-by-environment interaction.

```r
## Load functions (change later to download directly from github)

source("~/Desktop/AP Genetics/simrdatm.R")
source("~/Desktop/AP Genetics/simrdatvq.R")
source("~/Desktop/AP Genetics/simrdatvE.R")
source("~/Desktop/AP Genetics/simrdatvt.R")

## Load libraries

suppressMessages(library(car))
suppressMessages(library(ggplot2))
suppressMessages(library(lme4))
```

simrdatm.R

We begin by checking that the function is working as intended. We want to make sure that the simulated beta is estimated correctly by both a simple and mixed effect linear model. The latter is used to check if the phenotypic correlation is upheld after generating phenotypes in a linear mixed effect model. We then test that the variance effect of a SNP modeled through SNP-by-SNP or SNP-by-E interaction is detected in Levene´s Test of Homogeneity of Variance (Brown-Forsythe).

```r
## Define covariance matrix and take its inverse

r <- 0.8
t <- 3
covmat <- matrix(c(1,r,r,r,1,r,r,r,1),t,t)
Lambda <- chol(covmat)

## Generate data
# @Para: nsnp (double) is the number of snps being tested, N (double) is the
# number of unrelated individuals, t (double) is the number of time points,
# TIME (vector) is the spacing of measurement, Lambda (matrix) is the inverse
# of the covariance matrix defining the correlation among phenotypes across
# time points derived via cholesky decomposition, maf (double) is the
# minor allele frequency and bsnp (double) is the snp effect size. @Out: dat (df).

simrdatm(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## Check beta estimated aligns with beta simulated

summary(lm(qt ~ snp, data = dat))
```

```
Call:
lm(formula = qt ~ snp, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-4.7005 -0.6815 -0.0008  0.6811  4.2554

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.002657   0.002880   0.923    0.356
snp         0.795012   0.006145 129.374   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.009 on 149998 degrees of freedom
Multiple R-squared:  0.1004,    Adjusted R-squared:  0.1004
F-statistic: 1.674e+04 on 1 and 149998 DF,  p-value: < 2.2e-16
```

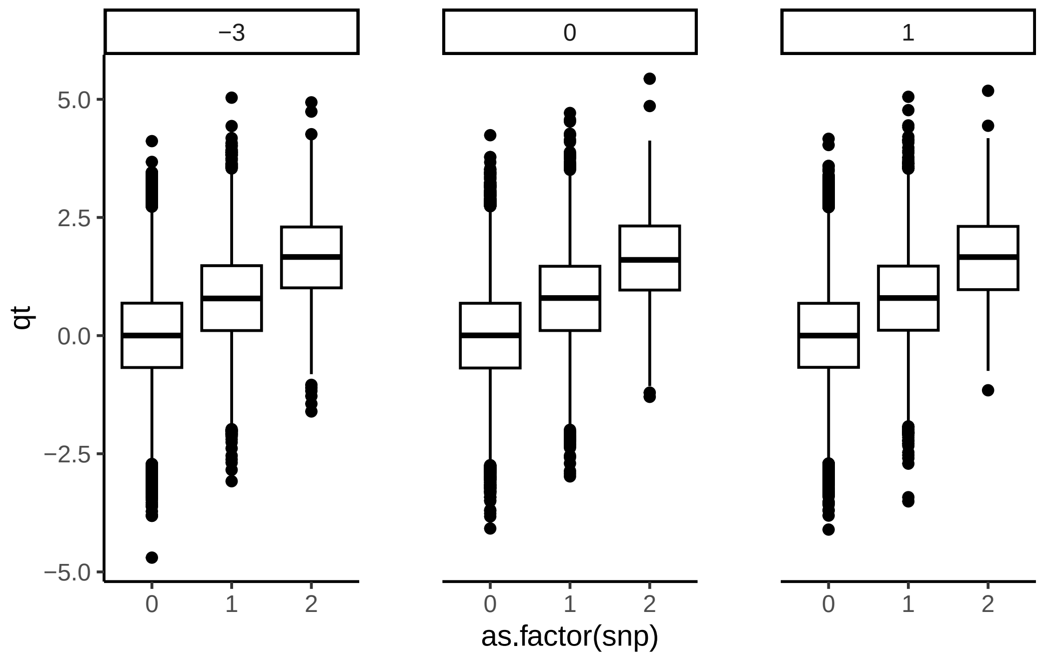
```r
## Check random intercept effect is equal to r

summary(lmer(qt ~ snp + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ snp + (1 | id)
   Data: dat

REML criterion at convergence: 314227.4

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.8295 -0.5592 -0.0019  0.5606  3.7997

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8158   0.9032
 Residual             0.2015   0.4489
Number of obs: 150000, groups:  id, 50000

Fixed effects:
            Estimate Std. Error t value
(Intercept) 0.002657   0.004647   0.572
snp         0.795012   0.009916  80.176

Correlation of Fixed Effects:
    (Intr)
snp -0.427
```

```r
## Check the absence of variance effect

leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value Pr(>F)
group      8  0.1062  0.999
      149991
```

```r
## Visualize the data

ggplot(dat, aes(x = as.factor(snp), y = qt, fill = as.factor(snp))) +
  geom_boxplot(position = position_dodge(width = 0.8), color = "black") +
  scale_fill_manual(values = c("NA", "NA", "NA")) +
  theme_classic() +
  theme(legend.position = "none", panel.spacing.x = unit(1.1, "cm")) +
  facet_grid(~time, scales = "free_x", space = "free_x")
```



Here we see that the results match our simulated beta and expected variance effect, which is nil since we modeled mean-only effect.The box plot confirms this finding where the median (i.e. the mean since we modeled the phenotype to be normally distributed) is incremental across SNP levels.

simrdatvq.R

Next, we test our function that models SNP variance effect via SNP-by-SNP interaction. The same input parameters are used as the prior function. A similar approach is used to validate the beta estimate and variance effect are in line with expectation.

```r
## Generate data

simrdatvq(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)
```

```r
## Check beta estimated aligns with beta simulated

summary(lm(qt ~ snp + snp*snp2, data = dat))
```

```
Call:
lm(formula = qt ~ snp + snp * snp2, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.8657 -0.6751  0.0041  0.6802  4.3908 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.0008344  0.0031670   0.263    0.792    
snp         -0.0048695  0.0067299  -0.724    0.469    
snp2        -0.0050229  0.0067132  -0.748    0.454    
snp:snp2     0.8396345  0.0144129  58.256   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.003 on 149996 degrees of freedom
Multiple R-squared:  0.03114,	Adjusted R-squared:  0.03112 
F-statistic:  1607 on 3 and 149996 DF,  p-value: < 2.2e-16
```

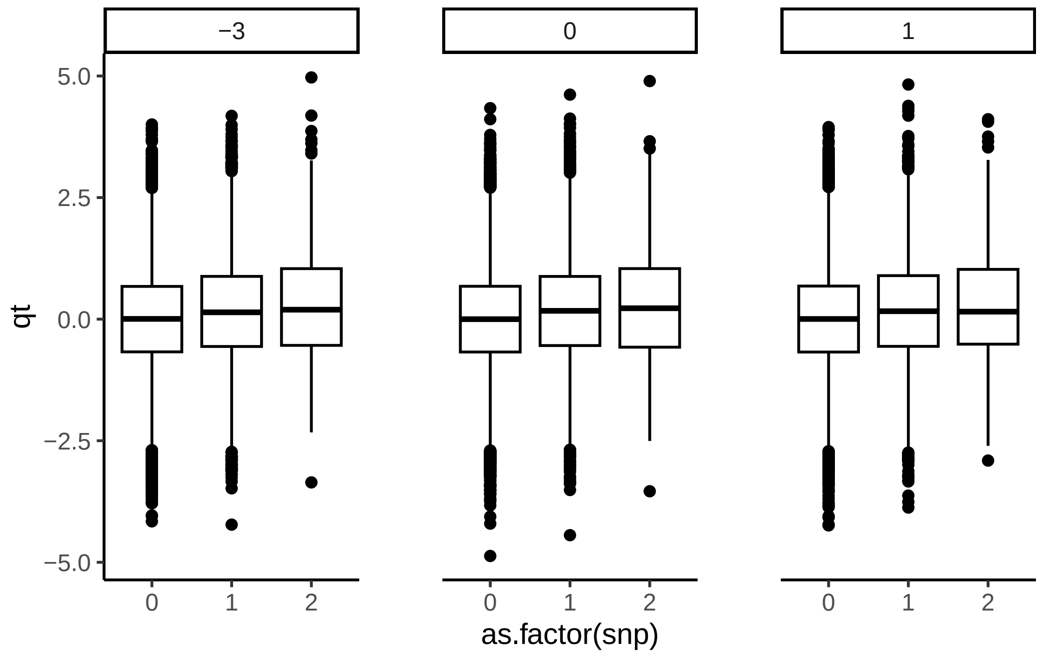
```r
## Check random intercept effect is equal to r

summary(lmer(qt ~ snp + snp*snp2 + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ snp + snp * snp2 + (1 | id)
   Data: dat

REML criterion at convergence: 312993.7

Scaled residuals: 
    Min      1Q  Median      3Q     Max 
-3.7708 -0.5569  0.0024  0.5581  3.6209 

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8058   0.8977
 Residual             0.2002   0.4475
Number of obs: 150000, groups:  id, 50000

Fixed effects:
              Estimate Std. Error t value
(Intercept)  0.0008344  0.0051087   0.163
snp         -0.0048695  0.0108560  -0.449
snp2        -0.0050229  0.0108290  -0.464
snp:snp2     0.8396345  0.0232492  36.115

Correlation of Fixed Effects:
         (Intr) snp    snp2
snp      -0.426
snp2     -0.427  0.182
snp:snp2  0.180 -0.428 -0.426
```

```
## Check the presence of variance effect

leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value    Pr(>F)
group      8  28.668 < 2.2e-16 ***
      149991
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Visualize the data

ggplot(dat, aes(x = as.factor(snp), y = qt, fill = as.factor(snp))) +
  geom_boxplot(position = position_dodge(width = 0.8), color = "black") +
  scale_fill_manual(values = c("NA", "NA", "NA")) +
  theme_classic() +
  theme(legend.position = "none", panel.spacing.x = unit(1.1, "cm")) +
  facet_grid(~time, scales = "free_x", space = "free_x")
```



The beta estimate aligns with the beta simulated. And, as expected, we reject the null hypothesis of variance homogeneity.

However, upon closer inspection of the box plot, SNP seems to also have an effect on mean and not just variance. We explore this more in the section **"The correlation of mean-variance effect"**.

We also need to check that when beta is 0 Levene's Test should indicate variance homogeneity.

```
## Generate data and set beta to 0

simrdatvq(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
          maf = 0.1, bsnp = 0)

## Check absence of variance effect

leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value Pr(>F)
group      8  0.6835 0.7066
      149991
```

The results above confirm variance homogeneity. So, we know our function is working as designed.

simrdatvE.R

Same as above except now we model a SNP's effect on phenotypic variance via SNP-by-E interaction, where E is varying per person and per time point.

```
## Generate data

simrdatvE(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## Check beta estimated aligns with beta simulated

summary(lm(qt ~ snp + snp*E, data = dat))
```

```
Call:
lm(formula = qt ~ snp + snp * E, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.0941 -0.6720  0.0017  0.6714  4.9606 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.0003601  0.0028540  -0.126    0.900    
snp         -0.0016880  0.0060152  -0.281    0.779    
E           -0.0029366  0.0028586  -1.027    0.304    
snp:E        0.8006561  0.0059881 133.708   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9987 on 149996 degrees of freedom
Multiple R-squared:  0.1268,	Adjusted R-squared:  0.1268 
F-statistic:  7262 on 3 and 149996 DF,  p-value: < 2.2e-16
```

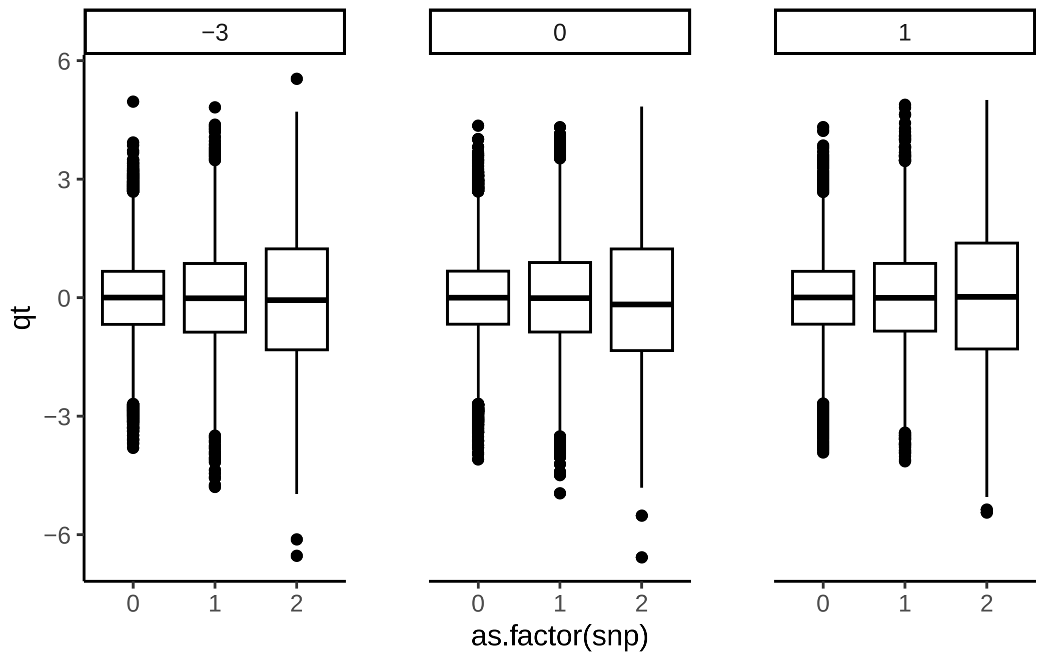
```
## Check random intercept effect is equal to r

summary(lmer(qt ~ snp + snp*E + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ snp + snp * E + (1 | id)
   Data: dat

REML criterion at convergence: 312284.1

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.8402 -0.5601  0.0008  0.5605  3.6948

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.7976   0.8931
 Residual             0.1998   0.4469
Number of obs: 150000, groups:  id, 50000

Fixed effects:
              Estimate Std. Error t value
(Intercept) -3.667e-04  4.602e-03  -0.080
snp         -1.672e-03  9.698e-03  -0.172
E            8.531e-05  1.536e-03   0.056
snp:E        7.990e-01  3.216e-03 248.415

Correlation of Fixed Effects:
      (Intr) snp    E
snp   -0.429
E     -0.001  0.001
snp:E  0.001  0.000 -0.430
```

```
## Check the presence of variance effect

leveneTest(dat$qt~ as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value    Pr(>F)
group      8  578.37 < 2.2e-16 ***
      149991
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
## Visualize the data

ggplot(dat, aes(x = as.factor(snp), y = qt, fill = as.factor(snp))) +
  geom_boxplot(position = position_dodge(width = 0.8), color = "black") +
  scale_fill_manual(values = c("NA", "NA", "NA")) +
  theme_classic() +
  theme(legend.position = "none", panel.spacing.x = unit(1.1, "cm")) +
  facet_grid(~time, scales = "free_x", space = "free_x")
```



Once again, the beta estimate is equal to the simulated value and we find heterogeneous variance across SNP levels. These findings are visualized in the graph where the difference in the interquartile range is clearly visible across SNP levels.

Similar to the prior function, we ask the question when beta is 0, will Levene Test confirm the null hypothesis of variance homogeneity?

```r
## Generate data and set beta to 0

simrdatvE(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
          maf = 0.1, bsnp = 0)

## Check absence of variance effect
```

```r
leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value Pr(>F)
group      8  0.4247  0.907
      149991
```

Indeed, it does. Therefore, our function behaves in line with expectation.

SCENARIOS

Now that we know our functions are working as intended, we can use them to explore a number of questions that may help us in setting up and further polishing our study design. They also give us an indication how the results might look like once we run our analyses on real data.

We start by exploring the relationship between mean and variance effects. We then check if one of our main hypotheses holds true, i.e. that a subset of variance effects will be indexed by our SNP-by-time interaction term in a linear mixed effect model. Next, we generate power curves under different scenarios to get an idea of the magnitude of effects we might expect to find in our study. We additionally look into the accuracy of our estimates from our power simulation to gain an insight as to how low power impacts these estimates.

In the final 2 sections, we discuss the concept of phenotypic variance explained (PVE) and how this might not be applicable in studies that use phenotype variance as outcome. We end by simulating multiple SNPs to construct a vPGI and investigate how this can be used as a tool to predict phenotypic dispersion.

The correlation of mean-variance effect

As we saw in one of the validation steps above, mean effect may be present even in situations where only variance effect is simulated. This is because mean and variance effects are innately correlated. A number of studies confirm this, and a few have attempted to decorrelate this mean-variance effect relationship (@Young2018a, @miao2022).

We will explore this further by looking at the nature of this relationship in two different but related scenarios: when SNP-by-SNP and SNP-by-environment interactions are unmodeled. This is important since our current study will attempt to detect genetic interactions without explicitly modeling them.

... in the case of an unmodeled SNP-by-SNP interaction

```
## Generate data

set.seed(0001)
simrdatvq(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## First check, is there variance effect?

leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value    Pr(>F)
group      8  24.787 < 2.2e-16 ***
      149991
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Is there also mean effect?

summary(lmer(qt ~ snp + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ snp + (1 | id)
   Data: dat

REML criterion at convergence: 314363.3

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.6030 -0.5584 -0.0019  0.5618  3.3327

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8286   0.9103
 Residual             0.2004   0.4477
Number of obs: 150000, groups:  id, 50000

Fixed effects:
```

```
             Estimate Std. Error t value
(Intercept) -0.000572   0.004678  -0.122
snp          0.158817   0.009953  15.957

Correlation of Fixed Effects:
    (Intr)
snp -0.426
```

Highly significant variance effect as expected. But more importantly, also highly significant mean effect underscoring our prior observation on the correlation of mean and variance effects.

… in the case of an unmodeled SNP-by-E interaction

```
## Generate data

set.seed(0002)
simrdatvE(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## First check, is there variance effect?

leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
```

```
Levene's Test for Homogeneity of Variance (center = median)
          Df F value    Pr(>F)
group      8  539.32 < 2.2e-16 ***
      149991
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Is there also mean effect?

summary(lmer(qt ~ snp + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ snp + (1 | id)
   Data: dat
```

```
REML criterion at convergence: 367778

Scaled residuals:
    Min      1Q  Median      3Q     Max
-7.3077 -0.4948 -0.0004  0.4959  7.5309

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8005   0.8947
 Residual             0.3385   0.5818
Number of obs: 150000, groups:  id, 50000

Fixed effects:
             Estimate Std. Error t value
(Intercept) -0.006798   0.004730  -1.437
snp          0.001740   0.010006   0.174

Correlation of Fixed Effects:
    (Intr)
snp -0.428
```

Once, again we see a highly significant variance effect. But unlike the prior example, the t statistic here is quite small and sits at the boundary of the critical value so it's hard to see if SNP has a main effect. In order to know for sure, we need to store the results of `lmer` as an object then use `confint` to generate the confidence interval around the estimate. Note that this takes a bit of time to run, but we should see that the effect is not significant in this case.

SNP effect on phenotypic variance may be captured in a SNP-by-time interaction term in a linear mixed effects model

One of the hypotheses in the study is the variance effect of SNP may appear as SNP-by-time effect. We test this hypothesis in the simulation below. In the first instance we test our hypothesis when phenotypic variance is driven by SNP-by-SNP interaction. And in the second instance when it's driven by SNP-by-E.

```r
## Generate dataset

simrdatvq(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## Will SNP-by-time term capture the variance effect of the SNP?
```

```
summary(lmer(qt ~ time + snp + snp*time + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ time + snp + snp * time + (1 | id)
   Data: dat

REML criterion at convergence: 314243.6

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.6381 -0.5627 -0.0018  0.5604  4.2486

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8309   0.9115
 Residual             0.1999   0.4471
Number of obs: 150000, groups:  id, 50000

Fixed effects:
              Estimate Std. Error t value
(Intercept)  0.0012749  0.0047089   0.271
time        -0.0003993  0.0007506  -0.532
snp          0.1566088  0.0099851  15.684
time:snp     0.0026788  0.0015916   1.683

Correlation of Fixed Effects:
         (Intr) time   snp
time      0.106
snp      -0.426 -0.045
time:snp -0.045 -0.426  0.106
```

No, it does not capture it even when we modeled a very strong effect size. It appears the variance effect is absorbed as mean effect consistent with our previous simulation.

```
## Generate dataset

simrdatvE(nsnp = 1, N = 50000, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda,
         maf = 0.1, bsnp = 0.8)

## Will SNP-by-time term capture the variance effect of the SNP?
```

```r
summary(lmer(qt ~ time + snp + snp*time + (1|id), data = dat))
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: qt ~ time + snp + snp * time + (1 | id)
   Data: dat

REML criterion at convergence: 367883.8

Scaled residuals:
    Min      1Q  Median      3Q     Max
-8.2203 -0.4948 -0.0012  0.4981  6.9518

Random effects:
 Groups   Name        Variance Std.Dev.
 id       (Intercept) 0.8001   0.8945
 Residual             0.3389   0.5821
Number of obs: 150000, groups:  id, 50000

Fixed effects:
              Estimate Std. Error t value
(Intercept) -0.0070750  0.0047636  -1.485
time        -0.0011928  0.0009765  -1.221
snp          0.0046194  0.0101745   0.454
time:snp     0.0005904  0.0020857   0.283

Correlation of Fixed Effects:
         (Intr) time   snp
time      0.137
snp      -0.424 -0.058
time:snp -0.058 -0.424  0.137
```

SNP-by-time again does not capture variance effect. But unlike the previous example, the variance effect is also not absorbed by the main effect.

Still need to do SNP-by-time variance effect simulation to answer if variance effect can be picked up by this term in the case of an unmodeled SNP-by-SNP or SNP-by-E interaction.

Power curves

The aim of this section is to evaluate how powered our study is in detecting mean and variance effects. We will perform power simulations in the case of correlated phenotypes consistent with our study design. We evaluate power on differing levels of N, MAF and beta in 3 broad scenarios:

1. Power of a linear model to detect SNP effect on phenotypic mean;
2. Power of Levene's Test (Brown-Forsythe) to detect SNP effect on phenotypic variance modeled via SNP-by-SNP interaction; and
3. Power of Levene's Test (Brown-Forsythe) to detect SNP effect on phenotypic variance modeled via SNP-by-E interaction.

For variance effects modeled by SNP-by-SNP interaction, we still need to evaluate power when MAFs are discordant. And for SNP-by-E interaction, potentially re-evaluate power in a range of known distributions. Note that @miao2022 has shown that power does not change much across methods so might not be worth doing unless we do it for trajGWAS as this may not have been done before.

First, we setup 2 functions designed to speed up our simulation by running them in parallel using multiple cores.

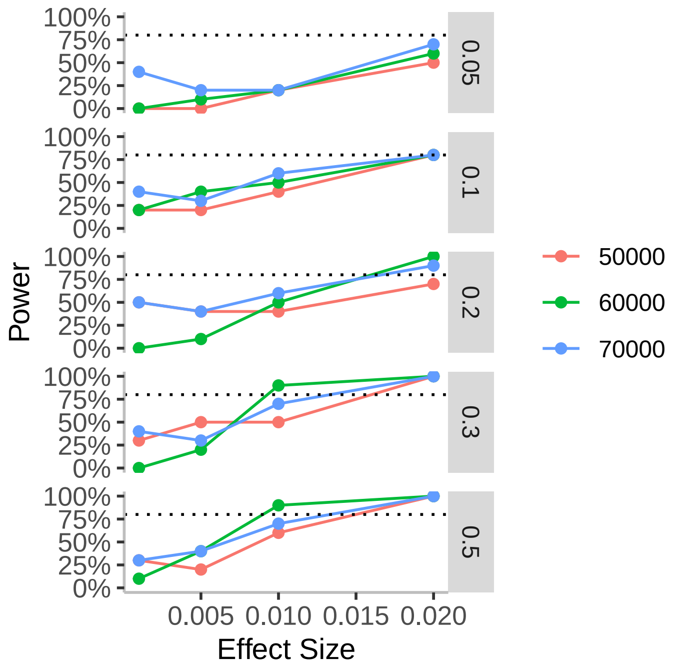
Setup functions for parallel processing

```r
## Load libraries

suppressMessages(library(parallel))
suppressMessages(library(tidyr))
suppressMessages(library(dplyr))

## Create function to generate data and perform testing

runsim <- function(N, maf, bsnp, type){
  if (type == 'mean'){
    dat <- simrdatm(nsnp = 1, N, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda, maf, bsnp)
    lm(qt~snp+snp*time,data = dat)
    }
  else if (type == 'ggi'){
    dat <- simrdatvq(nsnp = 1, N, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda, maf, bsnp)
    leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
    }
  else if (type == 'gei'){
    dat <- simrdatvE(nsnp = 1, N, t = 3, TIME = c(-3, 0, 1), Lambda = Lambda, maf, bsnp)
    leveneTest(dat$qt~as.factor(dat$snp)*as.factor(dat$time), center = median)
    }
  }

## Create another function to execute runsim nsim times via mclapply

parsim <- function(N, maf, bsnp, type, nsim) {
  mclapply(1:nsim, FUN = function(x) runsim(N, maf, bsnp, type), mc.cores = 7)
}
```

SNP effect on phenotypic mean: linear model

```r
## Run linear regression using parsim nested in for loop
# @Para: n (double) is the total number of individuals, m (double) is the minor
# allele frequency or maf, b (double) is the beta or effect size simulated,
# type (string) has 3 options calling each of the 3 main functions [simrdatm.R,
# simrdatvq.R and simrdatvE.R] to generate snps with mean, snp-by-snp and snp-
# by-E effects, repectively, and nsim (double) is the number of iterations per
# n, m and b combination. @Out: model (list lm).

set.seed(0003)
Ns <- c(50000,60000,70000)
mafs <- c(0.05, 0.1, 0.2, 0.3, 0.5)
bs <- c(0.001, 0.005, 0.01, 0.02)
type <- 'mean'
nsim <- 10
mods <- list()

RNGkind("L'Ecuyer-CMRG")

for (n in Ns){
  for (m in mafs){
    for (b in bs){
      tmod <- parsim(n, m, b, type, nsim)
      mods <- c(mods, tmod)
    }
  }
}
```

```r
## Create simres table then execute mclapply to extract p values from object lm

simres <- crossing(N=Ns, maf=mafs, b=bs, s=1:nsim, p=NA)

simres$p <- mclapply(1:length(mods), function(x){
  summary(mods[[x]])[["coefficients"]][[14]]
}, mc.cores = 10)

simres$p <- as.numeric(simres$p)

## Generate power table

power <-  simres %>%
  mutate(success = if_else(p < .05, 1, 0)) %>%
  group_by(N,maf,b) %>%
  summarise(pow = sum(success) / nsim, .groups = 'drop')

## Plot power curve

ggplot(power, aes(x = b, y = pow, color = as.factor(N))) +
  geom_line(linetype = 1) +
  geom_point() +
  geom_hline(yintercept = .80, linetype = 3) +
  scale_y_continuous(labels = scales::percent, limits = c(0, 1)) +
  theme(legend.position="right", legend.title = element_blank(),
        plot.title = element_text(size = 10),
        panel.background = element_blank(),
        axis.line = element_line(colour = "grey"),
        panel.spacing.y = unit(0.25, "cm"),
        panel.spacing.x = unit(0.25, "cm"),
        axis.text.y = element_text(size = 10),
        axis.text.x = element_text(size = 10),
        plot.margin = unit(c(0.1, 1, 0, 1), "inches")) +
  labs(x = "Effect Size", y = "Power") +
  facet_grid(rows = vars(maf))
```

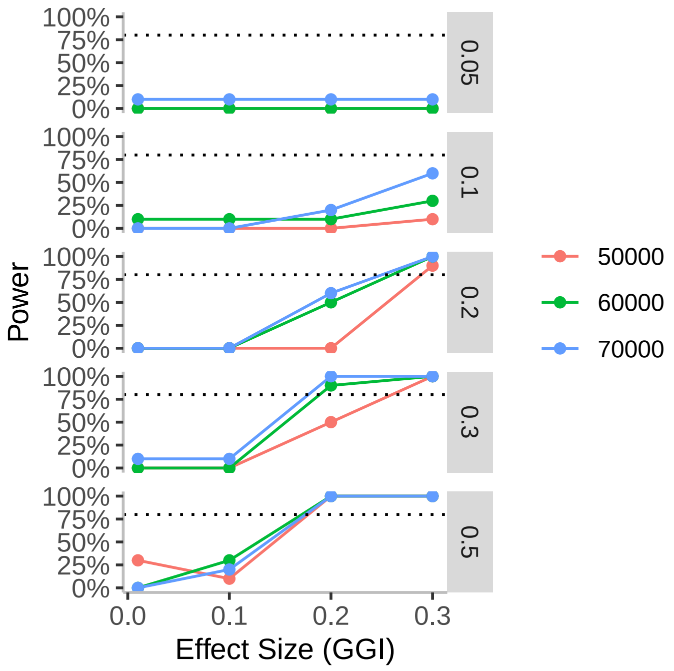


We can reasonably expect to detect betas as low as 0.015 at MAF 0.1 and above.

SNP effect on phenotypic variance (SNP-by-SNP): Levene Test Brown-Forsythe

```
## Run Levene Test using parsim nested in for loop
# @Para: same as above. @Out: list.

Ns <- c(50000,60000,70000)
mafs <- c(0.05, 0.1, 0.2, 0.3, 0.5)
bs <- c(0.01, 0.1, 0.2, 0.3)
type <- 'ggi'
nsim <- 10
pvals <- NULL

RNGkind("L'Ecuyer-CMRG")

for (n in Ns){
  for (m in mafs){
    for (b in bs){
      pval <- parsim(n, m, b, type, nsim)
      pvals <- c(pvals, pval)
    }
```

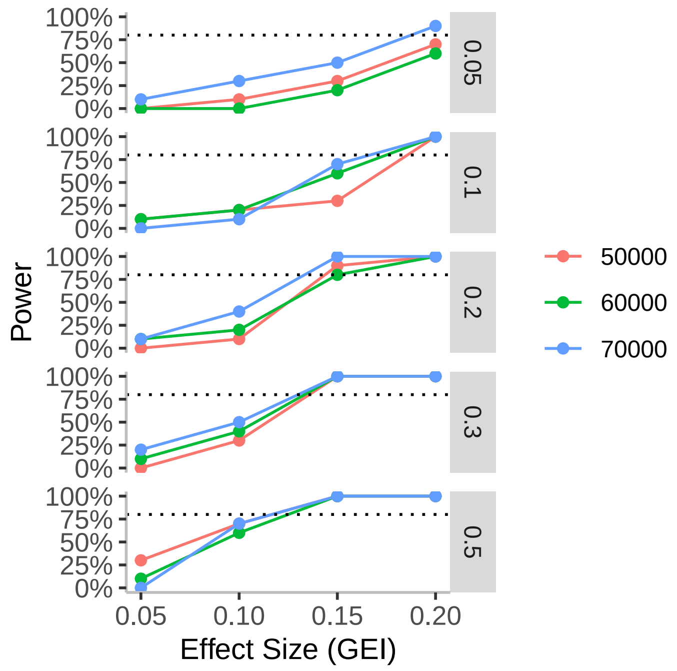
```r
  }
}

## Create simres table then execute mclapply to extract p values from object lm

simres2 <- crossing(N=Ns, maf=mafs, b=bs, s=1:nsim, p=NA)

simres2$p <- mclapply(1:length(pvals), function(x){
  pvals[[x]][[3]][[1]]}, mc.cores = 7)

simres2$p <- as.numeric(simres2$p)

## Generate power table

power2 <-  simres2 %>%
  mutate(success = if_else(p < .05, 1, 0)) %>%
  group_by(N,maf,b) %>%
  summarise(pow = sum(success) / nsim, .groups = 'drop')

## Plot power curve

ggplot(power2, aes(x = b, y = pow, color = as.factor(N))) +
  geom_line(linetype = 1) +
  geom_point() +
  geom_hline(yintercept = .80, linetype = 3) +
  scale_y_continuous(labels = scales::percent, limits = c(0, 1)) +
  theme(legend.position="right", legend.title = element_blank(),
        plot.title = element_text(size = 10),
        panel.background = element_blank(),
        axis.line = element_line(colour = "grey"),
        panel.spacing.y = unit(0.25, "cm"),
        panel.spacing.x = unit(0.25, "cm"),
        axis.text.y = element_text(size = 10),
        axis.text.x = element_text(size = 10),
        plot.margin = unit(c(0.1, 1, 0, 1), "inches")) +
  labs(x = "Effect Size (GGI)", y = "Power") +
  facet_grid(rows = vars(maf))
```



We can only detect SNPs with effect on phenotypic variance at around 0.2 between MAF of 0.2 and 0.3. This means that we are unlikely to find variance effects due to SNP-by-SNP interaction, since such interactions are estimated to be very low [@hivert2021].

SNP effect on phenotypic variance (SNP-by-E): Levene Test Brown-Forsythe

```r
## Run Levene Test using parsim nested in for loop
# @Para: same as above. @Out: table.

Ns <- c(50000,60000,70000)
mafs <- c(0.05, 0.1, 0.2, 0.3, 0.5)
bs <- c(0.05, 0.1, 0.15, 0.2)
type <- 'gei'
nsim <- 10
pvals <- NULL

RNGkind("L'Ecuyer-CMRG")

for (n in Ns){
  for (m in mafs){
    for (b in bs){
      pval <- parsim(n, m, b, type, nsim)
      pvals <- c(pvals, pval)
    }
  }
}
## Create simres table then execute mclapply to extract p values from object lm

simres3 <- crossing(N=Ns, maf=mafs, b=bs, s=1:nsim, p=NA)

simres3$p <- mclapply(1:length(pvals), function(x){
  pvals[[x]][[3]][[1]]}, mc.cores = 7)

simres3$p <- as.numeric(simres3$p)

## Generate power table

power3 <-  simres3 %>%
  mutate(success = if_else(p < .05, 1, 0)) %>%
  group_by(N,maf,b) %>%
  summarise(pow = sum(success) / nsim, .groups = 'drop')

## Plot power curve

ggplot(power3, aes(x = b, y = pow, color = as.factor(N))) +
  geom_line(linetype = 1) +
  geom_point() +
  geom_hline(yintercept = .80, linetype = 3) +
  scale_y_continuous(labels = scales::percent, limits = c(0, 1)) +
  theme(legend.position="right", legend.title = element_blank(),
        plot.title = element_text(size = 10),
        panel.background = element_blank(),
        axis.line = element_line(colour = "grey"),
        panel.spacing.y = unit(0.25, "cm"),
        panel.spacing.x = unit(0.25, "cm"),
        axis.text.y = element_text(size = 10),
        axis.text.x = element_text(size = 10),
        plot.margin = unit(c(0.1, 1, 0, 1), "inches")) +
  labs(x = "Effect Size (GEI)", y = "Power") +
  facet_grid(rows = vars(maf))
```



We can detect betas around 0.15 at MAF 0.2 and above, so a little better than when variance is driven by SNP-by-SNP interaction.

So far, we have simulated SNP mean and variance effects affecting all time-points, but what is the expected power of our study if SNP mean or variance effects are present at only one time point? Still need to do this.

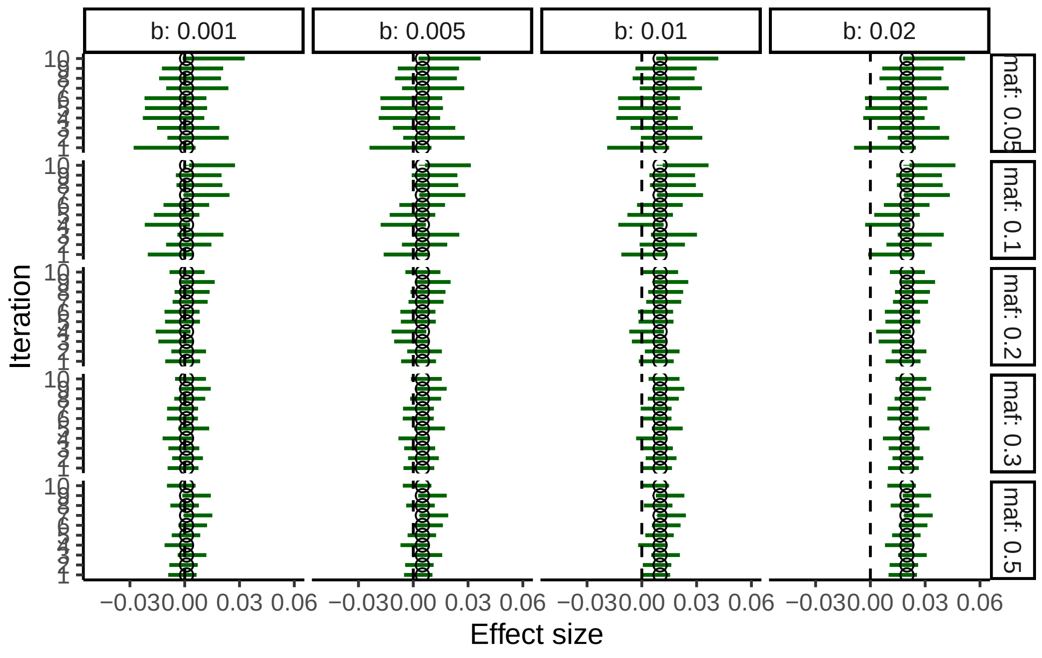
Beta estimates - DISREGARD FOR NOW, NEED TO RE-RUN SIMULATION WITH DIFF SEED

Apart from power, we also look at the accuracy of the beta estimates from our linear model.

```r
## Extract beta estimates and calculate CIs from mods

simres$best <- mclapply(1:length(mods), function(x){
  mods[[x]][["coefficients"]][[2]]
}, mc.cores = 7)

simres$LL <- mclapply(1:length(mods), function(x){
  confint(mods[[x]])[[2]]
}, mc.cores = 7)
```

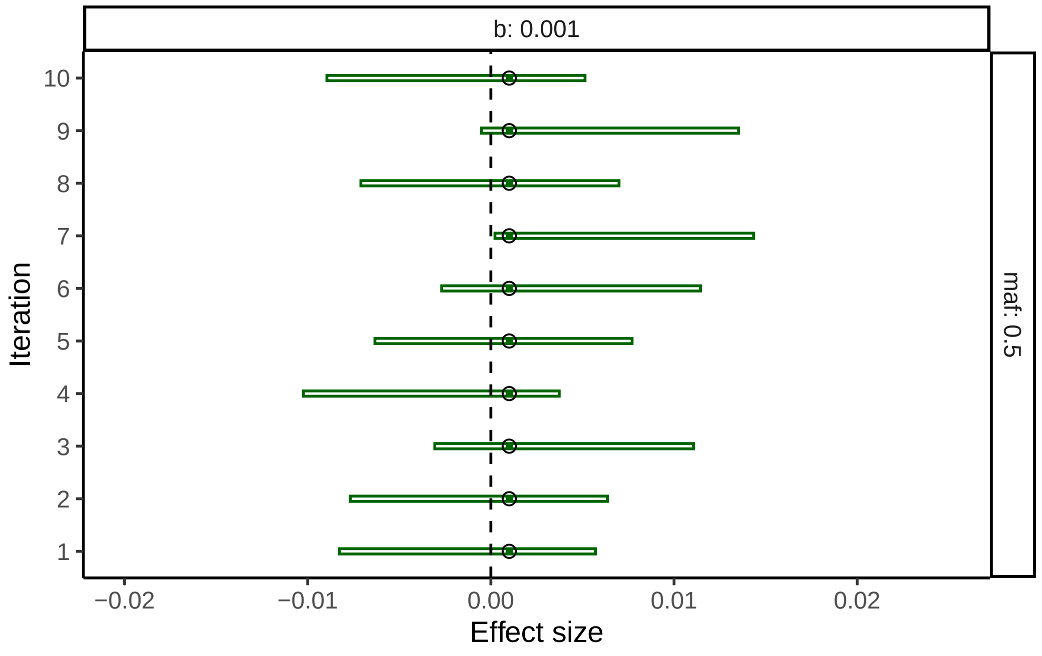
```r
simres$UL <- mclapply(1:length(mods), function(x){
  confint(mods[[x]])[[6]]
}, mc.cores = 7)

simres$best<-as.numeric(simres$best)
simres$LL<-as.numeric(simres$LL)
simres$UL<-as.numeric(simres$UL)

# plot beta with CIs

ggplot(simres[simres$N == Ns[2],], aes(x = b, y = s)) +
  geom_crossbar(aes(xmin = LL, xmax = UL), width = 0.1, color = "darkgreen", size = 0.5) +
  geom_point(size = 2, shape = 21) +
  geom_vline(xintercept = 0, color = "black", linetype = "dashed") +
  labs(x = "Effect size", y = "Iteration") +
  facet_grid(rows = vars(maf), cols = vars(b), labeller = label_both) +
  theme_classic() +
  coord_cartesian(xlim = c(-0.05, 0.06)) +
  scale_y_continuous(breaks = seq(1, 10, 1)) +
  guides(fill = "none") +
  theme(panel.spacing.x = unit(0.1, "cm"), panel.spacing.y = unit(0.1, "cm"))
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



In looking at the above graph, we can see that our estimates are unbiased. But this is not the case for all beta levels. We zoom in on the graph where beta = 0.001, MAF = 0.5.

```r
ind1 <- simres$N == Ns[2]
ind2 <- simres$b == 0.001
ind3 <- simres$maf == 0.5

ggplot(simres[c(ind1&ind2&ind3),], aes(x = b, y = s)) +
  geom_crossbar(aes(xmin = LL, xmax = UL), width = 0.1, color = "darkgreen", size = 0.5) +
  geom_point(size = 2, shape = 21) +
  geom_vline(xintercept = 0, color = "black", linetype = "dashed") +
  labs(x = "Effect size", y = "Iteration") +
  facet_grid(rows = vars(maf), cols = vars(b), labeller = label_both) +
  theme_classic() +
  coord_cartesian(xlim = c(-0.02, 0.025)) +
  scale_y_continuous(breaks = seq(1, 10, 1)) +
  guides(fill = "none") +
  theme(panel.spacing.x = unit(0.1, "cm"), panel.spacing.y = unit(0.1, "cm"))
```



Here we can see that at very low beta the estimates can sometimes turn significant but sign discordant as in the case for iterations 1 and 9 above.

Still need to plot variance effect from Levene Test. Figure out how to extract between and within sum of squares from the function to derive the effect size, η^2.

Phenotypic variance explained (PVE)

In this simulation, we use only 1 time point instead of 3 for simplicity (results are the same).

```
## Load library for plotting

suppressMessages(library(patchwork))

## Simulate snp effect on phenotype then plot density curve

maf <- 0.3
snp = rbinom(N, 2, maf)
phenotype = snp * 0.8 + rnorm(N) * sqrt(1-0.8^2)
dat <- data.frame(phenotype,snp)
p3 <- ggplot(dat,aes(x=phenotype, color=as.factor(snp))) + geom_density() +
  theme_classic()
```

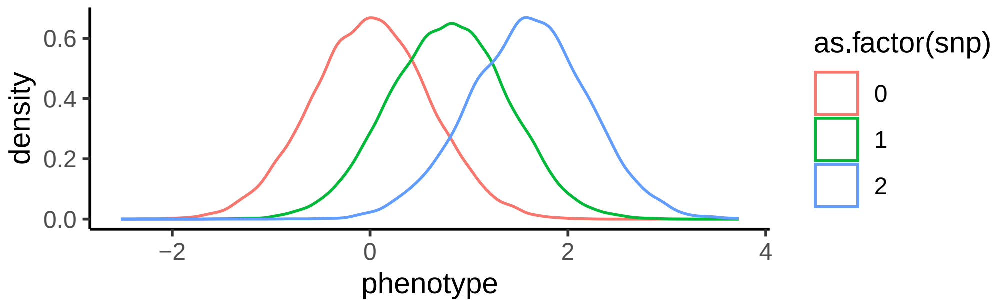
```
# Calculate mean and variance effect per snp level

dat %>% group_by(dat$snp) %>% summarise(var = var(phenotype), mean = mean(phenotype))
```

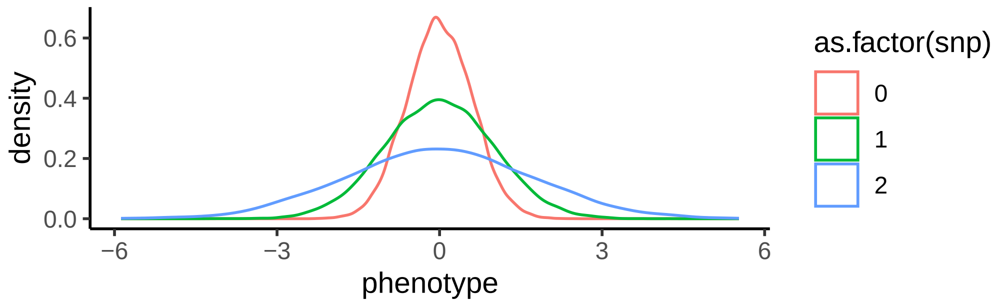
```
# A tibble: 3 x 3
  `dat$snp`   var    mean
      <int> <dbl>   <dbl>
1         0 0.357 0.00453
2         1 0.363 0.794
3         2 0.363 1.60
```

```
## Simulate snp effect on phenotypic variance via SNP-by-E then plot density curve

snp = rbinom(N, 2, maf)
phenotype = snp*rnorm(N, 0, 1) * 0.8 + rnorm(N) * sqrt(1-0.8^2)
dat <- data.frame(phenotype,snp)
p4 <- ggplot(dat,aes(x=phenotype, color=as.factor(snp))) + geom_density() +
  theme_classic()

# Calculate mean and variance effect per snp level

dat %>% group_by(dat$snp) %>% summarise(var = var(phenotype), mean = mean(phenotype))
```

```
# A tibble: 3 x 3
  `dat$snp`   var      mean
      <int> <dbl>     <dbl>
1         0 0.364 -0.000233
2         1 0.998 -0.00403
3         2 2.95  -0.00761
```

```
## Combine both plots

p3 / p4
```





PVE in the case of mean-only effect is clearly visible, where the mean increases as a function of allele count (top graph). In comparison, the PVE is 0 when variance-only effect is simulated: ***there is no increase or decrease in phenotypic mean across groups*** (bottom graph). Given the above, we can see that the concept of PVE does not hold when the outcome is variance. In brief, we can show that

$$\sigma_Y^2 = \beta\mathbf{SNP} + \epsilon \tag{2}$$

$$Var(\sigma_Y^2) = \beta^2 Var(\mathbf{SNP}) + Var(\epsilon) \tag{3}$$

$$2\sigma_Y^4 = \beta^2 Var(\mathbf{SNP}) + Var(\epsilon) \tag{4}$$

For a full derivation of $Var(\sigma_Y^2) = 2\sigma_Y^4$, see: https://en.wikipedia.org/wiki/Variance#Distribution_of_the_sample_variance.

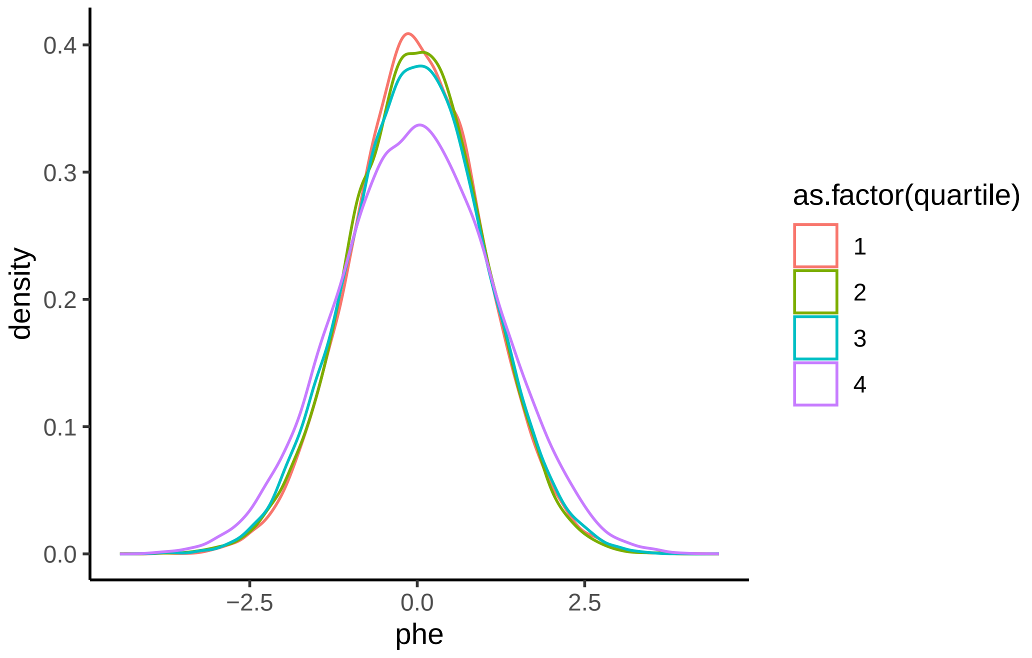
! Important

We do not observe a mean-variance effect relationship here because we used `simrdatvE.R` function in generating our phenotype. From our earlier simulation, we know that SNP-by-E interaction is less likely to generate mean effect.

vPGI

We generate 10 independent SNPs and a single set of normally distributed environmental factor (E). We generate 1 set of E because we use a single time point, similar to the prior simulation. But here instead of 1 SNP we model E to interact with each of the 10 SNPs. Each interaction term is simulated to have between 0 and 0.1 beta.

```
## Generate SNP-by-E matrix

N <- 100000
nsnps <- 10
maf <- 0.3
snpEmat <- matrix(rbinom(N*nsnps,2,maf) * rep(rnorm(N,0,1), times = nsnps), N, nsnps)

## Set SNP-by-E betas

betas <- runif(nsnps,0,0.1)

## Simulate phenotype and vPGI

phe <- snpEmat %*% betas + rnorm(N,0,1) * sqrt(1-sum(betas^2))
pgi <- snpEmat %*% betas

# Plot phenotype per vPGI quartile

quartile <- ntile(abs(pgi), 4)
df <- data.frame(phe,pgi,quartile)
ggplot(df,aes(x=phe, color=as.factor(quartile))) + geom_density() + theme_classic()
```



```
# Calculate mean and variance effect per vPGI quartile

df %>% group_by(df$quartile) %>% summarise(var = var(phe) , mean = mean(phe))
```

```
# A tibble: 4 x 3
  `df$quartile`   var      mean
          <int> <dbl>     <dbl>
1             1 0.965  0.0178
2             2 0.981 -0.000298
3             3 1.05  -0.000283
4             4 1.35   0.00609
```

We see a similar result as the prior example (bottom graph). The effect of multiple SNPs on phenotypic variance is expressed as a scale effect. This is expected in the case where $Cov(SNP_i, SNP_j) = 0$ and $i \neq j$ since,

$$\sigma_Y^2 = \beta_1 \mathbf{SNP}_1 + \beta_2 \mathbf{SNP}_2 + ... + \beta_n \mathbf{SNP}_n + \epsilon \tag{5}$$

$$Var(\sigma_Y^2) = \beta_1^2 Var(\mathbf{SNP}_1) + \beta_2^2 Var(\mathbf{SNP}_2) + ... + \beta_n^2 Var(\mathbf{SNP}_n) + Var(\epsilon) \tag{6}$$

$$2\sigma_Y^4 = \sum_1^n \beta_n^2 Var(\mathbf{SNP}_n) + Var(\epsilon) \tag{7}$$

Based on the above, we recommend that vPGI should be used as a way to quantify the magnitude of dispersion of a phenotype around a mean. It should not be used to predict differences in phenotypic means across vPGI levels, as in the use case for PVE. That is, the total contribution of a set of independent SNPs on phenotypic variance should be viewed as the **phenotypic dispersion explained** by SNPs. By extension, no heritability estimates can be gathered from methods like LDSC score regression which relies on a similar derivation, where the effect of independent SNPs on pheynotypic variance is summated.